Acme Exchange
Enable an Exchange to be a Ripple Gateway

Contents

1. Ripple Gateway Explained in Diagrams
   a. Notation
   b. State of Exchange Before Ripple Integration
   c. Updating Core Accounting System at Exchange
   d. Introduction to Trust Lines and Issuances
   e. Trust Lines - Hot and Cold Ripple Wallets
   f. Trust Lines - Exchange With Three Users
   g. Deposit From Exchange to Ripple Wallet
   h. Transactions on Ripple
   i. Withdrawal From Ripple Wallet to Exchange

2. Prerequisites
   a. Wallet Generation
   b. Ripple Gateway Checklist
   c. Trust Requirement

3. Flow of Funds
   a. Deposit from Exchange to Ripple Wallet
      i. GET a Payment Path
      ii. POST Payment to the Ripple Network
      iii. GET a Payment Confirmation
   b. Withdrawal from Ripple Wallet to Exchange
      i. GET notification

4. Appendix
   a. Setting a Trust Line on the Ripple Trade Client

Version 0.2
December 22, 2014
**Introduction**

An existing online financial service, such as a payment system or digital currency exchange, can provide additional value to customers by acting as a Ripple Gateway. This provides users the ability to send cross-currency payments to users linked by other Ripple Gateways, and potentially provides a new revenue source for balances deposited, withdrawn, or transferred in Ripple.

Expanding an existing exchange system to support Ripple is a relatively simple task. This document explains the concepts necessary to set up a system, and covers the details of doing so. In this document, we use a fictional online currency exchange named "ACME" and its users as examples of how ACME can expand its business to include being a Ripple Gateway.

**Ripple Gateway Explained**

A Ripple Gateway is an entity that exchanges balances in the Ripple Network for balances outside the Ripple Network — in other words, performing deposits and withdrawals from Ripple. Typically, a Gateway holds money (or other assets of value) outside of Ripple, and creates *issuances* in the Ripple Network to represent them. Within Ripple, issuances can be sent, traded, and exchanged atomically without the gateway’s intervention.

**Notation**

BTC - Digital asset with ownership verified on the Bitcoin blockchain.
BTC@Acme - Digital asset issued by Acme Gateway on the Ripple network.
Acme Exchange - A currency exchange with pre-existing core accounting systems.
Ripple Acme - The Ripple gateway managed by Acme Exchange.
Alice, Bob, Charlie - Users of Acme Exchange and Ripple Acme.
State of Exchange Before Ripple Integration

Deposit From Alice to Acme Exchange

Figure 1 assumption:
- Acme Exchange only has two users, Alice and Bob

Figure 1 notes:
- Acme exchange should wait ample time for each deposit to clear.
State of Exchange Before Ripple Integration

Withdrawal From Acme Exchange to Alice

- Figure 2 assumption:
  - Acme Exchange only has Alice and Bob’s funds on deposit.
- Figure 2 notes:
  - Acme Exchange always has enough assets to payout on demand.
  - Alice may withdraw any amount up to 1 BTC.
    - Acme Exchange should set minimum withdrawal amount in their terms.
Top of Figure 3 assumptions:
- Acme Exchange has three users: Alice, Bob, and Charlie.
  - Each user has made a deposit of BTC into Acme Exchange.
  - The sum of all BTC on deposit is stored in Acme Exchange’s BTC wallet.
  - Trading on Acme Exchange is enabled for Alice, Bob, and Charlie.
- Acme Exchange must have an existing proprietary exchange

Bottom of Figure 3 assumptions:
- Acme Exchange is a Ripple enabled exchange.
- Acme Exchange has a new BTC wallet to finance funds issued on Ripple.
  - Note that this step is not strictly necessary.

Bottom of Figure 3 - Actions taken:
- Alice has sent 1 BTC from Acme Exchange to her Ripple wallet.
  - See API Example: Deposit From Exchange to Ripple Wallet
- Charlie has sent 2 BTC from Acme Exchange to his Ripple wallet.
- Acme Exchange has moved 3 BTC to their BTC wallet financing Ripple.
  - Note that this step is not strictly necessary.
- Acme Exchange updated accounting system for Alice and Charlie
Trust Lines, Gateways, and Issuances - Introduction

A “trust line” is a link between two accounts in Ripple that represents an explicit statement of willingness to hold gateway debt obligations, also called issuances. All currencies on Ripple, except for the native cryptocurrency XRP, are represented as issuances.

