swapblocks

Technical Specifications

Lance Rogers

March 3, 2019

1 SWAPBlocks Protocol

1.1 Base Protocol

Assets are registered on the network by creating a new asset object containing a unique_id, a pointer to the issuer, an asset_id created by hashing the pointer and unique_id, a prepended weight associated with the number of signatures needed for transactions to be considered valid and a valid_through_date. Once the asset is created it is signed by the issuer and broadcast to the network as an asset genesis transaction. The issuer then runs a node supporting the network and listening for transactions containing its managed assets.

When a transaction is detected containing a managed asset the issuer processes the transaction and decides to approve or reject it. This approval can be determined by any preset or future case set by the issuer. Some examples include verifying that the address receiving the asset is legally allowed to receive the asset, verifying that an appropriate transaction fee has been included in the transaction, verifying that an encrypted form sent with the transaction contains the correct information, etc.... It is the responsibility of the issuer to provide counterparties with asset transfer instructions and the responsibility of the consumer to interpret the instructions.

To approve a transaction the issuer must sign the transaction and rebroadcast the transaction to the network. Once the transaction is signed by

the issuer the transaction will be eligible for the next block. If the issuer does not approve the transaction it will time out and be cleared from all network nodes.

1.2 Consortium Extension

The base protocol forces all transactions pertaining to a specific asset through the issuer's node which can cause performance issues and creates a single point of failure. To solve these problems, issuers can extend the base protocol by forming consortiums that operate under consortium agreements. These consortium agreements keep track of member nodes, consortium governing rules and verification scripts. By joining a consortium the issuer agrees to process transactions of member nodes in accordance to the agreement and/or store the additional data attached to member transactions.

2 Consortium

Leaving an entity to verify all transactions pertaining to its managed assets can create a centralized bottleneck. To solve this problem a municipal node can join or create a consortium. Consortiums are groups of nodes that require the same transaction verification process. This can be the full transaction verification needed or a subset of the full transaction verification. Consortiums follow rules set out in a consortium agreement which includes a transaction verification script, consortium data packet storage and routing fees.

Consortiums create a semi-decentralized, semi-private transaction verification network that enables a central entity to benefit from the distributed properties of a decentralized network. Consortiums also reduce the amount of trust needed to interact with any particular central entity since the operating agreement can be read and verified on the blockchain.

Consortiums use a group signature scheme which allows any member node to publicly verify a transaction on behalf of the consortium without publicly exposing their identity. However the consortium manager is able to identify the signer of a message, allowing rogue group members to be exposed and kicked out of the consortium.

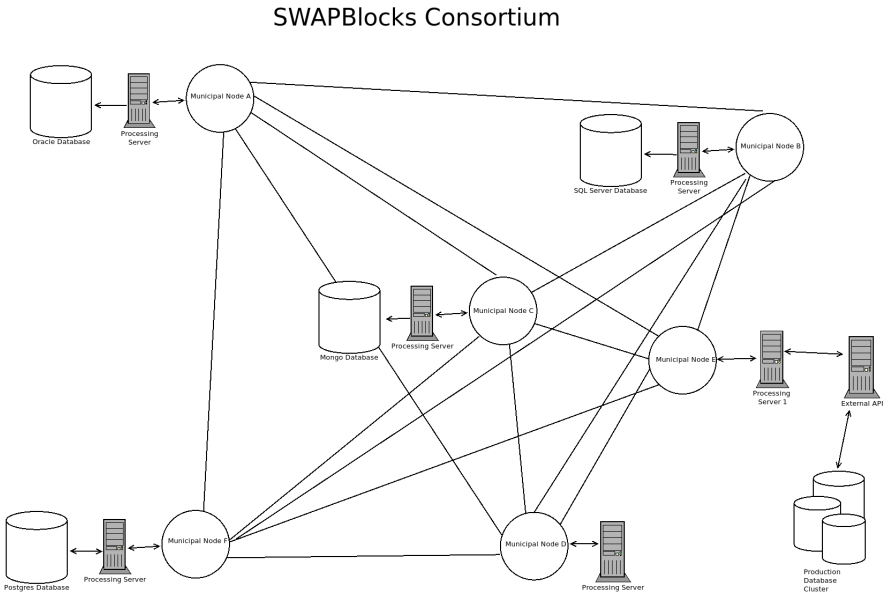

Figure 1: Consortium data storage infrastructure can vary between consortium nodes.

2.1 Consortium Agreement

A consortium agreement is stored on the blockchain and acts as the operating agreement for all member nodes. This agreement outlines the processing script, consortium members, number of member signature needed for amendments, routing fee prices, a list of all member nodes' static IP's, etc....

