Quick Swap: Faster Settlement in Cross-Chain Atomic Swaps

Quick Swap: Faster Settlement in Cross-Chain Atomic Swaps

Subhra Mazumdar
Indian Statistical Institute
Kolkata, India
subhram_r@isical.ac.in

Abhinandan Sinha
Ahmedabad University
Ahmedabad, India
abhinandan.sinha@ahduni.edu.in

ABSTRACT

Hashed Timelock (HTLC)-based atomic swap protocols enable the exchange of coins between two or more parties without relying on a trusted entity. This protocol is like the *American call option* without premium. It allows the finalization of a deal within a certain period. This puts the swap initiator at liberty to delay before deciding to proceed with the deal. If she finds the deal unprofitable, she just waits for the time-period of the contract to elapse. However, the counterparty is at a loss since his assets remain locked in the contract. The best he can do is to predict the initiator's behavior based on the asset's price fluctuation in the future. But it is difficult to predict as cryptocurrencies are quite volatile, and their price fluctuates abruptly. We perform a game theoretic analysis of HTLC-based atomic cross-chain swap to predict whether a swap will succeed or not. From the strategic behavior of the players, we infer that this model lacks fairness. We discuss the properties of an ideal atomic swap and propose *Quick Swap*, a two-party protocol based on hashlock and timelock that fosters faster settlement of the swap. The parties are required to lock with griefing-premium along with the principal amount. If the party griefs, he ends up paying the premium. If a party finds a deal unfavorable, he has the provision to cancel the swap. Our work is the first to extend *Quick Swap* to a cyclic multi-party atomic swap setting.

KEYWORDS

Cryptocurrencies, Atomic Swap, Hashed Timelock (HTLC), Griefing Attack, Game Theory, Fairness, Faster Settlement

1 INTRODUCTION

While one can easily envision an atomic swap functionality leveraging a trusted server, the blockchain community has put significant efforts into decentralized protocols for atomic swaps [1, 9, 10, 12, 14, 17–19, 22, 23]. Exchange of assets leads to change of ownership, it either succeeds or fails in entirety. Bitcoin-based blockchains primarily leverage on Hashed Timelock Contracts or HTLC [4, 5, 9, 14] for exchanging Bitcoins (BTC) with other cryptocurrencies like Litecoins (LTC), Ether (ETH), ERC Tokens.

We explain a two-party HTLC-based atomic swap protocol with an example shown in Figure 1. *Alice* wants to exchange x_a coins for y_b coins of *Bob*. Both of them have accounts in two different blockchains `Chain-a` and `Chain-b`. *Alice* samples a secret s, generates a hash $H = \mathcal{H}(s)$ and shares it with *Bob* at time t_0. The protocol comprises two phases: *Lock* and *Claim*. *Lock phase* defines the time interval within which the parties have locked their assets in the contracts instantiated in the respective blockchains. *Alice* locks x_a coins in `Chain-a` at time t_1. The coin is locked in a 2-of-2 multi-sig contract where the spending conditions are as follows: *if Bob provides the secret s within t_a units of time, then he claims x_a coins else Alice initiates a refund after t_a elapses.* Once the

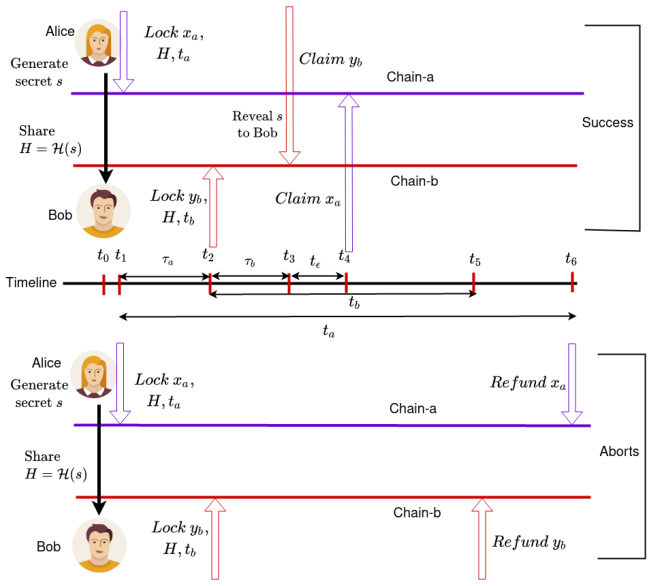


Figure 1: HTLC-based atomic swap. We assume that there is no waiting involved when parties lock their coins and are willing to exchange it as well.

transaction is confirmed in `Chain-a` in the next τ_a units of time, *Bob* locks y_b coins in `Chain-b` at time t_2. He uses the same hash value H in the contract deployed in `Chain-b` for locking his coins. The spending conditions are different: *if Alice provides the secret s within t_b units of time, then she claims y_b coins else Bob initiates a refund after t_b elapses*, where $t_a > t_b$. It takes τ_b units for the lock transaction to be confirmed in `Chain-b`. *Claim phase* signals the period within which the parties claim their assets. In the best case involving zero waiting, *Alice* broadcasts the claim transaction at t_3, releases the preimage s and claims y_b coins from *Bob*. The latter uses the preimage s at broadcasts a transaction to claim x_a coins at time t_4, where $t_4 - t_3 = t_\epsilon$ is the time taken by *Bob* to observe *Alice*'s transaction in `Chain-b`. Once the ownership of the assets changes successfully, an instance of the protocol succeeds.

One of the main disadvantages of HTLC-based atomic swap is that parties are not enforced to settle the transaction. It has already been shown in [8], [2] that atomic swap is equivalent to *American call option* without premium. In *American call option*, the buyer is allowed to exercise the contract no later than the strike time. We illustrate the situation in Figure 2. *Bob* may speculate within duration t_2 to $t_6 - (t_b + \epsilon + t_\epsilon)$, after *Alice* has locked her coins at time t_1. Either he may delay or he may choose to not lock his coins. If *Bob* does not lock his coins, then *Alice*'s coins get locked unnecessarily and she loses her coins in paying fees to miners for successful

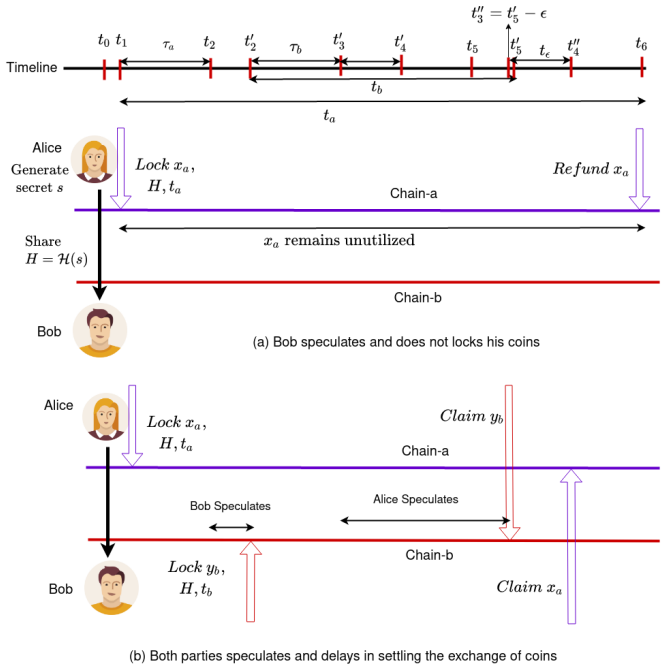


Figure 2: Coins remain unutilized in HTLC-based atomic swap due to (a) Bob speculates the deal and does not lock his coins and (b) Bob speculates, finds t_2' favourable for locking his coins, delaying by $t_2' - t_2$, and Alice speculates, finds $t_3'' = t_5' - \epsilon$ favourable for claiming the coins from Bob

mining of refund transaction. This is termed as *draining attack* [7]. On the other hand, suppose *Bob* finds a favourable at time $t_2' > t_2$, so he delays for $t_2' - t_2$ units. If *Alice* chooses not to delay, then she will claim the coins at time $t_3' = t_2' + \tau_b$. If she doesn't find the deal favourable at time t_3', chooses to speculate and finds the price in her favour at $t_3'' = t_5' - \epsilon$ where $t_5' = t_2' + \tau_b$. She claims y_b coins at time t_3'' and *Bob* gets to claim x_a coins at $t_4'' = t_3'' + t_\epsilon$. Here *Bob* has to wait a duration of $t_4'' - t_4'$ where $t_4' = t_3' + t_\epsilon$. Upon simplification, we observe that he waits $t_3'' - t_3' = t_5' - \epsilon - t_3'$, which is the delay due to *Alice*'s speculation. It may so happen that *Alice* does not want to claim y_b coins, she waits for t_b units to elapse, and *Bob* broadcasts refund transaction at time $t_5' = t_2' + t_b$. *Alice* broadcasts her refund transaction after t_a units elapse, i.e., at time $t_6 = t_1 + t_a$. If *Alice* chooses not to respond, then it leads to *griefing attack* [16]. Coins remain unutilized in either of the blockchains leading to a substantial rise in opportunity cost. Enforcing the parties not to back out from the deal is a major challenge. Additionally, HTLC lacks flexibility as it does not provide an option to cancel the contract when the situation turns out to be unfavorable.