A gateway deposit is when someone sends currency (outside the Ripple Network) to a gateway in exchange for issuances on the Ripple network. An issuance represents a promise that whoever holds the issuance can redeem it with the issuer on the Ripple Network for that amount of currency (off the Ripple Network). Doing so is a gateway withdrawal.

All issuances are held within a trust line to the account that issued them. Issuances get their value from the promise that the gateway issuing them will honor the obligation that the issuances represent. There is no computer system that can force a Ripple gateway to honor that obligation, so it is important that a gateway’s policies for withdrawals is clear and well-publicized.

Trust Lines - Hot and Cold Ripple Wallets

- Acme Exchange should have two Ripple Wallets to comply with security best practice
  - See - Ripple wallet generation requirement
  - Acme Cold
    - Issues currency on the Ripple network.
    - Used to receive payments from users on Ripple.
  - Acme Hot
    - Has established trust to Acme Cold for BTC@Acme.
      - See - Trust requirement
      - Amount of trust should increase proportional to Ripple volume.
      - Used to make payments to users on Ripple.
Trust Lines - Exchange With Three Users

- **Figure 5 assumptions:**
  - Alice trusts Acme cold for 4 BTC@Acme on Ripple.
  - Bob trusts Acme cold for 7 BTC@Acme on Ripple.
  - Charlie trusts Acme cold for 3 BTC@Acme on Ripple.
- **Action taken on Ripple network from Figure 3**
  - Ripple Acme has issued 1 BTC@Acme to Alice on Ripple.
  - Ripple Acme has issued 2 BTC@Acme to Charlie on Ripple.
- **Figure notes:**
  - Payments to users are always sent from Ripple Acme’s hot wallet on Ripple.
  - Acme cold periodically fills Acme hot with BTC issued by Acme on Ripple.
Deposit From Exchange to Ripple Wallet

- **Use case:**
  - Alice elects to move 1 BTC from Acme Exchange to her Ripple Wallet.
  - See the core accounting state prior to this transaction in Figure 3.

- **Figure 6 - Action taken on Acme Exchange:**
  - 1 BTC is moved to the BTC wallet that finances assets on Ripple.
  - Alice's balance is debited in the Acme Exchange core accounting system.
  - Ripple’s balance is credited in the Acme Exchange core accounting system.

- **Figure 7 - Action taken on Ripple network**
  - Acme sends Alice 1 BTC@Acme on the Ripple network.
  - See API Example: [Deposit From Exchange to Ripple Wallet](#)

*Action items should be taken with timing that completely eliminates risk.*
Transactions on Ripple

Use case:
- Alice sends a Ripple payment of 2 BTC@Acme to Bob.
  - Alice must own at least 2 BTC@Acme to make payment.
  - Bob must trust Ripple Acme (cold) for at least 2 BTC@Acme to receive.
  - Payment happens atomically on Ripple.

Figure 8 - Action taken on Acme Exchange:
- Transactions on the Ripple network do not require any action by Acme Exchange.
  - Acme must only track outstanding BTC@Acme issued on Ripple.

Figure 9 - Action taken on the Ripple network:
- The decentralized Ripple exchange
  - Anyone may post offers to buy/sell BTC@Acme on Ripple.
  - Acme Exchange should honor their debt obligations issued on Ripple.
  - Acme should still confirm their customers’ identities before issuing their withdrawals, especially since the users withdrawing may be different from the users depositing.
Withdrawal From Ripple Wallet to Exchange

Use case:
- Bob elects to move 1 BTC@Acme from his Ripple Wallet to Acme Exchange.

Figure 10: Action taken on Acme Exchange
- 1 BTC is moved from the BTC wallet that finances assets on Ripple.
- Bob’s balance is credited in the Acme Exchange core accounting system.
- Ripple’s balance is debited in the Acme Exchange core accounting system.

Figure 11: Action taken on Ripple network
- Bob redeems 1 BTC@Acme at Acme Exchange on Ripple
  - See API Example: Receiving a Payment

Draft Version 0.2
December 22, 2014
Prerequisites

Wallet Generation

It is strongly recommended that Ripple gateways employ a hot and cold wallet. This enforces a separation of roles that promotes strong security.