2.2 Consortium Routing

In order to limit the impact asset transactions have on the network as a whole mining nodes receive a portion of the attached fee for sending asset transactions to the correct consortium network. Consortiums can adjust the fee regularly to stay competitive for both consumers and miners or build out a network. For a miner to increase their chances of receiving a routing fee they can store a dictionary of consortiums' node addresses for quick routing.

2.3 Routing DAG

The routing DAG contains the information associated with the routing of asset transfer transactions. For example node X receives the transaction and performs standard verification, signs the transaction and broadcast it to the network where it is picked by node Y. Node Y looks up the consortium's address in its dictionary of consortiums, signs and sends the transaction to consortium member node A. Node A performs consortium verification, signs the transaction and routes the transaction directly to municipal node F. Node F verifies and signs the transaction. Now this transaction can be included in a block. When the transaction is included in a block the last 3 addresses to sign the transaction receive shares of the attached routing fee. Each signature in the routing DAG points to the previous one and can only contain one signature per level of verification. The first signature can be any node in the network, the second signature can be either a consortium member or the municipal node and the third can only be the municipal node. The amount of signatures needed is determined by the asset's assigned weight value.

3 Node States

SWAPBlocks assets view nodes in three different states. The state that the node is viewed in by the asset determines the level of verification the node is allowed to and should perform.

1. Standard Node State
    - Any network node that has no ownership stake in an asset
    - Checks that each transaction output points to valid UTXOs and that all signatures are correct.
2. Consortium Node State
    - Node belongs to the same consortium as the asset issuer, thus abiding by the same consortium agreement.
    - Typically responsible with processing or storing consortium data packets containing information needed for additional verification
3. Municipal Node State

- Node owned by the asset issuer.
- Responsible for verifying municipal data packets.
- Responsible with updating their own outside record keeping systems with status changes of registered assets.

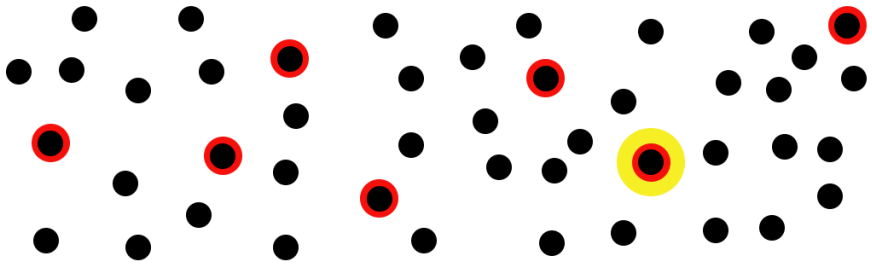

Figure 2: Node space in relation to some registered asset

Figure 2 represents the node space in relation to a registered asset. Each circle represents a node and each color represents the state that the node is in for a particular asset. The yellow circle represents the municipal node, meaning that the yellow node registered and manages the asset. The red circles represent consortium nodes, meaning that each red node is a member of the same consortium as the municipal node. The black dots represent standard network nodes that do not have any special permission or obligation to provide additional verification for this asset.

4 Asset Registration

Municipal nodes register assets in an asset genesis transaction. When an asset is registered there are only 4 required fields, the registering node's pub key, the consortium's address if there is one, a UAI and a valid_through date. This valid_through date is the date that the entity guarantees the information about the asset to be correct. The asset must be verified through an asset probe transaction to update the valid_through date.

4.1 Asset ID

Assets on the SWAPBlocks network are represented by a unique asset ID (UAI). The UAI is created by hashing the consortium address with the mu-

nicipal address and then hashing that with the municipal ID to create the base_id.

Now that the base ID has been formed the municipal node assigns a weight, determining the level of verification required to transfer the asset, and prepends the weight to the beginning of the asset ID.

4.2 Weights

Assets are assigned weights when they are first created. These weights are used to to determine the level of verification a transaction containing the asset needs before being eligible to be included in the next block. Weight is calculated based on the number of valid signatures a transaction has in it's routing DAG.

- Weight 0, Standard Verification
  - Default verification requiring the transaction to include valid UTXOs and signatures.
- Weight 1, Consortium Level Verification
  - Transactions must pass weight 1 verification and contain an additional signature from a consortium node.
  - Assets with this weight may require a consortium data packet to be sent along with every transfer transaction
- Weight 2
  - Transactions must pass weight 0 verification and include a municipal signature in the routing DAG.
  - Consortium nodes can perform consortium verification on the consortium data packet, sign the DAG and send the transaction directly to the municipal node, enabling the consortium node to receive some of the routing fee. If this verification has been done and signed off by a trusted consortium node the municipal node does not need to reverify and will only need to perform municipal verification.

