AIP[1]-011: Service Account & Transfer

platfowner, liayoo 2021-02-08

Goals

- Provide new features for 1) service accounts (like business bank account) and 2) money transfer from/to them

Problem Definition

So far we were providing only individual accounts and now we need service accounts for business-like use cases (e.g. credit charging, payment, etc).

Requirements

We have the following requirements for service accounts:

- (Value-holding balance)
  - Service accounts have value-holding balances (i.e., sum(service account balances) + sum(individual account balances) = total circulated tokens)
- (Transfer method)
  - Users or other service admins can deposit their money to service accounts
  - Admins of a service account can withdraw money from the service account
- (Configurable)
  - Service accounts have explicit admin addresses
  - Admin addresses are configurable (add / remove)

Proposed Design

Key Ideas

- (Value-holding balance)
  - Use a separate db path for service accounts (**/service_accounts/<account name>**)
  - Allow only **non-address** account names, i.e., starts with alphabetical characters (not numeric characters like '0x')
- (Transfer method)

[1] AI Network Improvement Proposal. Visit https://docs.ainetwork.ai for the full list.

    - Use the existing transfer feature
        - /transfer/<FROM user address>/<TO service account name>/<account key>/value
        - /transfer/<FROM service account name>/<TO user address>/<account key>/value
        - /transfer/<FROM service account name>/<TO service account name>/<account key>/value
    - Discriminate user accounts and service accounts via naming patterns (e.g. user addresses start with ‘0x’ while service accounts start with alphabetical characters)
- (Configurable)
    - Admin can add / remove other admin addresses
    - Admin can remove admin addresses (final admin cannot be removed)

Design Details

Use Cases

Deposit / Withdraw (AS-IS):

- _deposit
    - /accounts/<userAddr>/balance -> /deposit_accounts/<serviceName>/<userAddr>/balance
- _withdraw
    - /deposit_accounts/<serviceName>/<userAddr>/balance -> /accounts/<userAddr>/balance

Payment (AS-IS):

- _pay (by service admin)
    - /accounts/<adminAddr>/balance -> /payment/<serviceName>/<userAddr>/balance
- _claim (by service admin)
    - /payment/<serviceName>/<userAddr>/balance -> /accounts/<adminAddr>/balance

Service Account Name Mapping

Service type / service name / account key tuples can be mapped into service account names as follows:

- (<service type>, <service name>, <account key>) => ‘<service type>|<service name>|<account key>’

Usually, the address of the user who uses the service can be used as the account key value, so '<service type>|<service name>|<address>'. For example, /service_accounts/**payments|collaborative_ai|0x_asdf**/ .

To avoid conflicts, we can pose some restrictions on service type and service name:

- Only alpha-numeric characters are allowed
- Should start with alphabetic characters (not e.g. '0x')

Internal Use of Transfer Feature

For flexibility, we allow **direct** use of 'transfer feature' by other native functions. This is done by rule configs:

```
  "transfer": {
    "$from": {
      "$to": {
        "$key": {
          "value": {
            ".write": "(auth.addr === $from || auth.fid === '_deposit' ||
auth.fid === '_withdraw' && ..."
          },
          "result": {
            ".write": "auth.fid === '_transfer' || auth.fid === '_deposit' ||
auth.fid === '_withdraw'"
          }
        }
      }
    }
  }
```

Account Admin Configuration

Permission for the configuration of account admins is given by rule configs:

```
  "service_accounts": {
    "$service_type": {
      "$service_name": {
        "$account_key": {
          "admin": {
            "$admin_addr": {
              ".write": "(getValue('/service_accounts/' + $service_type + '/'
+ $service_name + '/admin/' + auth.addr) !== null || auth.fid === '_deposit'
|| auth.fid === '_withdraw' && ..."
            }
          },
          "balance": {
            ".write": "auth.fid === '_transfer' || auth.fid === '_deposit' ||
auth.fid === '_withdraw' && ..."
          }
```

```
            }
        }
    }
}
```

Conclusion

A deign is provided for service account & its value transfer with the following requirements:

- Value-holding balance
- Transfer method
- Configurable

Document History

Date	Who	Change	Notes
2021-02-08	platfowner	Initial draft	
2021-02-17	platfowner, liayoo	Review & revise	
2021-05-07	platfowner	Published	
2021-05-12	platfowner	Github IDs Link to full list	
2021-05-12	platfowner	Republished	