# CONTENTS

I  Introduction........................................... 1

II  Cross-chain Transactions.......................... 2

III  Sequence Transformation.......................... 3

IV  Transaction Protocol............................... 4

V  Implementation........................................ 7

VI  Related Works........................................ 8

VII Conclusion............................................ 8

References................................................. 9

VIII Appendix............................................... 11
   VIII-A  Proof........................................... 11

IX  Use-cases.............................................. 12
   IX-A  Use-case 1........................................ 12
   IX-B  Use-case 2........................................ 13
   IX-C  Use-case 3........................................ 14
   IX-D  Use-case 4........................................ 15
   IX-E  Use-case 5........................................ 16
   IX-F  Use-case 6........................................ 17
   IX-G  Use-case 7........................................ 18
   IX-H  Extensions........................................ 19

IX-H  Extensions........................................... 19
VIII. APPENDIX

A. Proof

Lemma 2 (Uniformity): If the representative source vertices are conforming, 3PP is uniform.

PROOF: Uniformity is the conjunction of three conditions. We prove them in turn.

(a) All the contracts are eventually created. We prove that every party eventually creates its outgoing contracts. This fact is proved by induction on the maximum of the distances of the party from the leaders. In the base case, the leaders themselves create their outgoing contracts with no condition (lines $P_{2}-P_{3}$). In the inductive case, every follower eventually receives her last incoming contract from her predecessor that is on the path from the leader with the longest distance. In response, she creates her outgoing contracts (lines $P_{5}-P_{11}$). We note that as the parties are assumed to follow the protocol, the contracts that they create are valid. Thus, parties do not stop because of invalid contracts.

(b) Leaders do not wait indefinitely. By [a], leaders eventually observe their incoming contracts and in response, stop waiting for incoming contracts (line $P_{3}$). Further, from [a], the representative sinks receive their incoming contracts and send messages to representative sources (lines $P_{12}-P_{13}$). Therefore, the representative sources receive messages from all the representative sinks and stop waiting (lines $P_{6}-P_{7}$).

(c) Every secret eventually reaches a representative source. By [b], the feedback vertex set eventually start the second phase and release their secrets on their incoming edges (lines $P_{15}-P_{16}$). Parties propagate secrets set backwards (lines $P_{17}-P_{18}$). Consider a leader $v_l$ in the feedback vertex set. Consider its super-vertex $w_l$ in the condensation graph. The vertices in $w_l$ are an SCC; thus, they are reachable from each other. Therefore, the secret of $v_l$ propagates to every party in $w_l$. Further, every super-vertex is reachable from at least one super-source. Therefore, the secret of $v_l$ eventually propagates back to at least one super-source $w_s$. The vertices in $w_s$ are an SCC and the clearing service has chosen one of them as the representative source. Therefore, the secret of the leader $v_l$ eventually propagates to that representative source. In addition to the feedback vertex set, the representative sources are leaders themselves. Their secrets are trivially at the representative sources.

(d) All vertices eventually receive all secrets. The representative sources relay their secrets to the pseudo-sinks (lines $P_{20}-P_{21}$). Thus, by [c], every pseudo-sink eventually receives all secrets. In the condensation graph, every super-vertex is reachable to at least one super-sink. Further, vertices of a super-vertex are an SCC and reachable from each other. Parties other than the representative sources propagate secrets backwards (lines $P_{23}-P_{24}$). Therefore, the secrets are eventually propagated from the pseudo-sinks back to all vertices.

(e) The transaction is eventually committed. First, from [a], all contracts are eventually created. Second, from [d], all vertices eventually receive all secrets. Parties apply the secrets that they receive to their incoming contracts (lines $P_{25}-P_{24}$). Therefore, every contract is eventually triggered i.e. the transaction is committed.