1.1 Contributions

- We model HTLC-based atomic cross-chain swap as a two-player sequential game and analyze the success rate of the protocol as a function of exchange rate and delay.
- We observe that delay in settling the swap drastically reduces the success rate and infer that an ideal model for atomic swap must allow faster settlement of the transaction.
- We discuss the salient features of an ideal atomic swap and propose *Quick Swap*, a hashlock and timelock-based protocol compatible with Bitcoin script. Our protocol is more robust and efficient, allowing easy cancellation of trade and penalizing a party if it griefs.
- We extend our protocol to multi-party cyclic atomic swap. We claim that to date no protocol designed for countering griefing attacks has been generalized to the multi-party case.

2 BACKGROUND

Hashed Time-lock Contract (HTLC) was first discussed in [15]. A payer S want to transfer funds to a payee R, The payee R creates a condition y defined by $y = \mathcal{H}(\tilde{x})$ where $\tilde{x}$ is a random string and $\mathcal{H}$ is a random oracle [3]. The condition y is shared with S. The latter forwards a hashed time-lock contract, defined by $HTLC(S, R, y, b, t)$, to R. It implies that this contract locks b units of fund of party S that can be released to party R only if it releases the correct preimage $\tilde{x}$ for which $y = \mathcal{H}(\tilde{x})$ within timeout t. If R doesn't release $\tilde{x}$ within time t, then S settles the dispute on-chain by broadcasting and b coins get added to $S's$ balance. However, in this process, the funds remain locked in the channel (S, R) and cannot be utilized before t elapses. In this way, R manages to mount *griefing attack* on S, locking a collateral of $b * t$ coins.

3 GAME THEORETIC ANALYSIS

The atomic swap protocol comprises two phase: *Lock Phase* - for the duration t_0 to just before t_3, and *Claim Phase* - from time t_3 onward. It proceeds sequentially, with the assets being locked first and then the assets being claimed in the next phase. Given two parties Alice and Bob, their strategy space consists of the actions *continue* and *stop*. After *Alice* locks x_a coins in `Chain-a`, *Bob* can choose to either *continue*, i.e., lock y_b coins in `Chain-b`, or *stop* and abort from the process. There is no way by which *Bob* can inform *Alice* that he wants to abort, except wait for timeout period t_a to elapse. If *Bob* locks his coins before that, *Alice* can choose continue with the swap by claiming y_b coins and releasing the preimage of H before t_b elapses. If she choose to *stop*, she will wait for t_b to elapse and *Bob* initiates a refund after that.

3.1 Basic Setup

We denominate coins locked by *Bob* in terms of *Alice*'s coins. Thus we express y_b coins as $x(y_b, t)$, where the price is decided based on the exchange rate prevailing at time t. At time $t_0 \approx t_1$, $x(y_b, t_1)$ is the price of *Bob*'s coins decided to be exchanged for x_a coins of *Alice*. Price of *Bob*'s coins at any time t follow a geometric Brownian motion [6].

$$\ln \frac{x(y_b, t+\tau)}{x(y_b, t)} = \left(\mu - \frac{\sigma^2}{2}\right)\tau + \sigma(W_{t+\tau} - W_t) \tag{1}$$

where W follows a Wiener Process [11], with drift μ and variance σ^2. Given Eq.1, the Expected price of $x(y_b, t)$ at time $t + \lambda$, $(\mathcal{E}(x(y_b, t), \lambda)$, Probability density function of $x(y_b, t)$ at time $t+\lambda$, $(P(x, x(y_b, t), \lambda))$, and Cumulative density function of $x(y_b, t)$

coins at time $t+\lambda$, $(C(x, x(y_b,t),\lambda))$ is expressed as follows:

$$\mathcal{E}(x(y_b,t),\lambda) = \mathbb{E}[x(y_b,t+\lambda)|x(y_b,t)] = x(y_b,t)e^{\mu\lambda}$$

$$\begin{aligned} P(x,x(y_b,t),\lambda) &= \mathbb{P}[x(y_b,t+\lambda)=x|x(y_b,t)] \\ &= \frac{e^{-\left(\frac{\ln\frac{x}{x(y_b,t)}-\left(\mu-\frac{\sigma^2}{2}\right)\tau}{2\tau\sigma^2}\right)^2}}{\sqrt{2\pi\tau}\sigma x} \\ C(x,x(y_b,t),\lambda) &= \mathbb{C}[x(y_b,t+\lambda)\le x|x(y_b,t)] \\ &= \frac{erfc\left(\frac{\ln\frac{x}{x(y_b,t)}-\left(\mu-\frac{\sigma^2}{2}\right)\tau}{\sqrt{2\tau}\sigma}\right)}{2} \end{aligned} \tag{2}$$

where $erfc$ is the complementary error function. $erfc(x)$ is defined as:

$$erfc(x) = \frac{2}{\Pi}\int_x^\infty e^{t^2}\,dt \tag{3}$$

The expressions are taken from [20]. We assume both parties know each other's parameters. The rest of the notations are defined in Table 1.

Notation	Description
x_a	Coins possessed by *Alice* or **A**
y_b	Coins of *Bob* or **B** that **A** decides to buy for x_a coins at time t_0
τ_a	Time taken for a transaction to get confirmed in `Chain-a`
τ_a	Time taken for a transaction to get confirmed in `Chain-b`
t_a	HTLC forwarded by **A** to **B** in `Chain-a`, locking x_a, expires
t_b	HTLC forwarded by **B** to **A** in `Chain-b`, locking y_b, expires
t_ϵ	Propagation delay
ϵ	Short time gap
u_a	Success premium of **A**, if swap succeeds
u_b	Success premium of **B**, if swap succeeds
$x(y_b,t)$	Price of y_b coins in terms of x_a at time a given time t
r_a	Time discounting factor of **A**
r_b	Time discounting factor of **B**
f_a	Transaction fee in `Chain-a`
f_b	Transaction fee in `Chain-b`
μ	Wiener process drift
σ^2	Wiener process variance
$\mathcal{E}(x(y_b,t),\lambda)$	Expected price of $x(y_b,t)$ at time $t+\lambda$, also expressed as $\mathbb{E}[x(y_b,t+\lambda),x(y_b,t)] = x(y_b,t)e^{\mu\lambda}$
$P(x,x(y_b,t),\lambda)$	Probability density function at time $t+\lambda$ given that price at time t is $x(y_b,t)$
$C(x,x(y_b,t),\lambda)$	Cumulative density function at time $t+\lambda$ given that price at time t is $x(y_b,t)$

Table 1: Notations used in the paper

3.2 Game Model

We model the interaction between two entities *Alice*, denoted as **A**, and *Bob*, denoted as **B**, as a sequential game Γ_{swap} similar to the one discussed in [20]. We assume that all the miners in the blockchains `Chain-a` and `Chain-b` are honest, and only **A** and **B** are the strategic players. **A** initiates the lock phase at time t_1, followed by **B** choosing either to lock y_b coins or abort. If **B** doesn't lock his coins before t_1+t_a elapses, then the swap stands canceled. If **B** locks his coins at t_2, then **A** makes the next move, choosing either to claim y_b coins or abort at t_3. If **A** claims her coins (or aborts), **B** follows her action.

Definition 3.1. The extensive-form game Γ_{swap}, is defined as:

- The set of players $N = \{\mathbf{A},\mathbf{B}\}$
- The set of actions for player **A**, $S_\mathbf{A} = \{\text{continue, stop }\}$.
- The set of actions for player **B**, $S_\mathbf{B} = \{\text{continue, stop }\}$.
- The payoff function $u_k : S\times\mathbb{N}\to\mathbb{R}$ for any player $k\in\{\mathbf{A},\mathbf{B}\}$, where $S = S_\mathbf{A}\times S_\mathbf{B}$, is denoted as $u_{k_{s,t}}$ where $s\in S$ and $t\in\mathbb{N}$. It specifies the payoff the player k would get at time $t\in\mathbb{N}$, if the players chose their action as in S.

We explain the model with respect to Figure 2, using backward induction to calculate the payoff of **A** and **B** for the game Γ_{swap}. If **A** locks coins at time t_1, then in no waiting case, **B** locks his coins at time $t_1+\tau_a$. However, the exchange rate at $t_1+\tau_a$ may not be favorable and **B** may choose to wait. If we label the strike time of **B** as t_2, then he speculates in the duration $t_2-t_1\in[\tau_a, t_a-(t_b-\epsilon+t_\epsilon)]$ before choosing to lock coins. After **B** locks his coins at t_2, **A** is expected to lock his coins at $t_2+\tau_b$ under no waiting situation. However, **A** has the liberty to speculate in the duration $t_3-t_2\in[\tau_b, t_b-\epsilon]$, where t_3 is label assigned to **A**'s strike time, and then proceed with the swap.

3.2.1 Preference Structure. The extensive form of the game is shown in Fig.4. We perform a backward induction, starting from t_3 i.e, the time at which **A** decides whether to initiate the claim phase or abort. At time t_2 and time t_1, each party makes a decision, choosing either to continue or stop depending on the situation.