The cold wallet should remain offline, and serves as the asset issuer. The hot wallet should be funded by the cold wallet and is available to make payments from the gateway to the gateway’s users. The cold wallet should only be used to fill up the hot wallet, and the gateway should notify the system administrator when the hot wallet contains a low balance.

Reference:
https://wiki.ripple.com/Hot__cold_wallet_setup

Ripple Gateway Checklist

Notable items for starting a gateway:

- Set the DisallowXRP flag for the cold and hot wallets:
  - Avoids confusion when users inadvertently send XRP to the gateway.
- Set the DestTagRequired flag for the cold and hot wallets:
  - Each user of Acme gateway is mapped to a unique integer Destination Tag.
  - This flag rejects payments to the gateway that omit a Destination Tag.
  - Payments with unknown destination tags should be bounced.
- Set the RequireAuth flag for the hot wallet:
  - Disallows users to hold balances sent by the hot wallet.

Reference
https://wiki.ripple.com/Gateway_Integration_Manual#Check_list_for_typical_gateway_setup
https://wiki.ripple.com/Require_a_destination_tag_for_inbound_payments

Trust Requirement

Trustlines in Ripple are an explicit statement of willingness to hold gateway debt obligations.

- Acme’s hot wallet must trust Acme’s cold wallet for a fixed amount.
- Users of Acme Gateway must trust Acme’s cold wallet.

Ripple REST API Reference:
https://ripple.com/build/ripple-rest/#grant-trustline

Reference:
Example Accounts

For the purposes of demonstration, the following three Ripple names will be used:

<table>
<thead>
<tr>
<th>Ripple name</th>
<th>Ripple address</th>
<th>Functionality</th>
</tr>
</thead>
<tbody>
<tr>
<td>~user-ripple</td>
<td>rXtt8v4HLz88QD3jTdTxB1YPbAcxHdPri</td>
<td>An Acme user</td>
</tr>
<tr>
<td>~gateway-hot</td>
<td>raxrufdDLKTJ17Fvd1WVmWKMx7Vunvbm3u</td>
<td>Acme’s hot wallet</td>
</tr>
<tr>
<td>~gateway-cold</td>
<td>rKeMKZypwiPdush32L4xjxHpxdwpSymN</td>
<td>Acme’s cold wallet</td>
</tr>
</tbody>
</table>

Flow of Funds

Integration requires two fundamental flows:

- User deposits from exchange to Ripple wallet.
  - Example: Alice deposits 1 BTC into Ripple
  - See Figure 7
- User withdrawals from Ripple wallet to exchange.
  - Example: Bob withdrawals 1 BTC from Ripple
  - See Figure 11

Each of these processes will be outlined with corresponding Ripple API calls below.

Deposit from Exchange to Ripple Wallet

This example will assume that Acme Gateway:

- Has one user, Alice.
- Has an internal core accounting system to track their users balances.
- Has one hot and one cold Ripple wallet.
  - Hot wallet should make outgoing payments to Ripple.
- Has a user interface for Alice to deposit into Ripple.
Acme Gateway should provide the following user experience:

1. Alice signs in to Acme Gateway.
2. Alice selects deposit to Ripple.
3. Acme Gateway asks Alice to provide a Ripple name.
4. Alice enters an amount that:
   a. Is less than her current balance in the core accounting system (1 BTC)
   b. Is less than her current trustline limit in Ripple (4 BTC@Acme)
5. Alice clicks accept.

This example assumes that a user Alice:

- Has been properly verified for local KYC and AML policies by Acme Gateway.
- Deposited 2 BTC to Acme Gateway:
  - Credited for 2 BTC in the core accounting system
- Owns a Ripple wallet:
  - Uses www.RippleTrade.com to manage her Ripple wallet.
- Established a trust line to Acme Gateway for 4 BTC@Acme.
  - Alice is willing to hold up to 4 BTC issued by Acme on the Ripple network.
  - Notice that this number is independent of Alice’s current balance at Acme.
  - User with insufficient trust should be redirected to www.RippleTrade.com
  - Reference: https://wiki.ripple.com/Ripple_URIs#trust
- Understands Ripple introductory material provided by Acme Gateway.

Example deposit:

- Alice will withdrawal 1 BTC from Acme Gateway into her Ripple wallet.
  - Alice’s balance in the core accounting system will be debited 1 BTC.
    - See Figure 3 - Overview of accounting at Acme Exchange.
  - Acme Gateway will issue Alice 1 BTC@Acme on the Ripple network.
    - See Figure 7 - Deposit from Acme Exchange to Ripple Wallet.