(2) If any set of parties deviate from 3PP, no conforming party finishes with an UnderWater state. A party ends up in an UnderWater state when some of her outgoing contracts are triggered, but all her incoming contracts are not triggered. It is trivial that a source party cannot end up in an UnderWater state as it does not have any incoming contracts. If the party is a conforming leader, she releases her secret only after she receives her incoming contracts (lines $P_{4}-P_{5}$). Therefore, her outgoing contracts can be triggered only after her incoming contracts are already created because all of her outgoing contracts are locked by her own secret. Thus, if any of her outgoing contracts are triggered, she can learn the secrets and subsequently trigger her incoming contracts. Conforming followers create outgoing contracts only after they receive their incoming contracts (lines $P_{6}-P_{11}$). Therefore, if an outgoing contract is triggered, they can learn the secrets and trigger their incoming contracts. Thus, if an outgoing contract of a conforming party is triggered, she can trigger all her incoming contracts. Thus, she cannot end up in an UnderWater state.

(3) $P$ is end-to-end. Assuming that the transaction is source-paid, we show that it is sink-paying. Since the transaction is source-paid, a contract is triggered in the transaction. Every contract is hashlocked with the secret of all leaders. The representative sources are in the set of leaders. Therefore, the secrets of the representative sources should be known. Therefore, the third phase must have started. Similar to 1.(c), we can show that by the end of the second phase, all the secrets reach the representative sources. At the beginning of the third phase, the representative sources send secrets to pseudo-sinks. These secrets include representative sources’ own secrets and the secrets of the feedback vertex set that they have received in the second phase. Every complying vertex applies the secrets that it receives to its incoming contracts. Therefore, the complying pseudo-sinks apply the secrets to the incoming contracts and trigger them. Thus, the transaction is sink-paying. □
IX. Use-cases

A. Use-case 1

Fig. 8. First Use-Case

<table>
<thead>
<tr>
<th>Leaders</th>
<th>Synthesis Time (ms)</th>
<th>Gas Consumption</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>294</td>
<td>Deploy</td>
<td>Initite</td>
</tr>
<tr>
<td></td>
<td>1220926</td>
<td>77772</td>
<td>47366</td>
</tr>
<tr>
<td></td>
<td>1221334</td>
<td>77794</td>
<td>47322</td>
</tr>
</tbody>
</table>

TABLE II

Execution of First Use-case
B. Use-case 2

(a) Transaction with 3 nodes

(b) Equivalent transformation of the transaction (a)

Fig. 9. Second Use-case

<table>
<thead>
<tr>
<th>Leaders</th>
<th>Synthesis Time (ms)</th>
<th>Gas Consumption</th>
<th>Deploy</th>
<th>Initiate</th>
<th>Redeem</th>
<th>Claim</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>B</td>
<td>375</td>
<td></td>
<td>1263228</td>
<td>78350</td>
<td>47300</td>
<td>55644</td>
<td>1444522</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1367090</td>
<td>78328</td>
<td>47366</td>
<td>55622</td>
<td>1548406</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1263160</td>
<td>78350</td>
<td>47300</td>
<td>55644</td>
<td>1444454</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1263368</td>
<td>78350</td>
<td>47300</td>
<td>55644</td>
<td>1444662</td>
</tr>
</tbody>
</table>

TABLE III
EXECUTION OF SECOND USE-CASE
### C. Use-case 3

(a) Transaction with 4 nodes

(b) Equivalent transformation of the transaction (a)

(c) Equivalent transformation of the transaction (b)

Fig. 10. Equivalent transformation of the transaction (b)

Fig. 11. Third Use-case

<table>
<thead>
<tr>
<th>Leaders</th>
<th>Synthesis Time (ms)</th>
<th>Gas Consumption</th>
<th>Gas Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Deploy</td>
<td>Initiate</td>
<td>Redeem</td>
</tr>
<tr>
<td>A</td>
<td>330</td>
<td>78902</td>
<td>47342</td>
</tr>
<tr>
<td></td>
<td>1721271</td>
<td>78902</td>
<td>47342</td>
</tr>
<tr>
<td></td>
<td>1617422</td>
<td>78946</td>
<td>47254</td>
</tr>
<tr>
<td></td>
<td>1513707</td>
<td>78906</td>
<td>47258</td>
</tr>
</tbody>
</table>