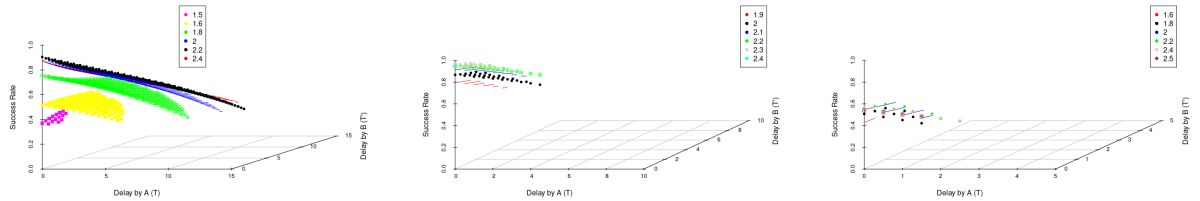
The claim phase starts at time t_3, when **A** decides whether to *continue* and reveal preimage of H, or *stop* for t_b to elapse. If **A** decides to claim coins, then a rational **B** will claim the coins as well. We define the utility or payoff for each strategy. If **A** chooses to continue at t_3, then the time taken for the redeem transaction to get confirmed in `Chain-b` is τ_b. We multiply payoff of **A** upon continuing with a factor $(1+u_a)$, where u_a is the success premium (or u_b for **B**) to emphasize that rational parties would prefer to swap their assets rather than abort. The utility of **A** is expressed with their time discounted for duration τ_b. Similarly, when **B** claims asset in `Chain-a` at time t_3+t_ϵ, time taken for the transaction to be confirmed is τ_a, hence the utility is expressed with their time discounted for duration $t_\epsilon+\tau_a$.

$$\begin{aligned} u_{A_{continue},t_3} &= (1+u_a)\mathcal{E}(x(y_b,t_3),\tau_b)e^{-r_a\tau_b} - f_b \\ u_{B_{continue},t_3} &= (1+u_b)x_a e^{-r_b(\tau_a+t_\epsilon)} - f_a \end{aligned} \tag{4}$$

where $t_3 = t_2+\tau_b+T$, and $T\in[0, t_b-\tau_b-\epsilon]$ being the delay by **A** before she decides to claim the coins. If **A** decides to *stop*, then **B** has to abort as well. $T'\in[0, t_a-(t_b-\epsilon+t_\epsilon)-\tau_a]$ is the delay by **B** before choosing to lock coins after **A** has locked x_a coins. If $t_2 = t_1+\tau_a+T'$, time left before **A** can withdraw her coins is $t_a-\tau_b-T-T'$. Time left before **B** can withdraw his coins is t_b-T. The utility is expressed as: $u_{A_{stop},t_3} = x_a e^{-r_a(t_a-\tau_b-T-T')} - f_a$ and $u_{B_{stop},t_3} = \mathcal{E}(x(y_b,t_3),t_b-T)e^{-r_b(t_b-T)} - f_b$. **A** will decide on *continue* over *stop* at t_3, if the following condition holds:

$$\begin{aligned} (1+u_a)\mathcal{E}(x(y_b,t_3),\tau_b)e^{-r_a\tau_b} - f_b &\ge x_a e^{-r_a(t_a-\tau_b-T-T')} - f_a \\ or, x(y_b,t_3)e^{\mu\tau_b} &\ge \frac{x_a e^{-r_a(t_a-2\tau_b-T-T')} - (f_a-f_b)e^{r_a\tau_b}}{1+u_a} \\ or, x(y_b,t_3) &\ge \frac{\left(x_a e^{-r_a(t_a-2\tau_b-T-T')} - (f_a-f_b)e^{r_a\tau_b}\right)e^{-\mu\tau_b}}{(1+u_a)} \end{aligned}$$

We derive $x(y_b,t_3)^*$ in terms of x_a for which the above inequality holds. If $x(y_b,t_3)\ge x(y_b,t_3)^*$ then **A** claims the coins.



(a) $u_a = u_b = 0.3, r_a = r_b = 0.005, \sigma = 0.1$ **(b)** $u_a = u_b = 0.3, r_a = r_b = 0.01, \sigma = 0.1$ **(c)** $u_a = u_b = 0.3, r_a = r_b = 0.005, \sigma = 0.2$

Figure 3: Swap Success Rate as a function of delay (T, T') and A's coins (x_a), with different parameter values

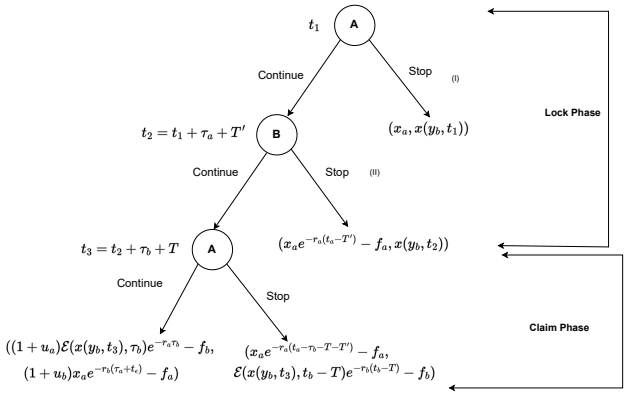


Figure 4: Extensive form of the game Γ_{swap}

In the second round of *lock phase*, **B** has to decide whether he will *continue* (i.e., lock y_b coins) or *stop* after **A** has locked x_a coins. Payoff of **B** at $t_2 = t_1 + \tau_a + T'$ where $T' \in [0, t_a - (t_b - \epsilon + t_\epsilon) - \tau_a]$, can be expressed with their time discounted, expected utility at $t_2 + \tau_b + T$. **B** calculates what will be the probability that price of his coins will rise to $x(y_b, t_3)^*$ within time $\tau_b + T$.

$$u_{B_{continue},t_2} = \frac{\left[1-C\left(x(y_b,t_3)^*, x(y_b,t_2), \tau_b+T\right)\right] u_{B_{continue},t_3}\, dp}{e^{r_b(\tau_b+T)}} + \frac{\int_0^{x(y,t_3)^*} P\left(p, x(y_b,t_2), \tau_b+T\right) u_{B_{stop},t_3}\, dp}{e^{r_b(\tau_b+T)}} \tag{5}$$

$$u_{A_{continue},t_2} = \frac{\int_{x(y_b,t_3)^*}^{\infty} P\left(p, x(y_b,t_2), \tau_b+T\right) u_{A_{continue},t_3}\, dp}{e^{r_a(\tau_b+T)}} + \frac{C\left(x(y_b,t_3)^*, x(y_b,t_2), \tau_b+T\right) u_{A_{stop},t_3}\, dp}{e^{r_a(\tau_b+T)}} \tag{6}$$

If **B** *stops* at t_2, then time left before **A** can unlock x_a coins is $t_a - T'$. The utility is expressed as $u_{A_{stop},t_2} = x_a e^{-r_a(t_a - T')} - f_a$ and $u_{B_{stop},t_2} = x(y_b, t_2)$. **B**'s decision is dependent on how price of y_b evolves until t_3. He will decide to *continue* over *stop*, if the following condition holds: $u_{B_{continue},t_2} \geq x(y_b, t_2)$. We derive the range in which $x(y_b, t_2)$ must lie for the above inequality to hold. **B** will continue if $x_1 < x(y_b, t_2) \leq x_2$. If $x(y_b, t_2) > x_2$ or $x(y_b, t_2) \leq x_1$ then **B** will stop.

In the first round of the lock phase, **A** has to decide whether she wants to lock coins or abort at time t_1. Payoff of **A** can be expressed with their time-discounted, expected utility at $t_2 = t_1 + \tau_a + T'$.

$$u_{A_{continue},t_1} = \frac{\int_{x_1}^{x_2} P\left(p, x(y_b,t_1), \tau_a+T'\right) u_{A_{continue},t_2}\, dp}{e^{r_a(\tau_a+T')}} + \frac{\left[1-C\left(x_2, x(y_b,t_1), \tau_a+T'\right)+C\left(x_1, x(y_b,t_1), \tau_a+T'\right)\right] u_{A_{stop},t_2}\, dp}{e^{r_a(\tau_a+T')}} \tag{7}$$

$$u_{B_{continue},t_1} = \left(\frac{\int_{x_1}^{x_2} P\left(p, x(y_b,t_1), \tau_a+T'\right) u_{B_{continue},t_2}\, dp}{e^{r_b(\tau_a+T')}} + \frac{\int_0^{x_1} P\left(x, x(y_b,t_1), \tau_a+T'\right) u_{B_{stop},t_2}\, dp}{e^{r_b(\tau_a+T')}}\right) \tag{8}$$

If **A** stops at t_1, then no one locks coins and we express the utility as $u_{A_{stop},t_1} = x_a$ and $u_{B_{stop},t_1} = x(y_b, t_1)$. **A**'s decision is based on how price of y_b evolves until t_2. **A** will decide on *continue* over *stop*, if the following condition holds: $u_{A_{continue},t_1} \geq x_a$. We derive the range $(x^*, x^{*'})$, in which x_a must lie for the swap to start.