On the backend, Acme gateway will complete the RESTful API sequence:

1. GET a payment path in the Ripple network.
   - Search for path from Acme Hot to Alice.
2. POST a payment to the Ripple network.
3. GET a payment confirmation.
1. GET a payment path

GET /v1/accounts/{:source_address}/payments/paths/{:dest_address}/{:dest_amount}

Response:
{
  "success": true,
  "payments": [
    {
      "source_account": "raxrufdDLKTJ17Fvd1WVmWKMx7Vunvb3u",
      "source_tag": "",
      "source_amount": {
        "value": "1",
        "currency": "BTC",
        "issuer": "raxrufdDLKTJ17Fvd1WVmWKMx7Vunvb3u"
      },
      "source_slippage": "0",
      "destination_account": "rfXtt8v4HLz88QD3jTdTxB1YPbAcxHdPri",
      "destination_tag": "",
      "destination_amount": {
        "value": "2",
        "currency": "BTC",
        "issuer": "rKeMKZyxpwiPdush32L4XjxxHpduxwpSymN"
      },
      "invoice_id": "",
      "paths": [
        [
          {
            "account": "rKeMKZyxpwiPdush32L4XjxxHpduxwpSymN",
            "type": 1,
            "type_hex": "0000000000000001"
          }
        ]
      ],
      "partial_payment": false,
      "no_direct_ripple": false
    }
  ]
}
Note:

- The gateway should parse out a payment object for use in the POST request.

Ripple REST API Reference:

https://ripple.com/build/ripple-rest/#prepare-payment

2. POST a payment to the Ripple network

POST /v1/accounts/{source_address}/payments

Payload:

```javascript
{
    secret: "ssss...",
    client_resource_id: "ee1bd8f4-28eb-4aa5-9559-f4d7a160b25b",
    payment: {
        source_account: "raxrufdDLKTJ17Fvd1WVmWMx7Vunvbm3u",
        source_tag: "",
        source_amount: {
            value: "1",
            currency: "BTC",
            issuer: "raxrufdDLKTJ17Fvd1WVmWMx7Vunvbm3u"
        },
        source_slippage: "0",
        destination_account: "rfXtt8v4HLz88QD3jTdTxB1YPbAcxHdPri",
        destination_tag: "",
        destination_amount: {
            value: "2",
            currency: "BTC",
            issuer: "rKeMKzxywpwPdush32L4XjxxHpxdwpSymN"
        },
        invoice_id: "",
        paths: [
            [
                {
                    account: "rKeMKzxywpwPdush32L4XjxxHpxdwpSymN",
                    type: 1,
                    type_hex: "000000000000000000000001"
                }
            ]
        ],
        partial_payment: false,
        no_direct_ripple: false
    }
}
```
Response:
{
    success: true,
    client_resource_id: "ee1bd8f4-28eb-4aa5-9559-f4d7a160b25b",
    status_url: "http://<ripple-rest-server>/v1/accounts/raxrufdDLKTJ17Fvd1WVmWKx7Vunvbm3u/payments/ee1bd8f4-28eb-4aa5-9559-f4d7a160b25b"
}

Notes:
● The client_resource_id is a universally unique identifier.
  ○ Used to reference the payment in the local Ripple REST database.
  ○ UUID generation included as a Ripple REST endpoint.
● The status_url checks the status of the request.
  ○ Returns the GET confirmation of payment.

Ripple REST API Reference:
https://ripple.com/build/ripple-rest/#submit-payment
https://ripple.com/build/ripple-rest/#create-client-resource-id
3. GET confirmation of payment

