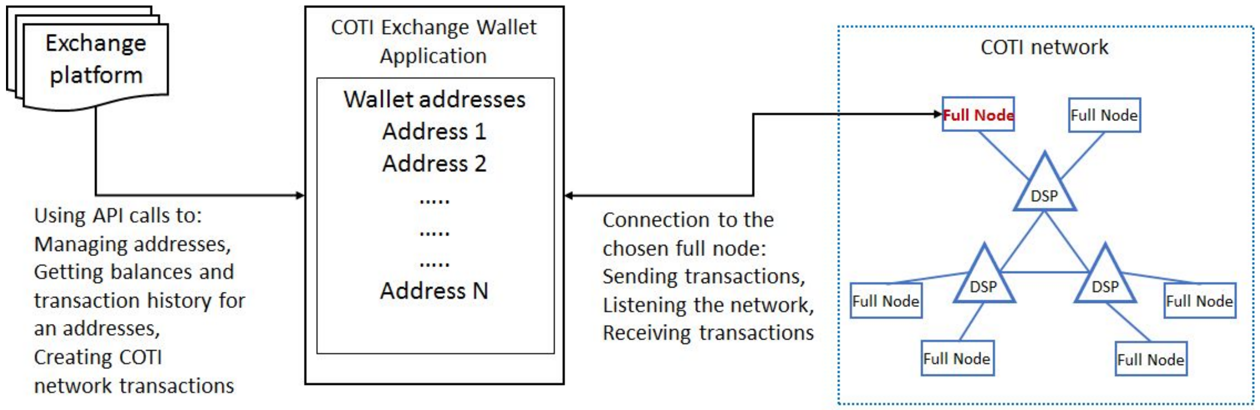
COTI - Exchange Token Integration

COTI Exchange Wallet Application

As part of the technical integration with any exchange as the platform and infrastructure is new and not supported by the exchange, it is requires us to prepare specialized wallet application for the deployment of the coin. The Exchange Wallet Application will in future support all tokens issued in our network intended to be listed on other exchange platforms.
The requirements for integration consists on the following:

- **COTI Exchange Wallet Application is a specialized version of COTI wallet, with advanced features allowing smooth integration with exchange clearing system.**
- **COTI Exchange Wallet Application allows an exchange to -**
    - Generate address - the wallet should allow easy access (API) for generating new addresses per exchange user.
    - Generate transaction - ability to send transaction from the exchange wallet to the user COTI wallet and vise versa.

    - Deposit - the exchange should have a coti wallet which can receive the transaction and can be monitor once transaction is approved.
    - Withdraw - send transaction from the exchange wallet to the user's wallet address.
    - Monitor the network in order to verify address balance and transaction status.
- **Running a full node is optional.** Running a full node allows to set a fixed fee for transaction sent from the exchange to other COTI wallet address. To fix fees is also possible by an agreement with an existing Full node or several Full nodes. Own Full node can be required also for security reasons, but we are sure that COTI network provides enough security regardless of which Full Node you are connected to.


Exchange platform
Using API calls to: Managing addresses, Getting balances and transaction history for an addresses, Creating COTI network transactions
COTI Exchange Wallet Application
Wallet addresses
Address 1
Address 2
.....
.....
.....
Address N
Connection to the chosen full node: Sending transactions, Listening the network, Receiving transactions
COTI network
Full Node
Full Node
DSP
DSP
DSP
Full Node
Full Node
Full Node
Full Node

Technical Requirements

COTI Exchange Wallet Application is a lightweight node JS app that can be run both on the exchange's own server or on a cloud server.

For minimal server configuration for COTI Full Node please refer to Full Node documentation.

COTI Exchange Wallet Specification

Counterparty Exchange Integration

By adding support for COTI token, your exchange will be able to connect to COTI exchange application and full node and send and monitor transactions performed on the network.

Technically, the process is rather straightforward. The below information is a guide for exchanges that wish to automate deposit and withdrawal of tokens using standard-conforming process on the COTI network. All COTI tokens, including COTI native token (COTI), conform to this standard. In this tutorial, the methods for depositing and withdrawing from the exchange account are shown.