3.3 Evaluating Success of the Protocol

The success rate *(SR)* of a swap is the probability that the swap succeeds after it has been initiated, i.e. after **A** has locked coins at t_1 [20]. *SR* is defined as function of x_a (**A**'s coins or tokens), delay by **A** (T) before claiming coins, and delay by **B** (T') before locking y_b coins. It is expressed as:

$$SR(x_a, T', T) = \int_{x_1[x_a]}^{x_2[x_a]} P_A(p, T') P_B(p, T)\, dp \tag{9}$$

where $P_A(p, T') = P(p, x(y, t_1), \tau_a + T')$ and $P_B(p, T) = (1 - C(x(y, t_3)[x_a], p, \tau_b + T))$. We set $t_a = 48\ hrs, t_b = 24\ hrs$ as per the standard practice [8]. $t_\epsilon \approx \epsilon = 1\ hr$ and $\tau_a = \tau_b = 3\ hrs$. We select $x_a \in [1, 3]$ given $x(y_b, t_1) = 2$. The parameters T is varied between $[0, 20]$ and T' is varied between $[0, 21]$. The fee f_a and f_b is negligible, so we consider them to be 0. The success premiums $u_a = u_b = 0.3$, time-discounting factor $r_a = r_b$ is chosen from $\{0.005, 0.01\}$, σ is varied between 0.1 and 0.2, and μ is selected from $\{-0.002, 0.002\}$.

Observations: We plot the success rate of the protocol in Fig. 3 (a-c), and observe that the success rate is ≥ 0.9 in Fig. 3 (a-b), and around 0.6 for Fig. 3 (c), when $T = T' = 0$. When T or T' increases, the success rate drops and beyond certain range, it becomes *NA* (not applicable) as $u_{A_{continue},t_1} < x_a$. **A** has knowledge of x_a before

the swap starts but the delay T' and T depends on the volatility of the exchange rate, and not under **A**'s control at t_1. In the worst case, T' and T can be very high leading to non-utilization of both **A**'s and **B**'s coins. Thus **A** will be discouraged to proceed as *participation constraint* might be violated.

The flaw in the protocol is that either of the parties can speculate and delay in settling the transaction. They are not penalized for their behavior but the counterparty suffers a loss due to a rise in the opportunity cost of locked coins. Our objective is to ensure that the parties settle the transaction faster, either choosing to continue or cancel the deal. We discuss the properties of an ideal atomic swap in the next section.

4 AN IDEAL ATOMIC-SWAP MODEL

Before proposing a model for an ideal atomic swap, we discuss a few naive approaches to mitigate the problem of griefing attack:

- *Allowing the party locking coins in an HTLC to cancel as per her own convenience*: To prevent the parties from speculating a deal, we modify the terms of HTLC so that *Alice* locks x_a coins in the contract forwarded to *Bob* but she is allowed to unlock her coins before t_a elapses.
  Similarly, if *Alice* delays in claiming coins after *Bob* has locked his coins, the latter can initiate a refund at his own will. Apparently, the solution looks good but it has its own problems as well. Suppose *Alice* claims y_b coins releasing the preimage s but at the same time she initiates a refund in `Chain-a`. Then *Bob* is at a loss since he cannot claim x_a coins from *Alice*. On the other hand, if *Alice* has broadcasted the claim transaction and at the same time, *Bob* has initiated a refund in `Chain-b`, and if the miner chooses to mine the later, then is at a loss. *Bob* gets the preimage for claiming coins from *Alice* but *Alice* loses her coins. The possibility of race conditions is a major problem leading to an unfair protocol.
- *Each party locks a certain amount as a penalty that is used for compensating griefing attacks*: Instead of allowing the parties to cancel the HTLC at their own will, we ask each of them to lock collateral that would be used for penalizing a party if it griefs or mounts a draining attack. The idea has been implemented in [13]. In [20] the authors have shown that asking parties to lock extra collateral as a penalty for griefing boosts the success rate. If the party chooses to stop, he or she loses the penalty. Since payoff upon stopping becomes quite low, parties are incentivized to proceed with the swap. However, the method is not effective in preventing a party to speculate. A party is penalized only when the time period elapses but not for any intermediate delay.

To prevent any sort of speculation, we propose a construction to accommodate faster resolution of atomic swap. We modify *HTLC* to *HTLC*^{c} that allows cancellation of HTLC on-chain using a different preimage. The protocol proceeds in a similar way, where at time t_0, *Alice* decides on exchanging x_a coins for y_b coins of *Bob*. *Alice* samples secret s_1 and s_3, generates $H_1 = \mathcal{H}(s_1)$ and $H_3 = \mathcal{H}(s_3)$. *Bob* samples secret s_2, generates $H_2 = \mathcal{H}(s_2)$. Both the parties share H_1, H_2 and H_3 with each other. None of the parties are asked to lock extra coins in the form of a penalty. We discuss the steps of the protocol where there is no delay involved:
(a) *Alice locks principal amount x_a in* HTLC^c *at time t_1*: The instance of *HTLC*^{c} locks x_a coins for t_a units. The condition for spending the locked coins is: If *Bob* reveals the preimage of H_1 before t_a elapses, he claims x_a coins. Else, either after t_a elapses or if preimage of H_2 gets revealed, *Alice* refunds the amount.
(b) *Bob locks principal amount y_b at time $t_2 = t_1 + \tau_a$ in an* HTLC^c: The condition for spending the locked coins is: If *Alice* reveals the preimage of H_1 before t_b elapses, she claims y_b coins. Else, either after t_b elapses or if preimage of H_3 gets revealed, *Bob* refunds the amount.
(c) *Alice claims x_a coins from* Bob *at time $t_3 = t_2 + \tau_b$*: If *Alice* reveals the preimage of H_1, then she initiates the swap.
(d) *Bob claims y_b coins from* Alice *at time $t_4 = t_3 + t_\epsilon$*: Using the preimage of H_1 revealed by *Alice*, *Bob* claims x_a coins as well.

Analysis: If *Alice* broadcasts the redeem transaction, she releases s_1 at time t_3 and claims y_b coins. *Bob* uses s_1 to withdraw x_a coins from `Chain-a` at time $t_4 = t_3 + t_\epsilon$. We now discuss the cases when the swap gets canceled:
(i) *Swap cancelled by Alice at t_3*: *Alice* shares preimage s_2 at t_3 with *Bob*. *Bob* uses s_2 to unlock y_b coins at t_4. After the refund transaction of *Bob* is confirmed at $t_3 + \tau_b$, *Bob* shares preimage s_3 with *Alice*. She uses the preimage to unlock x_a coins at time $t_3 + \tau_b$.
Fallacy: When *Alice* has canceled the swap at t_3 by sharing the preimage s_2, *Bob* withdraws y_b coins immediately. However, *Bob* is not under any compulsion to share preimage s_3 with *Alice*. He remains indifferent between canceling and griefing.
(ii) *Swap cancelled by Bob at t_2*: If *Bob* finds that locking y_b coins at t_2 to be unfavourable, then he cancels the swap by sharing preimage s_3 with *Alice* at time $t_2 = t_1 + \tau_a$. She uses the preimage to unlock x_a coins at time t_2.
Fallacy: *Bob* has no incentive to cooperate and take a decision at time $t_1 + \tau_a$. Since he has no collateral at stake, after *Alice* locks coins in t_1, *Bob* will proceed only if the situation is favourable. He will speculate in the duration $(\tau_a, t_a - (t_b - \epsilon + t_\epsilon)]$, where $T' \in [0, t_a - (t_b - \epsilon + t_\epsilon) - \tau_a]$, until he strikes a profitable deal. So the problem of delay persists.

We observe that the protocol fails if *Bob* delays or grief. We observe that until *Bob* has no collateral at stake, he may or may not choose to co-operate with *Alice*. On the other hand, if *Bob* is asked to lock extra collateral then *Alice* must lock extra collateral as well to ensure fairness. We discuss the salient features of an ideal atomic swap:
(i) If *Alice* has locked x_a coins and *Bob* has locked y_b coins, then at the end of a successful swap, *Alice* must have y_b coins and *Bob* must have x_a coins.
(ii) A party must decide immediately whether to proceed or cancel the swap and must be discouraged from speculating.
(iii) A party must be allowed to withdraw his or her asset immediately if the situation is unfavorable.
(iv) If any of the party stops responding, he or she must be penalized. The penalty is used for compensating the counterparty who suffered a loss due to locking of coins.

We propose a protocol that satisfies the above properties to a great extent and discusses the construction in detail in the next section.

5 QUICK SWAP: A PROTOCOL BASED ON IDEAL SWAP MODEL

We discuss a protocol where a party will lock the principal amount for swap provided he or she gets a guarantee of compensation upon suffering from a griefing attack. If the situation is unfavorable, he or she is allowed to withdraw the coins immediately without depending on the counterparty.