5 Transactions

5.1 Transaction Types

There are 5 basic types of transactions in the SWAPBlocks ecosystem.

1. Currency
    - Involves the transfer of SBX coins from one address to another.
    - Currency transactions can be validated by checking UTXOs and the validation can be performed by every node in the network.
2. Asset Genesis
    - Transaction that registers an asset on the network for the first time.
3. Asset Transfer
    - Transaction used to send an asset from one account to another with no contingencies.
4. Asset Swap
    - Transaction in which two parties agree to swap ownership of two assets with the contingency that each transaction clears before the swap completes.
    - Contains maker and taker sections. Both sections must be complete and verified for the transaction to be eligible for the next block.
    - If the bid section is blank, then the transaction will be considered pending and can be filled by any bidder and rebroadcast to the network. These pending SWAP transactions will be stored in the SWAP order book, which is like a mempool for SWAP transactions.
5. Asset Probe
    - Transaction completed by the asset issuer or trusted third party that validates that the asset's condition is the same as the current record stored by the issuer.

- These will have to be completed periodically to ensure the asset has not been discarded, destroyed or transferred off network.
- Managing entities can charge fees to complete an asset probe transaction for an asset they manage.
- Asset probe transactions can also be completed by third party networks and will be linked to the assets transaction chain but the valid_through date will still remain the same.

5.2 SWAP Order Book (The SWAP DEX)

The SWAP order book is a specialized mempool for storing unverified SWAP transactions. What this means is that all SWAP transactions without a counterparty will be stored in a separate mempool for quick lookup. Decentralized exchange portals will be able to utilize this mempool to quickly place and fulfil orders.

5.3 Transaction Timeout

In order to limit the number of unverified transactions in the mempool, transactions are created with a max_verify_time. The max_verify_time is the latest time a transaction may be kept in the mempool. If a node has transactions in its mempool past the max_verify_time the node can discard the transactions. This helps to prevent spam transactions from bloating the mempool without needing to charge a fee.

For verified transactions that have not been included in a block and that contain data packets, the max_verify_time acts as a data_storage time limit. Once the max_verify_time has been reached all nodes can delete the data packets of non-consort assets, leaving only the hash value.

The max_verify_time will have a constant max time on network launch. In the future a proof of importance algorithm may be used to allow users to extend the max_verify_time.

To prevent users from issuing transactions with an extremely long max_verify_time, a transaction's max_verify_time must be less than or equal to a network constant as part of standard verification.

6 Permissioned Smart Contracts

SWAPBlocks enables asset transactions to include temporary encrypted data packets. These data packets are used by municipal and consortium nodes to make verification decisions. Asset transactions can include two separately encrypted data packets for multi-level verification and increased privacy.

1. Consortium Data Packet:
    - Contains information that can be verified by consortium member nodes.
    - Enables entities to validate standard form data quickly.
    - This data packet is typically spread across the consortium for backup and high availability.
    - Data is encrypted with a randomly generated key that is then encrypted with the consortium's public key and added to the transaction.
    - All consortium members have access to a shared key pair
2. Municipal Data Packet:
    - Contains information that is either confidential or that the consortium has not agreed to process.
    - Can only be verified by municipal nodes
    - Only stored by municipal nodes

Once the data packet has been verified by the approved node, the node signs the transaction. This signature is added to the routing DAG and the transaction is either routed to the next required level or broadcast to the network. When a transaction's DAG has reached its required weight, the transaction can be included in the next block and the data packet will be cleared from all non-member nodes.

If the data packet has not passed verification the verifying node will not sign the transaction leaving the transaction to time out and be cleared from all network nodes.

This verification process can be thought of as using permissioned smart contracts. In the case of municipal verification, registering accounts are the

only approved account to run the verification script on the asset. With consortium verification the smart contract will be defined in the consortium agreement.

7 Decided/Undecided Protocol Specifications

1. All hashes should use SHA-3 256 (Keccak)
2. Default compression and encryption of data packets not yet decided
   (a) Will need to take into consideration compression duplication of common values
   (b) Need to consider size of data packet and padding
   (c) Need to consider time to send data transmission
   (d) Need to consider chaining transactions to send large packets
3. Consortiums will be based on a Group Signature Scheme as first described by Chaum and Van Heyst
   (a) Particular scheme not yet decided
   (b) Possibilities:
      i. "Fair traceable multi-group signatures"
      ii. "Democratic Group Signatures with Threshold Traceability" by Zheng et al.
      iii. "Short group Signatures with Distributed Traceability" by Blomer et al.