Trustless Cryptographic Gateway

Raymond Ng
raymond@katalystcoin.com
https://katalystcoin.org
Telegram : @raymondngkh

5 November 2017

License of White Paper

Abstract

As there are more blockchain application platforms created to allow the general users to create their own tokens, there is an immediate problem - the token in their respective platform would not be cross referenceable with one another. An Ethereum token would not be accessible on a Waves blockchain

Present Problems & Needs

Users of cryptoinventories (commonly misnamed as cryptocurrency by most media over the world) often have needs to allow their cryptoinventory to exist in different blockchain like Waves vs Ethereum.

For example, a Bitcoin owner may exchange part of the Bitcoin for Ethereum. The problem is more pronounced for less dominant cryptoinventories. For example, a owner of a waves token - wagerr - may want to exchange wagerr for an ethereum token, status.

Present methods proposed to solve the problems are as follows;

1) **Crypto Gateways** - On the wavesplatform for example, gateways are implemented to allow people to store other cryptoinventory on the Waves blockchain.

wBTC for example, is a waves token for Bitcoin. When existing on the Waves blockchain, transactions are tremendously fast and cheap. Waves would become even faster with the implementation of NG protocol which would support up to 1,000 over token transfer or smart contracts per second. Bitcoin on Waves blockchain can even be transacted with less than USD 0.01 fee.

However, the implementation of such gateways are not yet trustless. For us to use the gateways, we inherently have to trust the provider providing this service. The service provider of wBTC is acting as custodian for as long as the Bitcoin is transacting within the Waves blockchain.

The way it is implemented, requires the users to trust the gateways. As long as it is true, expert blockchain users would not trust the gateways with a lot of cryptoinventory.

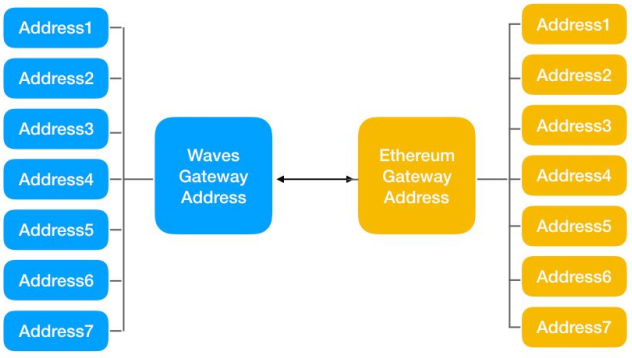
3) **Blockswap** - The promising software was actually implemented for some of the ICOs. Unfortunately, it did not turn out very well. Upon just superficial inspection of the code, it is discovered that the software coding does not seem to support Atomicity.

```
function moveToWaves(string wavesAddress, uint256 amount) {
        if (!transfer(owner, amount)) throw;
        WavesTransfer(msg.sender, wavesAddress, amount);
}
```

There are no provision of rollback should one of the transfer operations fail for whatever reasons.

Trustless Gateway

The basic principle of implementing trustless gateways is to rely on a mathematical principle of Atomicity.



The above figure is a basic framework from which cryptogateways is to be implemented according to the principles in this white paper.

If an ethereum token is to be transferred to Waves blockchain, it is to be sent to a designated Ethereum gateway address which is designed to issue a script

instructions with the intended recipient's wave address to the Waves blockchain. In the case where the instructions is transferred via the Internet, it must be encrypted via a session key that is generated for that specific instruction.

In the core of maintaining Atomicity, the pseudo code of the instruction must be in the following format that maintains Atomicity;

Ethereum-Waves Gateway

```
function movetowaves (waveaddress, amount) {
 // comment, potential failure point 1
 If (sendwavestoken(wavesgatewayaddress, recipientaddress)) {
  // sendwavestoken to write to attachment the txid from ethereum
  return 1;
 }

 else {
  // rolling back transaction
  sendbackethereumtoken (ethgatewayaddress, senderaddress) ;
 }
}
```

Looking at the **potential failure point 1** above. Given that the probability of such an event is unlikely to happen but the probability is not zero. This can happen if massive number of nodes or the processing node went down.

There must a routinely run function call that goes through the Ethereum blockchain and check for the corresponding transactions on the Waves blockchain. If the check reveals that there is no corresponding transactions on the Waves blockchain it means that the gateway transaction to transfer an Ethereum token to be on the Waves blockchain has failed.

However such checks should be a number of blocks away to give time for the Waves blockchain to reflect the transaction. The pseudo code of this would be as follows;

```
function checkatomicity(blocktime) {
 // check blockchain record for last checked block height
 $lastcheck = checklastrecordblockheight () ;
 $presentblock = returnpresentblockheight () ;

 for (x = $lastcheck; x <= $presentblock; x++) {
  for every transaction in block to gateway {
   // find non matching pair in the waves blockchain
   If (nomatchingpair($txid)) {
    // manifesting by sending tokens back
    rollback($txid, $senderaddress) ;
   }
  }
 }
```

```
}
```

The above example is for transferring from an Ethereum token on Ethereum blockchain to be an equivalent token on Waves blockchain. For waves token to travel from Waves blockchain to Ethereum blockchain, the logic is the same. So I reproduce the pseudo code here for completion purposes.

Waves-Ethereum Gateway

```
function movetoeth (ethaddress, amount) {
 // comment, potential failure point 1
 If (sendethtoken(ethgatewayaddress, recipientaddress)) {
  // sendethstoken to write to attachment the txid from ethereum
  return 1;
 }

 else {
  // rolling back transaction
  sendbackwavestoken (ethgatewayaddress, senderaddress) ;
 }
}

function checkatomicity(blocktime) {
 // check blockchain record for last checked block height
 $lastcheck = checklastrecordblockheight () ;
 $presentblock = returnpresentblockheight () ;

 for (x = $lastcheck; x <= $presentblock; x++) {
  for every transaction in block to gateway {
   // find non matching pair in the etherum blockchain
   If (nomatchingpair($txid)) {
    // manifesting by sending tokens back
    rollback($txid, $senderaddress) ;
   }
  }
 }
}
```

Crosschain Explorer

With the trustless crypto gateway implemented in the above mentioned manner, you can implement the crosschain explorer that would look like this;

Waves Sender	Ethereum Recipient	transaction id1
Ethereum Sender	Waves Recipient	transaction id2
Ethereum Sender	Waves Recipient	transaction id3
Ethereum Sender	Waves Recipient	transaction id4
Ethereum Sender	Waves Recipient	transaction id5
Waves Sender	Ethereum Recipient	transaction id6
Waves Sender	Ethereum Recipient	transaction id7
Ethereum Sender	Waves Recipient	transaction id8
Ethereum Sender	Waves Recipient	transaction id9

Concurrent Multiple Gateway

The mathematical logic expressed in this white paper here should be extensible for the implementation of concurrent multiple gateways.

For example, Monero is represented as XMR in cryptoexchanges.

You can do up a gateway for Waves and also Ethereum, applying the logic similarly in those gateways. So you can create a cryptogateway to implement the following tokens.

XMR present in its own native blockchain
wXMR present on Waves Blockchain
eXMR present on Ethereum Blockchain.

To be expanded on future versions of the draft White Paper.