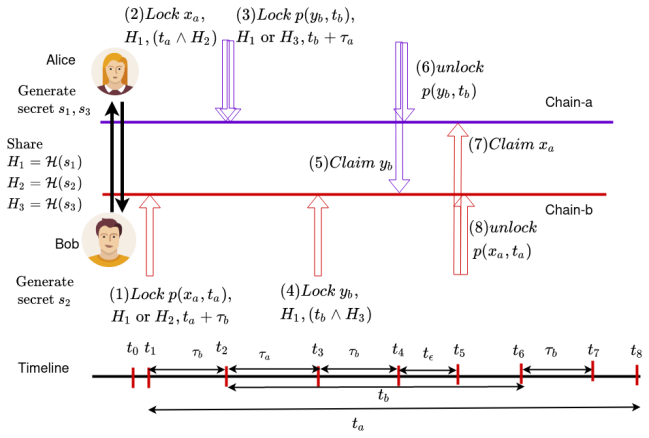


Figure 5: An instance of successful Quick Swap between Alice and Bob

High Level Overview: *Alice* has to lock x_a coins for a period of t_a units. If *Bob* griefs, *Alice*'s collateral locked will be $x_a * t_a$. She calculates the compensation that *Bob* must lock in terms of the native currency of `Chain-b`. We denote it as $p_y(x_a, t_a)$, indicating the griefing-premium is function of x_a and t_a. *Bob* provides a guarantee of compensation by locking $p_y(x_a, t_a)$ in `Chain-b` followed by *Alice* locking x_a coins in `Chain-a`. If *Bob* doesn't want to proceed and griefs, then *Alice* gets the compensation $p_y(x_a, t_a)$ after t_a elapses. Similarly, *Bob* will lock y_b coins for t_b unit of time provided he gets a guarantee of compensation from *Alice*. He calculates the compensation (also griefing-premium), $p_x(y_b, t_b)$, that *Alice* must lock in `Chain-a`. Once *Alice* locks the compensation, *Bob* locks y_b coins. If *Alice* initiates the swap, she claims y_b coins and withdraws the compensation from `Chain-a`. *Bob* gets to claim x_a coins and he withdraws the compensation from `Chain-b`. Both *Alice* and *Bob* have the freedom to refund the compensation they have locked in either chain independently, without initiating the swap. This is termed as *cancellation of swap*. If *Alice* unlocks the compensation, *Bob* initiates the refund of y_b coins and when *Bob* unlocks the compensation, *Alice* gets to refund x_a coins as well.
Assumption: In general cases, $t_a \approx 2t_b$. For ease of analysis, we consider $p_y(x_a, t_a) \geq 2p_x(y_b, t_b)$. For a fixed rate of griefing-premium, we consider $Q = p_y(x_a, t_a)$ and $\frac{Q}{2} = p_x(y_b, t_b)$.

5.1 Detailed Description of Quick Swap

An instance of successful execution of *Quick Swap* is shown in Fig. 5. We briefly describe each of the steps followed. In the *preprocessing phase*, both the parties generate the hash values H_1, H_2 and H_3 and exchange them with each other. They generate the transactions for locking griefing-premium, locking principal amount, as well as corresponding transactions to claim or refund the coins. Signatures are exchanged off-chain before the protocol execution.

In the *lock phase*, *Bob* locks griefing-premium Q at time t_1 in `Chain-b`. Though the timeout period of the contract must be equal to the timeout period of the contract in which *Alice* will lock her coins, the confirmation time of the transaction in `Chain-b`, i.e., τ_b, must be taken into account. *Alice* will lock her coins only after she gets the confirmation of *Bob*'s transaction. Hence, the timeout period is set to $t_a + \tau_b$. After the transaction gets confirmed, *Alice* locks x_a coins and griefing-premium $0.5Q$ at time $t_2 = t_1 + \tau_b$ in `Chain-a`. The timeout period of the contract where griefing-premium is locked is set to $t_b + \tau_a$, taking into account the confirmation time in `Chain-a`. After *Bob* observes that *Alice* has locked her coins, he locks y_b coins in `Chain-b` at time $t_3 = t_2 + \tau_a$, the contract timeout period being set to t_b.

In the *claim phase*, *Alice* broadcasts the transaction for redeeming y_b coins at time $t_4 = t_3 + \tau_b$, revealing preimage of H_1, and unlocks $0.5Q$ coins as well. At time $t_5 = t_4 + t_\epsilon$, *Bob* gets preimage of H_1 and redeems x_a coins and unlocks griefing-premium Q. If *Alice* wants to cancel the swap, she reveals the preimage of H_3 and unlocks $0.5Q$ coins. *Bob* uses the preimage and unlocks his principal amount. Simultaneously, he refunds Q coins and releases the preimage of H_2, enabling *Alice* to unlock x_a coins before t_a elapses. The protocol is formally described in Fig. 6 and Fig. 7, where *Alice* is denoted as **A** and *Bob* is denoted as **B**.

5.2 Game-Theoretic Analysis

We model the interaction between the two entities **A** and **B** as a sequential game $\Gamma_{quick\ swap}$. **B** initiates the lock phase by locking griefing-premium Q for duration t_a units in `Chain-b` at time t_1, followed by **A** choosing either to lock x_a coins for duration t_a units or abort at $t_2 = t_1 + \tau_b$. If **A** has not responded then **B** either cancels the swap at time $t_2 + \tau_a$ by unlocking Q or he stops. If **A** wishes to continue, she locks the principal amount and the griefing-premium $0.5Q$ for t_b units in `Chain-a` at time t_2. At time $t_3 = t_2 + \tau_a$, if **B** observes that **A** has locked the principal amount as well as griefing-premium, then **B** either locks y_b coins for t_b units in `Chain-b` or stops. If he does not lock coins, then at time $t_3 + \tau_b$, **A** cancels the swap by unlocking $0.5Q$ coins from `Chain-a` or stops. If **B** has not responded, he loses the griefing-premium at time $t_1 + t_a$. Else if he chooses to cancel, then **A** will be able to withdraw the principal amount as well. If **B** has chosen to continue, **A** decides at time $t_4 = t_3 + \tau_b$, whether to continue or cancel the swap.

5.2.1 Game Model & Preference Structure. The extensive-form game $\Gamma_{quick\ swap}$ is similar to Γ_{swap}, except that **A** and **B**'s strategy space has the option to *cancel*, apart from *continue* and *stop*. The analysis is done by applying backward induction on $\Gamma_{quick\ swap}$.

If **A** delays instead of making a move at $t_3 + \tau_b$, then the opportunity cost of coins locked as a penalty will rise. Canceling the swap at $t_3 + \tau_b$ will lead to **A**'s utility as $\frac{x_a}{e^{r_a(t_\epsilon + 2\tau_a)}} + \frac{0.5Q}{e^{r_a \tau_a}}$. If **A** chooses to delay and cancel at time say $t_3 + \tau_b + t$ for $t > 0$, then the utility drops further, i.e, $\frac{x_a}{e^{r_a(t_\epsilon + 2\tau_a + t)}} + \frac{Q}{e^{r_a(\tau_a + t)}}$. In the previous HTLC-based atomic swap, if **A** finds that the utility on continuing

- **Preprocessing Phase**:
  a. *Sampling Cancellation Hash and Payment Hash*:
    (i) **A**'s pair of secret and public key is (sk_a, pk_a) and **B**'s pair of secret and public key is (sk_b, pk_b). **A** uses sk_a and **B** uses sk_b for signing transactions. pk_a and pk_b are used for verifying each such signed transaction.
    (ii) **A** samples random values s_1 and s_3, creates payment hash $H_1 = \mathcal{H}(s_1)$ and cancellation hash $H_3 = \mathcal{H}(s_3)$. She shares H_1 and H_3 with **B**.
    (ii) **B** samples a random value s_2 and creates cancellation hash $H_2 = \mathcal{H}(s_2)$ and shares it with **A**.
  b. *Signing of transactions generated by **B***:
    (i) **B** creates a transaction `griefing_premium_lock_B` using funding address $addr_{funding,penalty,B}$ that locks Q coins into 2-of-2 multisig address $addr_{lock,penalty,B}$.
    (ii) **B** creates another transaction `principal_lock_B` using funding address $addr_{funding,principal,B}$ that locks y_b coins into 2-of-2 multisig address $addr_{lock,principal,B}$.
    (iii) For redeeming coins locked in $addr_{lock,penalty,B}$, **B** creates the transaction `griefing_premium_refund_B` where the output of `griefing_premium_lock_B` can be spend by revealing the preimage of either payment hash H_1 or cancellation hash H_2.
    (iv) For redeeming coins locked in $addr_{lock,principal,B}$, **B** creates the transaction `principal_refund_B` having relative locktime t_b and the cancellation hash H_3. This means that output of `principal_lock_B` can be claimed by **B** after elapse of t_b, if not claimed by **A**, or **B** can refund his coins instantly when preimage of H_3 is released.
    (v) He shares the transactions `griefing_premium_lock_B`, `principal_lock_B`, `principal_refund_B` and `griefing_premium_refund_B` with **A**. The latter creates transaction `griefing_premium_compensate_A` that spends the output of `griefing_premium_lock_B` after a relative locktime of $t_a + \tau_a$. She asks **B** to sign `griefing_premium_compensate_A`.
    (vi) Upon receiving the signed transaction from **B**, **A** signs `principal_lock_B`, `principal_refund_B` and `griefing_premium_refund_B`, and sends it to **B**.
    (vii) **A** creates transaction `principal_claim_B` where the output of `principal_lock_B` can be spend by revealing the preimage of H_1.
  c. *Signing of transactions generated by **A***:
    (i) **A** creates a transaction `griefing_premium_lock_A` using funding address $addr_{funding,penalty,A}$ that locks $0.5Q$ coins into 2-of-2 multisig address $addr_{lock,penalty,A}$.
    (ii) **A** creates another transaction `principal_lock_A` using funding address $addr_{funding,principal,A}$ that locks x_a coins into 2-of-2 multisig address $addr_{lock,principal,A}$.
    (iii) For redeeming coins locked in $addr_{lock,penalty,A}$, **A** creates the transaction `griefing_premium_refund_A` where the output of `griefing_premium_lock_A` can be spend by revealing the preimage of either payment hash H_1 or cancellation hash H_3.
    (iv) For redeeming coins locked in $addr_{lock,principal,A}$, **A** creates the transaction `principal_refund_A` having relative locktime t_a and the cancellation hash H_2. This means that output of `principal_lock_A` can be claimed by **A** after elapse of t_a, if not claimed by **B**, or **A** can refund his coins instantly when preimage of H_2 is released.
    (v) She shares the transactions `griefing_premium_lock_A`, `principal_lock_A`, `principal_refund_A` and `griefing_premium_refund_A` with **B**. The latter creates transaction `griefing_premium_compensate_B` that spends the output of `griefing_premium_lock_A` after a relative locktime of $t_b + \tau_b$. He asks **A** to sign `griefing_premium_compensate_B`.
    (vi) Upon receiving the signed transaction from **A**, **B** signs `principal_lock_A`, `principal_refund_A` and `griefing_premium_refund_A`, and sends it to **A**.
    (vii) **B** creates transaction `principal_claim_A` where the output of `principal_lock_A`, i.e., x_a coins can be spend by revealing the preimage of H_1.

