K B A

Transaction Families

Transaction Family

- A transaction family includes:
  - A transaction processor to define the business logic for application
  - A data model to record and store data
  - A client to handle the client logic for your application

Transaction Family

Sawtooth includes following transaction families:

- Settings Transaction Family
- BlockInfo Transaction Family
- IntegerKey Transaction Family

Settings Transaction Family

- Settings are stored in state
- On-chain configuration settings
- Settings TP is required in production environment
- Setting data consists of setting/value pairs

Setting	value
sawtooth.validator.max_transactions_per_block	100
sawtooth.consensus.algorithm	poet
sawtooth.validator.transaction_families	{"family": "intkey", "version": "1.0"}, {"family":"sawtooth_settings", "version":"1.0"}

Settings Transaction Family

- sawset command is used to work with settings proposals

- For changing a setting, first a proposal is created

  #sawset proposal create [--url URL] [-k KEY]   <setting>=<value>

  eg: # sawset proposal create --url http://rest-api:8008 --key /root/.sawtooth/keys/my_key.priv sawtooth.validator.max_transactions_per_block='100'

- sawtooth.settings.vote.authorized_keys contains list of public keys allowed to vote or create proposal.

- sawtooth.settings.vote.approval_threshold contains minimum number of votes required to accept or reject a proposal (default: 1)

Settings Transaction Family

- Proposal will be recorded in sawtooth.settings.vote.proposals, with one "accept" vote counted

- Lists the currently proposed (not active) settings using the command
  #sawset proposal list --url http://rest-api:8008

- Votes for a specific settings change proposal using the command
  #sawset proposal vote [--url URL] [-k KEY] proposal_id {accept,reject}

- Lists existing settings using the command
  #sawtooth settings list --url http://rest-api:8008

Settings Transaction Family

Addressing

- Setting keys are broken into four parts, based on the dots in the string.
- A short hash computed (the first 16 characters of its SHA256 hash in hex) for each part and is joined into a single address
- Settings namespace (000000) added at the beginning

eg: sawtooth.settings.vote.proposals

```
>>> '000000' + hashlib.sha256('sawtooth'.encode()).hexdigest()[:16] + \
   hashlib.sha256('settings'.encode()).hexdigest()[:16] + \
   hashlib.sha256('vote'.encode()).hexdigest()[:16] + \
   hashlib.sha256('proposals'.encode()).hexdigest()[:16]
'000000a87cb5eafdcca6a8b79606fb3afea5bdab274474a6aa82c1c0cbf0fbcaf64c0b'
```

Settings Transaction Family

Execution

- Current values of sawtooth.settings.vote.authorized_keys is read from the state
- Valid proposals will be recorded in the SettingProposals stored in sawtooth.settings.vote.proposals, with one “accept” vote counted
- Value of sawtooth.settings.vote.approval_threshold is read from the state
- If the “accept” vote count is equal to or above the approval threshold, the proposal is applied to the state
- If the “reject” vote count is equal to or above the approval threshold, then it is deleted from sawtooth.settings.vote.proposals

BlockInfo Transaction Family

- For storing block information
- Stores BlockInfoConfig and BlockInfo
- BlockInfoConfig contains most recent block number, oldest block number, target number of blocks and network time synchronization tolerance
- BlockInfo contains block number, previous_block_id, signer_public_key, header_signature and timestamp

BlockInfo Transaction Family

- BlockInfo transactions are added to a block by a BlockInfo Injector

```
# sawset proposal create --key /root/.sawtooth/keys/my_key.priv
sawtooth.validator.batch_injectors=block_info --url http://rest-api:8008
```

- Validation rules should ensure that only one transaction of this type is included at the beginning of the block

```
# sawset proposal create --key /root/.sawtooth/keys/my_key.priv
sawtooth.validator.block_validation_rules='NofX:1,block_info;XatY:block_info,
0;local:0' --url http://rest-api:8008
```

Only N of transaction type X can be included in a block.

Transaction type X can only occur at position Y in a block.

BlockInfo Transaction Family

Addressing

- The top-level namespace of this transaction family is 00b10c
- BlockInfoConfig namespace: 00b10c01
- Block info namespace: 00b10c00
- Additional information about blocks will be stored in state under the block info namespace at an address derived from the block number
    - Convert block_num to a hex string and remove the leading "0x"
    - Left pad the string with 0s until it is 62 characters long
    - Concatenate the block info namespace and the string obtained

```
>>> '00b10c00' + hex(block_num)[2:].zfill(62))
```

BlockInfo Transaction Family

Execution

- The payload is checked to make sure it contains a valid block number, the previous block id etc.
- If the config does not exist, then add the config and block info.
- If the config does exist, then update the config after validation
- If number of blocks stored in state is greater than the target number of blocks, delete the oldest BlockInfo message from state.
- Calculate the address for the new block number, write the new BlockInfo message to state at the address computed for that block.

IntegerKey Transaction Family

- Allows users to set, increment, and decrement the value of entries stored in a state dictionary

- An IntegerKey family transaction request is defined by the following values:
  - verb (eg: set, inc, dec)
  - name
  - value

```
# intkey set <name> <value>
```

```
eg: # intkey set MyKey 65 --url http://rest-api:8008
```

IntegerKey Transaction Family

set	:	Sets an intkey value (eg: # intkey set MyKey 65)
inc	:	Increments an intkey value (eg:# intkey inc MyKey 5)
dec	:	Decrements an intkey value (eg: # intkey dec MyKey 10)
show	:	Displays the specified intkey value (eg: # intkey show MyKey)
list	:	Displays all intkey values (eg: # intkey list)

IntegerKey Transaction Family

Addressing

- The first 6 characters of the address are the first 6 characters of a sha512 hash of the IntegerKey namespace prefix: “intkey”
- The following 64 characters of the address are the last 64 characters of a sha512 hash of the entry Name

eg: an IntegerKey address could be generated as follows:

```
>>> hashlib.sha512('intkey'.encode('utf-8')).hexdigest()[0:6] + \
    hashlib.sha512('name'.encode('utf-8')).hexdigest()[-64:]
'1cf126cc488cca4cc3565a876f6040f8b73a7b92475be1d0b1bc453f6140fba7183b9a'
```

IntegerKey Transaction Family

Execution

- The transaction request Verb , Name, and Value are checked. If any of these values are empty, the transaction is invalid.

- Name must be a string with a maximum of 20 characters. Value must be a 32-bit unsigned integer.

- If the Verb is ‘set’, the state dictionary is checked to determine if the Name associated with the transaction request already exists. If it does already exist, the transaction is invalid.

Sawtooth Transaction Family Prefixes

TRANSACTION FAMILY NAME	PREFIX
settings	000000
blockinfo	00b10c
intkey	1cf126

THANK YOU