GET /v1/accounts/{:source_address}/payments/{:id}
Response:
{
  success: true,
  payment: {
    source_account: "raxrufdDLKTJ17Fvd1WVmWKMx7Vunvbm3u",
    source_tag: "",
    source_amount: {
      currency: "BTC",
      issuer: "rKeMK2ypwiPdush32L4XjxxHpdxwpSymN",
      value: "2"
    },
    source_slippage: "0",
    destination_account: "rfXtt8v4HLz88QD3jTdTxB1YPbAcxHdPrf",
    destination_tag: "",
    destination_amount: {
      currency: "BTC",
      issuer: "rKeMK2ypwiPdush32L4XjxxHpdxwpSymN",
      value: "2"
    },
    invoice_id: "",
    paths: "[]",
    no_direct_ripple: false,
    partial_payment: false,
    direction: "outgoing",
    state: "validated",
    result: "tesSUCCESS",
    ledger: "9751209",
    hash: "DB35BC6A212FB6531FD6130E41350D50BE73EF4647E3E3F90B6773DBB78C57E1",
    timestamp: "2014-11-04T18:57:40.000Z",
    fee: "0.012",
    source_balance_changes: [
      {
        value: "-1",
        currency: "BTC",
        issuer: "rKeMK2ypwiPdush32L4XjxxHpdxwpSymN"
      },
      {
        value: "-0.012",
        currency: "XRP",
        issuer: ""
      }
    ],
    destination_balance_changes: [
      {
        value: "1",
        currency: "BTC",
        issuer: "rKeMK2ypwiPdush32L4XjxxHpdxwpSymN"
      }
    ]
  }
}

Notes:

- The {id} field is either a client_resource_id or a transaction hash
- The gateway should parse out and cross reference:
  - state: "validated"
    - The transaction has been included in a validated ledger
    - Does not imply the payment has succeeded.
  - result: "tesSUCCESS"
    - The transaction was successfully executed
  - If "partial_payment" flag is true
    - Then check "destination_balance_changes" array to see how much currency was actually delivered.
- source_balance_change
- destination_balance_change

Processing a payment can take several seconds to complete, depending on the consensus process. If the payment does not exist yet, or has not been validated, you should wait a few seconds before checking again.

Ripple REST API Reference:

https://ripple.com/build/ripple-rest/#confirm-payment
Withdrawal from Ripple Wallet to Exchange

This example will assume that Acme Gateway:

- Has an internal core accounting system to track users balances.
- Has one hot and one cold Ripple wallet.
  - Cold wallet should be monitored for incoming payments from Ripple.
- Has a user interface for Bob to withdrawal from Ripple.

Acme Gateway should provide the following user experience:

1. Bob signs in to Acme Gateway.
2. Bob selects deposit to Acme Gateway.
3. Acme Gateway provides a destination tag and a Ripple Address (cold wallet).

This example assumes that a user Bob:

- Has been properly verified for local KYC and AML policies by Acme Gateway.
- Owns BTC@Acme in his Ripple Wallet.

Example withdrawal:

- Bob withdrawals 1 BTC from his Ripple Wallet to Acme Gateway:
  - Bob sends Acme Gateway's cold wallet 1 BTC@Acme on the Ripple network.
    - See Figure 11 - Example withdrawal from Ripple
  - Bob’s balance in the core accounting system will be credited 1 BTC.
    - See Figure 3 - Overview of accounting at Acme Exchange.

On the backend, Acme gateway should constantly poll for notifications on Ripple:

1. GET next notification on the cold wallet
2. Parse the incoming payment
   a. Confirm transaction has been validated.
   b. Store the transaction hash and destination tag.
   c. Update the core accounting system.
1. GET notification

Notification Tree

Use notification_url's to walk an accounts transaction history.

< previous_notification_url > ⇒ < next_notification_url > ⇒ []

Endpoint

An ID is needed to walk the notification queue. An ID is defined by either:

- **client_notification_id**
  - A local ID contained in the Ripple REST database.
  - Found in JSON response after POSTing a payment.

- **hash**
  - A global ID which is constant throughout the Ripple network.
  - Contained in any historical transaction of a given account
Mid Queue Example

An example response from a notification which is not the most recent in a given accounts historical transactions.