Figure 6: Quick Swap: Preprocessing Phase

is less than the utility of the swap till the lock time of HTLC expires, she speculated till the situation turns in her favor. However, the situation is different now as **A** is allowed to abort the swap much earlier without waiting for the lock time to elapse. A rational **A** will choose not to delay anticipating that the situation may turn worse later. At time t_4, if **A** continues and **B** follows:

$$\begin{aligned} u_{A_{continue},t_4} &= (1+u_a)\frac{\mathcal{E}(x(y_b,t_4),\tau_b)}{e^{r_a\tau_b}} + \frac{0.5Q}{e^{r_a\tau_a}} - f_a - f_b \\ u_{B_{continue},t_4} &= (1+u_b)\frac{x_a}{e^{r_b(\tau_a+t_\epsilon)}} + \frac{Q}{e^{r_b(t_\epsilon+\tau_b)}} \\ &\quad -f_a - f_b \end{aligned} \tag{10}$$

- **Locking Phase**:
  (i) **B** publishes `griefing_premium_lock_B` in `Chain-b`.
  (ii) **A** checks whether `griefing_premium_B` is confirmed within τ_b units. Once that is confirmed, she publishes `principal_lock_A` and `griefing_premium_lock_A` in `Chain-a`.
  (iii) **B** checks whether the transactions broadcasted by **A** gets confirmed in another τ_a units. He then proceeds to publish `principal_lock_B` in `Chain-b`.
- **Claim Phase**:
  – *Redeem*:
    a. Before elapse of time t_b:
      (i) **A** releases the preimage s_1 for payment hash H_1 by publishing the transaction `principal_claim_B` with her signature in `Chain-b`. This allows her to claim y_b coins.
      (ii) **A** also publishes `griefing_premium_refund_A` to refund $0.5Q$ locked in `Chain-a` using s_1.
    b. Before elapse of time t_a:
      (i) **B** uses the preimage s_1 and publishes the transaction `principal_claim_A` with his signature in `Chain-a`, claiming x_a coins
      (ii) **B** unlocks Q by publishing `griefing_premium_refund_B` in `Chain-b` using s_1.
  – *Refund*:
    a. Before elapse of time t_b:
      (i) **A** releases the preimage s_2 for cancellation hash H_2 by publishing the transaction `griefing_premium_refund_A` with her signature in `Chain-a`, unlocking $0.5Q$ coins.
      (ii) **B** uses s_2 and publishes `principal_refund_B`, unlocking y_b coins from `Chain-b`.
    b. Before elapse of time t_a:
      (i) **B** uses the preimage s_3 and publishes `griefing_premium_refund_B` with her signature in `Chain-b`, unlocking Q coins.
      (ii) **A** uses s_3 and publishes `principal_refund_A`, refunding her x_a coins from `Chain-a`.

Figure 7: Quick Swap: Locking and Redeeming/Refunding

At time t_4, if **A** cancels then **B** cancels the deal as well. The payoffs are $u_{A_{cancel},t_4} = \frac{x_a}{e^{r_a(2t_\epsilon+\tau_a)}} + \frac{0.5Q}{e^{r_a\tau_a}} - 2f_a$ and $u_{B_{cancel},t_4} = \frac{\mathcal{E}(x(y_b,t_4),t_\epsilon+\tau_b)}{e^{r_b(\tau_b+t_\epsilon)}} + \frac{Q}{e^{r_b(t_\epsilon+\tau_b)}} - 2f_b$. **A** will continue at t_4 over canceling the swap, if the following condition holds: $(1+u_a)\frac{\mathcal{E}(x(y_b,t_4),\tau_b)}{e^{r_a\tau_b}} + \frac{0.5Q}{e^{r_a\tau_a}} - f_a - f_b > \frac{x_a}{e^{r_a(2t_\epsilon+\tau_a)}} + \frac{0.5Q}{e^{r_a\tau_a}} - 2f_a$

At time t_3, **B** decides to continue, the utility is expressed as follows: $u_{B_{continue},t_3} = \frac{\left[1-C(x(y,t_4)^*,x(y,t_3),\tau_b)\right]u_{B_{continue},t_4}\,dp}{e^{r_b\tau_b}} + \frac{\int_0^{x(y,t_4)^*} P(p,x(y,t_3),\tau_b)u_{B_{cancel},t_4}\,dp}{e^{r_b\tau_b}}$ and $u_{A_{continue},t_3} = \frac{\int_{x(y,t_4)^*}^{\infty} P(p,x(y,t_3),\tau_b)u_{A_{continue},t_4}\,dp}{e^{r_a\tau_b}} + \frac{\int_0^{x(y,t_4)^*} P(p,x(y,t_3),\tau_b)u_{A_{cancel},t_4}\,dp}{e^{r_a\tau_b}}$

At time t_3, **B** chooses stop, then the utility for **A** and **B** is defined as: $u_{A_{stop},t_3} = \frac{x_a}{e^{r_a(t_a+\tau_a)}} + \frac{0.5Q}{e^{r_a(t_b+\tau_a)}} - 2f_a + \frac{Q}{e^{r_b\tau_b}} - f_b$ and $u_{B_{stop},t_3} = x(y_b,t_3)$.

If at time t_3, **B** chooses cancel then he unlocks the premium Q locked in `Chain-b` by releasing preimage of H_2. **A** observes that swap is canceled, so she unlocks x_a coins and griefing-premium $0.5Q$ from `Chain-a`. The utility is defined as $u_{B_{cancel},t_3} = x(y_b,t_3) + \frac{Q}{e^{r_b\tau_b}} - f_b$ and $u_{A_{cancel},t_3} = \frac{x_a}{e^{r_a(\tau_a+t_\epsilon)}} + \frac{0.5Q}{e^{r_a(t_\epsilon+\tau_a)}} - 2f_a$. We observe that it is better to cancel the swap than wait for the contract to expire as **B** will lose his griefing-premium in the process. Thus *cancel* strictly dominates *stop* in our protocol. **B** will continue at t_3 over canceling the swap, if the following condition holds: $u_{B_{continue},t_3} > u_{B_{cancel},t_3}$

At time t_2, **A** decides to continue, then utility is: $u_{A_{continue},t_2} = \left(\frac{\int_{x_1^3}^{x_2^3} P(p,x(y,t_2),\tau_a)u_{A_{continue},t_3}\,dp}{e^{r_a\tau_a}} + \frac{\int_{x_2^3}^{\infty} P(p,x(y_b,t_2),\tau_a)u_{A_{stop},t_3}\,dp}{e^{r_a\tau_a}}\right)$ and $u_{B_{continue},t_2} = \left(\frac{\int_{x_1^3}^{x_2^3} P(p,x(y,t_2),\tau_a)u_{B_{continue},t_3}\,dp}{e^{r_b\tau_a}} + \frac{\int_{x_2^3}^{\infty} P(p,x(y_b,t_2),\tau_a)u_{B_{stop},t_3}\,dp}{e^{r_b\tau_a}}\right)$ At time t_2, **A** decides to abort then **B** initiates cancellation by releasing preimage of H_2. Note that **A** will not take any action if she intends to cancel as she has not locked any coins. Thus both cancel and stop means the same payoff for **A**. The payoff is defined as $u_{A_{stop},t_2} = u_{A_{cancel},t_2} = x_a + 0.5Q$ and $u_{B_{cancel},t_2} = x(y_b,t_2) + \frac{Q}{e^{r_b(\tau_a+\tau_b)}} - f_b$. **A** will continue at t_2 over stopping the swap, if the following condition holds: $u_{A_{continue},t_2} > u_{A_{cancel},t_2}$ At time t_1, **B** decides to continue, the payoffs are defined as $u_{B_{continue},t_1} = \left(\frac{\int_{x_1^2}^{x_2^2} P(p,x(y,t_1),\tau_b)u_{B_{continue},t_2}\,dp}{e^{r_b\tau_b}} + \frac{\int_0^{x_1^2} P(x,x(y,t_1),\tau_b)u_{B_{stop},t_2}\,dp}{e^{r_b\tau_b}}\right)$ and $u_{A_{continue},t_1} = \left(\frac{\int_{x_1^2}^{x_2^2} P(p,x(y,t_1),\tau_b)u_{A_{continue},t_2}\,dp}{e^{r_a\tau_b}} + \frac{\int_0^{x_1^2} P(x,x(y,t_1),\tau_b)u_{A_{stop},t_2}\,dp}{e^{r_a\tau_b}}\right)$.

