Atomic swaps

Lorenzo Tucci

March 12, 2024

Parties U_0 and U_1 hold assets a on blockchain $\mathbb{A}$ and assets b on chain $\mathbb{B}$ respectively.
We define with $\mathsf{swp}(a)$ and $\mathsf{swp}(b)$ the amount of the assets the parties agreed to swap before starting the protocol.

We define the following oracles to interact with the blockchains.

- $\mathsf{PubTx}(\sigma_{tx}, tx, \mathbb{A})$ publish the transaction tx with signature σ_{tx} on chain $\mathbb{A}$
- $\mathsf{InitTx}(pk_{tx}, pk_{rx}, amnt, \mathbb{A})$ create an unsigned transaction paying $amnt$ from pk_{tx} to pk_{rx} on chain $\mathbb{A}$
- $\mathsf{WatchTx}(tx, \mathbb{A})$ wait for the transaction tx to be confirmed on chain $\mathbb{A}$
- $\mathsf{GetBal}(pk, \mathbb{A})$ get the balance of assets held by pk
- $\mathsf{GetSig}(pk, \mathbb{A})$ get the signature σ_{tx} of the latest transaction in pk's record on chain $\mathbb{A}$

U_1 starts counting the timeout from the moment they send the VTD commitment to U_0, and respectively U_0 starts counting down from the moment they receive it.

If the timeout expires before the protocol is completed:

- U_1 will transfer the coins from $pk(1)$ to another wallet on $\mathbb{B}$. From this moment on, if U_0 tries to $\mathsf{PubTx}(\sigma_{\mathsf{swp(10)}}, tx_{\mathsf{swp}}, \mathbb{A})$, the transaction will get rejected.

- U_0 will wait until $\Pi_{\mathsf{VTD}}.\mathsf{ForceOp}(C)$ is completed to get the other secret key share $sk_1(01)$ of $pk(10)$ to retrieve $sk(10)$ and transfer back assets a to $pk(0)$.

$U_0(pk(0)\ , sk(0))$		$U_1(pk(1)\ , sk(1))$
	$\xleftrightarrow{\Gamma_{\mathsf{KeyGen}}(\mathbb{G}, G, q)}$ $\longleftarrow (sk_0(01), pk(01))$ $(sk_1(01), pk(01)) \longrightarrow$	
		$(C, \pi) \leftarrow \Pi_{\mathsf{VTD}}.\mathsf{Commit}(sk_1, T)$
	$\xleftarrow{(C, \pi)}$	
starts $\mathsf{Timeout}(T - \Delta)$		starts $\mathsf{Timeout}(T - \Delta)$
if $\Pi_{\mathsf{VTD}}.\mathsf{Verify}(pk, C, \pi) \neq 1$ abort $tx_{\mathsf{frz}} \leftarrow \mathsf{InitTx}(pk(0), pk(01), \mathsf{swp(a)}, \mathbb{A})$ $\sigma_{\mathsf{frz}} \leftarrow \Pi_{\mathsf{DS}}.\mathsf{Sign}(sk(0), tx_{\mathsf{frz}})$ $\mathsf{PubTx}(\sigma_{\mathsf{frz}}, tx_{\mathsf{frz}}, \mathbb{A})$ starts $\Pi_{\mathsf{VTD}}.\mathsf{ForceOp}(C)$		

		do $\mathsf{bal} \leftarrow \mathsf{GetBal}(pk(01), \mathbb{A})$
		while $\mathsf{bal} \neq \mathsf{swp(a)}$

	$\xleftarrow{pk(1)}$	
$(pk(10), sk(10)) \leftarrow \Pi_{\mathsf{DS}}.\mathsf{KeyGen}(1^{\lambda})$ $tx_{\mathsf{swp}} \leftarrow \mathsf{InitTx}(pk(1), pk(10), \mathsf{swp(b)}, \mathbb{A})$		
	Γ_{Swap} $U_0 \longrightarrow (sk_0(01), tx_{\mathsf{swp}})$ $\xleftrightarrow{(sk_1(01), sk(1)) \longleftarrow U_1}$ $lk := \sigma_{swp}(10) \oplus sk_0(01) \longrightarrow$ $\longleftarrow \sigma_{swp}(10)$	
$\mathsf{PubTx}(\sigma_{\mathsf{swp(10)}}, tx_{\mathsf{frz}}, \mathbb{A})$		

		do $\sigma_{swp}(10) \leftarrow \mathsf{GetSig}(pk(1), \mathbb{B})$
		    $sk(01) \leftarrow (lk \oplus \sigma_{swp}(10)) \oplus sk_1$
		    $\sigma_m \leftarrow \Pi_{\mathsf{DS}}.\mathsf{Sign}(sk(01), 1)$
		while $\Pi_{\mathsf{DS}}.\mathsf{Verify}(m, pk, \sigma_m) \neq 1$

Figure 1: Protocol execution for a successful swap

$U_0(pk(0)\ , sk(0))$	$U_1(pk(1)\ , sk(1))$
$sk(01) := sk_0(01) \oplus sk_1(01)$ $\sigma_{swp}(10) \leftarrow \Pi_{\mathsf{DS}}.\mathsf{Sign}(sk(1), tx_{\mathsf{swp}})$ $lk := \sigma_{swp}(10) \oplus sk_0(01)$	

Figure 2: Protocol definition of 2PC Γ_{Swap}