**TABLE IV**

**EXECUTION OF THIRD USE-CASE**
D. Use-case 4

(a) Transaction with 5 nodes

(b) Equivalent transformation of the transaction (a)

Fig. 12. Fourth Use-case

<table>
<thead>
<tr>
<th>Leaders</th>
<th>Synthesis Time (ms)</th>
<th>Gas Consumption</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Deploy</td>
<td>Initiate</td>
</tr>
<tr>
<td>A</td>
<td>310</td>
<td>1347499</td>
</tr>
<tr>
<td></td>
<td>79462</td>
<td>79462</td>
</tr>
<tr>
<td></td>
<td>47306</td>
<td>47306</td>
</tr>
<tr>
<td></td>
<td>58740</td>
<td>58740</td>
</tr>
<tr>
<td></td>
<td>58740</td>
<td>58740</td>
</tr>
<tr>
<td></td>
<td>1533007</td>
<td>1533007</td>
</tr>
<tr>
<td></td>
<td>1472464</td>
<td>1472464</td>
</tr>
<tr>
<td></td>
<td>1533139</td>
<td>1533139</td>
</tr>
</tbody>
</table>

TABLE V
EXECUTION OF FORTH USE-CASE
E. Use-case 5

(a) Transaction with 6 nodes

(b) Equivalent transformation of the transaction (a)

Fig. 13. Fifth Use-case

<table>
<thead>
<tr>
<th>Leaders</th>
<th>Synthesis Time (ms)</th>
<th>Gas Consumption</th>
<th>Deploy</th>
<th>Initiate</th>
<th>Redeem</th>
<th>Claim</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>A, D</td>
<td>318</td>
<td></td>
<td></td>
<td>1494188</td>
<td>80018</td>
<td>79636</td>
<td>79664</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1494252</td>
<td>80018</td>
<td>79636</td>
<td>59664</td>
<td>59664</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1494056</td>
<td>80018</td>
<td>79636</td>
<td>59664</td>
<td>59664</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1494252</td>
<td>80018</td>
<td>79636</td>
<td>59664</td>
<td>59664</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1494252</td>
<td>80018</td>
<td>79636</td>
<td>59664</td>
<td>59664</td>
</tr>
</tbody>
</table>

TABLE VI
EXECUTION OF FIFTH USE-CASE
F. Use-case 6

(a) Transaction with 7 nodes

(b) Equivalent transformation of the transaction (a)

Fig. 14. Sixth Use-case

<table>
<thead>
<tr>
<th>Leaders</th>
<th>Synthesis Time (ms)</th>
<th>Gas Consumption</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>A, F</td>
<td>343</td>
<td>Deploy</td>
<td>15342206</td>
<td>80552</td>
<td>797554</td>
<td>61190</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Initiate</td>
<td>1534214</td>
<td>80574</td>
<td>796878</td>
<td>61212</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Redeem</td>
<td>1534218</td>
<td>80574</td>
<td>796948</td>
<td>61212</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Claim</td>
<td>1534228</td>
<td>80596</td>
<td>796448</td>
<td>61212</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Total</td>
<td>1534286</td>
<td>80574</td>
<td>79700</td>
<td>61212</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1534882</td>
<td>80574</td>
<td>79694</td>
<td>61212</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1534882</td>
<td>80574</td>
<td>79694</td>
<td>61212</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1535954</td>
<td>80574</td>
<td>79700</td>
<td>61212</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>1535086</td>
<td>80574</td>
<td>79694</td>
<td>61212</td>
</tr>
</tbody>
</table>