At time t_1, **B** decides to stop, the utility is expressed as $u_{A_{stop},t_1} = x_a + 0.5Q$ and $u_{B_{stop},t_1} = x(y_b, t_1) + Q$. **B** will continue at t_1 over stopping the swap, if the following condition holds: $u_{B_{continue},t_1} > u_{B_{cancel},t_1}$.

5.2.2 Comparing efficiency of Quick Swap with HTLC-based atomic swap. In *Quick Swap*, success rate or *SR* is function of x_a (or **A**'s tokens) since there is no delay involved. It is expressed as:

$$SR(x_a) = \int_{x_1^2[x_a]}^{x_2^2[x_a]} P(q, x(y, t_1), \tau_b) \int_{x_1^3[x_a]}^{x_2^3[x_a]} A(x_a)\, dp\, dq \quad (11)$$

where $A(x_a) = P(p, q, \tau_a)(1 - C(x(y, t_3)[x_a], p, \tau_b))$

We plot the success rate of *Quick Swap* in Fig.8 (a-c), keeping the parameters as used in Fig. 3 (a-b), except for varying $x_a \in [1, 4]$. It is observed that the highest success rate of *Quick Swap* is lower (varying between $0.3 - 0.65$) compared to HTLC-based atomic swap with no waiting (i.e., $T = T' = 0$). However, **A** is able to estimate the success rate now as it is dependent solely on x_a. There is no uncertainty involved, unlike in HTLC-based atomic swap where a higher delay leads to violation of participation constraint. Additionally, the range of x_a for which *Quick Swap* has a non-zero success rate is larger. The provision of cancellation, even before the elapse of the contract's locktime, makes the protocol robust and more resilient to fluctuation in asset price.

5.3 Discussions

(a) Our protocol requires an extra round compared to *HTLC*-based atomic swap. The number of transactions created for *Quick Swap* is double the number of transactions needed for the latter.
(b) If *Bob* doesn't take any action by t_3, then by time $t_3 + \tau_b$, *Alice* refunds the griefing-premium $0.5Q$ by revealing preimage of H_3 in her own interest. *Bob* observes that *Alice* has canceled the swap by withdrawing the griefing-premium, so he releases preimage s_2 for H_2 and cancels the swap as well. *Bob* unlocks griefing-premium Q and *Alice* refunds x_a coins using preimage s_2.

6 CONSTRUCTION OF A FAIR MULTIPARTY CYCLIC ATOMIC SWAP

Suppose *Alice* wants to exchange coins with *Bob* where the former has some Bitcoins and later has Ethers, but *Bob* wants to accept coins in Ripple. In such a scenario, they take the help of some intermediaries for assisting in the exchange of coins. If we consider just a three-party situation, then there may exist a participant *Carol* who is willing to exchange Ripple for Bitcoins. So *Alice* send x_a BTC to *Carol* and *Carol* sends z_c XRP to *Bob*, and finally *Bob* sends y_b ETH to *Alice*. In a real situation, *Carol* will charge a fee from *Alice* for facilitating the swap but we ignore the fee in this paper.

Problem of Griefing: In an HTLC-based setting, *Alice* samples a secret s, shares $H = \mathcal{H}(s)$ with *Carol* and *Bob*. *Alice* forwards HTLC to *Carol* locking x_a BTC in `Chain-a` for T_1 unit of time contingent to providing preimage of H. The confirmation time of a transaction in `Chain-a` in τ_a. *Carol* forwards an HTLC to *Bob* using the same condition for a time period of T_2 units where $T_2 < T_1$, locking z_c XRP in `Chain-b`. The confirmation time of a transaction in `Chain-b` in τ_b, Finally, *Bob* forwards the HTLC, locking y_b BTC in `Chain-c` for time period of T_3 units where $T_3 < T_2$. The confirmation time of a transaction in `Chain-c` in τ_c. The problem of griefing persists as *Carol* may not choose to lock coins based on the fluctuation rate of Bitcoin and Ripple, and even if *Carol* locks her coins, *Bob* may abort. If all the parties have locked coins, *Alice* may abort, and makes *Bob* and *Carol* suffer. We discuss a fix to this problem by extending *Quick Swap* from a two-party setting to a three-party setting.

High Level Overview of the protocol. We depict the steps followed upon extending *Quick Swap* to three party setting in Fig. 9. *Bob* samples a hashes H_1 and H_3 using secrets s_1 and s_3 respectively, locks griefing-premium $p_y(x_a, T_1)$ in `Chain-c` for a time-period of $T_1 + \tau_c$ and hashlock $H_3 \vee H_1$, at time t_1. *Alice* locks principal amount x_a for time period T_1 using hashlock H_1 at time $t_2 = t_1 + \tau_a$, with the provision of refunding earlier if preimage of H_3 is revealed. She samples a hash H_2 using secret s_2, and locks griefing-premium $p_x(z_c, T_2)$ for a time period $T_2 + \tau_a$, hashlock $H_1 \vee H_2$ in `Chain-a` at time t_2. *Carol* locks principal amount z_c for time period T_2, hashlock H_1, at time $t_3 = t_2 + \tau_b$ in `Chain-b`. She can refund before T_2 elapses if preimge of H_2 is revealed. *Carol* samples hash H_4 using secret s_4, and locks griefing-premium $p_z(y_b, T_3)$ for timperiod $T_3 + \tau_b$, hashlock $H_1 \vee H_4$ in `Chain-b`. Finally, *Bob* locks z_c coins in `Chain-c` for timeperiod T_3, hashlock H_1, at time $t_2 = t_1 + \tau_a$. He has a provision to refund earlier if the preimage of H_4 is revealed.

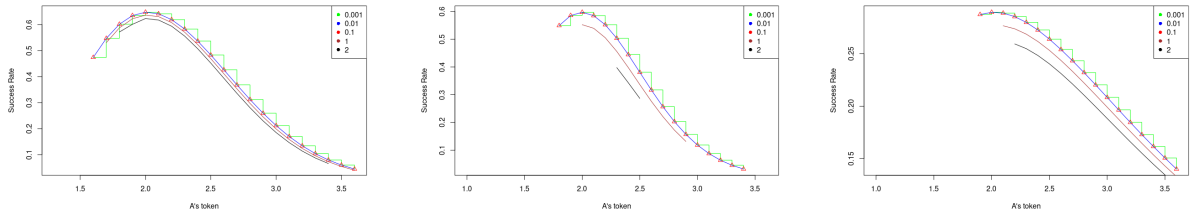
6.1 Generic n-party fair cyclic atomic swap

6.1.1 System Model & Assumption. Party P_0 wants to exchange α_0 coins for α_n coins of party P_n, taking help of $n-1$ intermediaries $P_1, P_2, \ldots, P_{n-1}$. A party P_i has account in `Chain-i` and `Chain-((`$i-1 \mod n+1$`)` . Blockchain `Chain-i` has transaction confirmation time τ_i where $i \in [0, n]$.

6.1.2 Detailed Construction. We describe the steps:
(a) P_n initiates, samples secret for payment hash $\bar{H}$ and cancellation hash H_n. He locks griefing-premium $p_n(\alpha_0, T_0)$ for locktime $T_0 + \tau_n$ in `Chain-n`, using hashlock $\bar{H} \vee H_n$, at time t_1.
(b) Rest of the parties $P_i, i \in [0, n-1]$ does the following: P_i generates a cancellation hash H_i and locks α_i coins for locktime T_i, using hashlock $\bar{H}$, at time $t_{i+2} = t_{i+1} + \tau_{(i-1) \mod n+1}$ in `Chain-i`. The coins can be refunded before T_i if preimage of $H_{(i-1) \mod (n+1)}$ is revealed. He also locks griefing-premium $p_i(\alpha_{i+1}, T_{i+1})$ at t_{i+2}, for locktime $T_{i+1} + \tau_i$, using hashlock $\bar{H} \vee H_i$.
(c) Finally, P_n locks α_n coins for locktime T_n, using hashlock $\bar{H}$, at time $t_{n+2} = t_{n+1} + \tau_{n-1}$ in `Chain-n`. He has an option to refund the coins if P_{n-1} cancels the swap by revealing the preimage of H_{n-1}. The locktimes assigned follow a strictly decreasing order: $T_0 > T_1 > \ldots > T_n$.