GET /v1/accounts/{:source_address}/notifications/{:id}
Response:
{
  "success": true,
  "notification": {
    "account": "rKeMKZyxpwiPdush32L4xHpxdwpSymN",
    "type": "payment",
    "direction": "incoming",
    "state": "validated",
    "result": "tesSUCCESS",
    "ledger": "9809657",
    "hash": "29698F1C73A9D5F3821F6BA02A374F42C072364E1501D38AFB17F946E8B72EFD",
    "timestamp": "2014-11-07T22:30:50.000Z",
    "transaction_url": "http://<ripple-rest-server>/v1/accounts/rKeMKZyxpwiPdush32L4xHpxdwpSymN/payments/29698F1C73A9D5F3821F6BA02A374F42C072364E1501D38AFB17F946E8B72EFD",
    "previous_hash": "9CB583FBB665DB2362A7AED60B5207113265FBE76D145F31A58ACC3AC8A3DB4D",
    "previous_notification_url": "http://<ripple-rest-server>/v1/accounts/rKeMKZyxpwiPdush32L4xHpxdwpSymN(notifications)/9CB583FBB665DB2362A7AED60B5207113265FBE76D145F31A58ACC3AC8A3DB4D",
    "next_hash": "FD3394055A9165377C007103438A445181286668180D982F0A98A7B242397C9",
    "next_notification_url": "http://<ripple-rest-server>/v1/accounts/rKeMKZyxpwiPdush32L4xHpxdwpSymN(notifications)/FD3394055A9165377C007103438A445181286668180D982F0A98A7B242397C9"
  },
  "client_resource_id": "7fec249e-091b-4c13-82b6-283f5b2b29ad"
}

Draft Version 0.2
December 22, 2014
Notes:

- validated
  - Transaction should only be may safely be considered final.
- next_notification_url
  - Is non-empty implying this is not the most recent transaction of the account.
- client_resource_id
  - Only present if {:ID} is given as a client_resource_id
- transaction_url
  - Confirm the details of a payment - See example above.

Most Recent Historical Transaction Example

GET /v1/accounts/{:source_address}/notifications/{:id}
Response:
{
  "success": true,
  "notification": {
    "account": "rKeMKZyxpwiPdush32L4XjxxHpxdwSymN",
    "type": "payment",
    "direction": "incoming",
    "state": "validated",
    "result": "tesSUCCESS",
    "ledger": "9885467",
    "hash": "FD3394055A9165377C007103438A44518128666F8180D982F0A98A7B242397C9",
    "timestamp": "2014-11-11T21:46:10.000Z",
    "transaction_url": "http://<ripple-rest-server>/v1/accounts/rKeMKZyxpwiPdush32L4XjxxHpxdwSymN/payments/FD3394055A9165377C007103438A44518128666F8180D982F0A98A7B242397C9",
    "previous_hash": "29698F1C73A9D5F3821F6BA02A374F42C072364E1501D38AFB17F946E8B72EFD",
    "previous_notification_url": "http://<ripple-rest-server>/v1/accounts/rKeMKZyxpwiPdush32L4XjxxHpxdwSymN/notifications/29698F1C73A9D5F3821F6BA02A374F42C072364E1501D38AFB17F946E8B72EFD",
    "next_notification_url": ""
  }
}
Note:

- **next_notification_url**
  - Is empty implying this is the most recent transaction of the account.
  - Only mutable field for the same RESTful API call.
- **transaction_url**
  - Confirm the details of a payment - See example above.

Ripple REST API Reference:

https://ripple.com/build/ripple-rest/#check-notifications
https://ripple.com/build/ripple-rest/#confirm-payment
Appendix

Setting a Trustline on www.RippleTrade.com

Login to www.RippleTrade.com
Select the Fund tab:

Select Gateways
The button will be found at the bottom of the currencies list shown in the sidebar:
<table>
<thead>
<tr>
<th>Currency</th>
<th>Gateway</th>
<th>Balance</th>
</tr>
</thead>
<tbody>
<tr>
<td>BTC</td>
<td>~gateway-cold</td>
<td>78.0000</td>
</tr>
</tbody>
</table>

Learn more about gateways
Select **Connect gateway**
Enter the amount desired for the trustline:

After pressing save, the user will be prompted for a password:

The account password decrypts the users blob that contains the account secret. The account secret is used to sign the transaction that will set the trustline in the Ripple network.

**Press Submit**
The trustline will be created and displayed on the **Trust** screen:
<table>
<thead>
<tr>
<th>Gateway</th>
<th>Balance</th>
<th>Limit</th>
<th>Min</th>
<th>Rippling</th>
<th>Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>gateway-cold</td>
<td>78.0000</td>
<td>100.0000</td>
<td>0.0000</td>
<td>Off</td>
<td>Edit</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Gateway</th>
<th>Balance</th>
<th>Limit</th>
<th>Min</th>
<th>Rippling</th>
<th>Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>osbusiness</td>
<td>0.00</td>
<td>1,000,000,000.00</td>
<td>0.00</td>
<td>Off</td>
<td>Edit</td>
</tr>
</tbody>
</table>