TABLE VII

Execution of Sixth Use-case
Fig. 15. Seventh Use-Case

<table>
<thead>
<tr>
<th>Leaders</th>
<th>Synthesis Time (ms)</th>
<th>Deploy</th>
<th>Initiate</th>
<th>Redeem</th>
<th>Claim</th>
<th>Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>A, B, E, F</td>
<td>351</td>
<td>178818</td>
<td>81192</td>
<td>144424</td>
<td>61574</td>
<td>2076008</td>
</tr>
<tr>
<td>1994975</td>
<td>81192</td>
<td>144566</td>
<td>61530</td>
<td>2282263</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1995371</td>
<td>81214</td>
<td>144478</td>
<td>61596</td>
<td>2282659</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1994975</td>
<td>81192</td>
<td>144544</td>
<td>61530</td>
<td>2282241</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1995371</td>
<td>81236</td>
<td>144478</td>
<td>61618</td>
<td>2282703</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1788550</td>
<td>81192</td>
<td>144424</td>
<td>61574</td>
<td>2075740</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1995375</td>
<td>81236</td>
<td>144478</td>
<td>61618</td>
<td>2282707</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1995375</td>
<td>81236</td>
<td>144434</td>
<td>61596</td>
<td>2282619</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1788546</td>
<td>81192</td>
<td>144424</td>
<td>61574</td>
<td>2075736</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1788278</td>
<td>81192</td>
<td>144360</td>
<td>61574</td>
<td>2075404</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1788814</td>
<td>81192</td>
<td>144424</td>
<td>61574</td>
<td>2076004</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1789490</td>
<td>81170</td>
<td>144496</td>
<td>61552</td>
<td>2076708</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

| TABLE VIII | EXECUTION OF SEVENTH USE-CASE |
**H. Extensions**

Sample of Smart Contract

```solidity
pragma solidity ^0.4.15;

contract AtomicSwap {
    modifier isRefundable() {
        require(emptied == false);
    }
}

modifier isInitiator(uint _i) {
    require(msg.sender == party);
}

modifier isCorrectValue() {
    require(msg.value == amount);
}

function initiate () isCorrectValue payable public {
    initTimestamp = block.timestamp;
    party = msg.sender;
    value = msg.value;
    emptied = false;
    Initiated(
        initTimestamp,
        delta,
        msg.sender,
        msg.value
    );
}

function redeem2_3_1(bytes32 _secret, bytes _sig) public {
    if (msg.sender == counterParty){
        if (Verify("2-3-1", _sig) &&
            ripemd160(_secret) == hashedSecret[1] &&
            block.timestamp < initTimestamp + (graphDiam + 3) * delta)
        {
            unlocked[1] = true;
        }
    }
}

function redeem2_4_7_6(bytes32 _secret, bytes _sig) public {
    if (msg.sender == counterParty){
        if (Verify("2-4-7-6", _sig) &&
            block.timestamp < initTimestamp + (graphDiam + 4) * delta)
        {
            unlocked[6] = true;
        }
    }
}
```

Fig. 16. Smart Contract for an edge in Solidity (Part 1).

Fig. 17. Smart Contract for an edge in Solidity (Part 2).
function claim() public {
    bool lock = false;
    if(msg.sender == counterParty){
        for(uint i=0; i<unlocked.length; i++){
            if(unlocked[i] == false && leaders[i] == true){
                lock = true;
                break;
            }
        }
        if(lock == false){
            counterParty.transfer(value);
            emptied = true;
        }
    }
}

function refund() public isRefundable() {
    bool lock = false;
    if(msg.sender == party){
        for(uint i=0; i<unlocked.length; i++){
            if(unlocked[i] == false && leaders[i] == true){
                lock = true;
                break;
            }
        }
        if(lock && block.timestamp > initTimestamp + (graphDiam + 1 + 3) * delta){
            party.transfer(value);
            emptied = true;
        }
    }
}

Fig. 18. Smart Contract for an edge in Solidity (Part 3).