7 RELATED WORKS

The HTLC-based atomic swap was first proposed in [1]. However, the design lacks fairness and is susceptible to griefing attacks. Later Hao et al. [8] suggested the use of premium to counter griefing attacks. However, the protocol assumed that in a two-party setting where *Alice* wants to exchange currency with *Bob*, only *Alice* can be at fault. So she must lock premium and *Bob* is not required to do so. In an American-style option-based swap, *Bob* gets the premium even though *Alice* initiates the swap on time. In currency exchange-based atomic swap, *Bob* gets the premium if *Alice* doesn't respond



(a) $u_a = u_b = 0.3, r_a = r_b = 0.005, \sigma = 0.1$ **(b)** $u_a = u_b = 0.3, r_a = r_b = 0.01, \sigma = 0.1$ **(c)** $u_a = u_b = 0.3, r_a = r_b = 0.005, \sigma = 0.2$

Figure 8: Quick Swap's Success Rate as a function of x_a with different parameter values

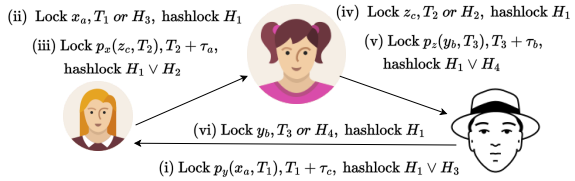


Figure 9: Quick Swap extended to three-party cyclic atomic swap

	TN [1] Swap	HLY [8] Swap	XH [21] Swap	NKW [13] Swap	*Quick Swap*
$P1$	✗	Partial	✓	✓	✓
$P2$	✗	✗	✗	Partial	✓
$P3$	✗	✗	✗	Partial	✓
$P4$	NA	✗	✗	✗	✓
$P5$	✓	✗	✗	✓	✓
$P6$	NA	✗	✗	✗	✓

Table 2: Comparative Analysis of *Quick Swap* with existing Atomic Swap protocols in terms of $P1$: Countering griefing attack, $P2$: Cancellation Allowed, P_3 Counters speculation, $P4$ Fairness of premium locked, P_5 Supported by Bitcoin script and $P6$: Extension to multi-party cyclic swap

within the time period of the contract. The protocol is not fair as *Bob* can grief as well. The construction cannot be realized in Bitcoin scripts as it requires the inclusion of an additional opcode.

Similar work has been done that talks about locking premium by both the parties involved in exchanging currency [21]. However, the protocol is not compatible with Bitcoin scripts and suffers from the problem of mismatched premiums, and lacking fairness. Further, the authors have bootstrapped the premium, whereby small valued premiums get locked first, and with each iteration, the premium amount increases. This leads to multiple round communication, creation of multiple contracts for each iteration, and longer lock time than [1] and griefing on the locked-up premium is possible [13]. Nadahalli et al. [13] have proposed a protocol that is *grief-free* and compatible with Bitcoin scripts. The protocol is efficient regarding the number of transactions and the worst-case timelock for which funds remain locked. However, the problem of mismatched premium exists. The model lacks flexibility due to the coupling of premium with the principal amount, and thus cannot be extended to multi-party atomic swap setting involving more than two blockchains [9]. Our proposed protocol overcomes several such shortcomings. However, there is a constant factor increase in overhead of transaction and communication round compared to [8] and [13]. We summarize the discussion by performing a comparative analysis of *Quick Swap* with other state-of-the-art protocols in Table 2.

8 CONCLUSION

In this paper, we perform a game-theoretic analysis of HTLC-based atomic swap. We observe that the protocol is susceptible to delay by either of the parties, resulting in the locking of coins for a large duration. We discuss a model for an ideal atomic swap and propose *Quick Swap* that satisfies the properties of an ideal atomic swap. Our proposed protocol is robust and allows faster settlement of the transaction. We discuss the step for extending *Quick Swap* to a multiparty setting involving more than two blockchains. As a part of our future work, we would like to analyze atomic swaps in presence of malicious participants. The problem becomes more complex as it is impossible to enforce malicious agents to complete the transaction.

REFERENCES

[1] 2013. *TierNolan.* Technical Report. https://github.com/TierNolan.
[2] December 2018. An Argument For Single-Asset Lightning Network. https://lists.linuxfoundation.org/pipermail/lightning-dev/2018-December/001752.html.
[3] Mihir Bellare and Phillip Rogaway. 1993. Random oracles are practical: A paradigm for designing efficient protocols. In *Proceedings of the 1st ACM Conference on Computer and Communications Security.* ACM, New York, 62–73.
[4] Michael Borkowski, Marten Sigwart, Philipp Frauenthaler, Taneli Hukkinen, and Stefan Schulte. 2019. DeXTT: Deterministic cross-blockchain token transfers. *IEEE Access* 7 (2019), 111030–111042.
[5] Bingrong Dai, Shengming Jiang, Menglu Zhu, Ming Lu, Dunwei Li, and Chao Li. 2020. Research and implementation of cross-chain transaction model based on improved hash-locking. In *International Conference on Blockchain and Trustworthy Systems.* Springer, 218–230.
[6] Stephen Dipple, Abhishek Choudhary, James Flamino, Boleslaw K Szymanski, and György Korniss. 2020. Using correlated stochastic differential equations to forecast cryptocurrency rates and social media activities. *Applied Network Science* 5, 1 (2020), 1–30.
[7] Thomas Eizinger, Philipp Hoenisch, and Lucas Soriano del Pino. 2021. Open problems in cross-chain protocols. arXiv:2101.12412 [cs.CR]
[8] Runchao Han, Haoyu Lin, and Jiangshan Yu. 2019. On the optionality and fairness of Atomic Swaps. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies.* 62–75.
[9] Maurice Herlihy. 2018. Atomic Cross-Chain Swaps. In *Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC 2018, Egham, United Kingdom, July 23-27, 2018*, Calvin Newport and Idit Keidar (Eds.). ACM, 245–254. https://doi.org/10.1145/3212734
[10] Léonard Lys, Arthur Micoulet, and Maria Potop-Butucaru. 2021. *R-SWAP: Relay based atomic cross-chain swap protocol.* Ph. D. Dissertation. Sorbonne Université.

[11] A. G. Malliaris. 1990. *Wiener Process.* Palgrave Macmillan UK, London, 316–318. https://doi.org/10.1007/978-1-349-20865-4_43

[12] Pedro Moreno-Sanchez, Arthur Blue, Duc Viet Le, Sarang Noether, Brandon Goodell, and Aniket Kate. 2020. DLSAG: Non-interactive Refund Transactions for Interoperable Payment Channels in Monero. In *Financial Cryptography and Data Security - 24th International Conference, FC 2020 (Lecture Notes in Computer Science, Vol. 12059)*, Joseph Bonneau and Nadia Heninger (Eds.). Springer, 325–345. https://doi.org/10.1007/978-3-030-51280-4_18

[13] Tejaswi Nadahalli, Majid Khabbazian, and Roger Wattenhofer. 2022. Grief-free Atomic Swaps. In *2022 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. 1–9. https://doi.org/10.1109/ICBC54727.2022.9805490

[14] Krishnasuri Narayanam, Venkatraman Ramakrishna, Dhinakaran Vinayagamurthy, and Sandeep Nishad. 2022. Generalized HTLC for Cross-Chain Swapping of Multiple Assets with Co-Ownerships. *arXiv preprint arXiv:2202.12855* (2022).

[15] Joseph Poon and Thaddeus Dryja. 2016. The bitcoin lightning network: Scalable off-chain instant payments.

[16] Daniel Robinson. 2019. HTLCs considered harmful. In *Stanford Blockchain Conference*.

[17] Erkan Tairi, Pedro Moreno-Sanchez, and Matteo Maffei. 2021. A^2L: Anonymous Atomic Locks for Scalability and Interoperability in Payment Channel Hubs. In *2021 IEEE Symposium on Security and Privacy (SP)*. IEEE, 1834–1851.

[18] Stefan Thomas and Evan Schwartz. 2015. A protocol for interledger payments. *URL https://interledger. org/interledger. pdf* (2015).

[19] S. Thyagarajan, G. Malavolta, and P. Moreno-Sanchez. 2022. Universal Atomic Swaps: Secure Exchange of Coins Across All Blockchains. In *2022 2022 IEEE Symposium on Security and Privacy (SP) (SP)*. IEEE Computer Society, Los Alamitos, CA, USA, 1083–1100. https://doi.org/10.1109/SP46214.2022.00063

[20] Jiahua Xu, Damien Ackerer, and Alevtina Dubovitskaya. 2021. A game-theoretic analysis of cross-chain atomic swaps with HTLCs. In *2021 IEEE 41st International Conference on Distributed Computing Systems (ICDCS)*. IEEE, 584–594.

[21] Yingjie Xue and Maurice Herlihy. 2021. Hedging against sore loser attacks in cross-chain transactions. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*. 155–164.

[22] Victor Zakhary, Divyakant Agrawal, and Amr El Abbadi. 2020. Atomic commitment across blockchains. *Proceedings of the VLDB Endowment* 13, 9 (2020), 1319–1331.

[23] Alexei Zamyatin, Dominik Harz, Joshua Lind, Panayiotis Panayiotou, Arthur Gervais, and William J. Knottenbelt. 2019. XCLAIM: Trustless, Interoperable, Cryptocurrency-Backed Assets. In *2019 IEEE Symposium on Security and Privacy, SP 2019, San Francisco, CA, USA, May 19-23, 2019*. IEEE, 193–210. https://doi.org/10.1109/SP.2019.00085