We will also show the means by which activity can be monitored.

Basic Setup

There are many ways to architect an exchange. This guide uses the following design:

Exchange wallet: a collection of COTI network addresses belonging to the exchange used to deposit, hold and withdraw funds.

Wallet address: COTI network address generated (normally by the wallet application) for a client depositing funds or other purposes. COTI Exchange Wallet Application has no information which client deposited funds to which address, it is exchange clearing system functionality.

The main COTI Exchange Wallet Application functions are:
1) Managing wallet addresses.
2) Listening for deposit transactions from the COTI network.
3) Executing withdrawal transactions in the COTI network.

All calls to communicate with COTI Exchange Wallet Application are JSON REST API calls.

Handling Deposits using Separate Addresses

- Generate COTI wallet address for each user wanting to deposit COTI using the API of COTI Exchange Wallet Application.
- There are two possible options of keeping exchange assets in COTI: 1) keep on the deposit addresses, 2) create a holding address (or several holding addresses) and transfer funds to them.
- Depending on the chosen keeping option, withdrawal transactions may spend deposit addresses or holding addresses.
- Generation of a deposit address does not mean ipso facto that all the user’s tokens will be transferred to the address or user’s withdrawals will spend this address. It is up to exchange from which address or addresses withdraw tokens for a user.
- All calls to and from the exchange application will be IP protected.

Authentication & Security

Requests to the exchange app are protected by an api key (required) and whitelist ip filtering (optional) defined in the application configuration settings.

The following header is required on all requests:

```
Headers:
X-API-Key: “<the api key>”
```

For security purposes all requests to the exchange application are POST/PUT requests.

Generate Address Call

Through the COTI Exchange Wallet Application it is possible to generate a new address associated with a user or a certain transaction.

The POST query will be in the following format

```
POST generateNewAddress

{}
```

The response should return a coin wallet address for the user to deposit to.

```
{
  "addressHex":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
  "index”: 0,
  "status": "Success"
}
```

Get Address List

The GET query will be in the following format

```
GET /addresses
```

The response should return all addresses generated through the exchange app, ordered starting from index 0:

```
{
  “addresses”:[
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
"25f88a77e5f3d48eb18ec875c527f58e9c9121470278d7bc47e807c29268316823562eeb46b44f
06a19a9f2861d8e6800f45c6d138114671ee0aaa3cf21f079f9e91018e"
]
  "status": "Success"
}
```

Send Transaction Call

When a user requests a withdrawal from your exchange, they will need to provide you with their COTI address account and the amount to be withdrawn. You can then call the Send Transaction API to perform the withdrawal.

The send transaction call requires the following inputs:
- sourceAddresses (array) - Addresses from which to send the coins along with amounts
- feeAddress - the address to deduct the network fees from
- destinationAddress - the address to which the coins will be sent

Let's assume the customer with allocated address of f68967ea61d19cb9079f633f232f503d8114cdc906ad4a9470c56bfae1ad8954110187b82d663ed0db1014 3ca80dbea7595c8fc221f340c7f5f02f6f3e3fbffe613b2e91 would like to withdraw 50.0000 COTI’s.

The exchange has to define an address from which to deduct the fees. This address can be the customer’s address, or a designated exchange address for handling transaction fees.

```
PUT /transaction
{
  "sourceAddresses":[{“addressHex”:
"588125c9b4b9b4a83c3c8be557153129595f050f1a94b516e5958268dff36c76b3fb84448652ff
c808c3d73bb7f3087d9b93b6d51b368b5499d663703c78cc82fb329c08”,”amount”:100},{“add
ressHex”:”ba47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",”amount”
:50}],

"feeAddressHex":"588125c9b4b9b4a83c3c8be557153129595f050f1a94b516e5958268dff36c
76b3fb84448652ffc808c3d73bb7f3087d9b93b6d51b368b5499d663703c78cc82fb329c08",
  "destinationAddressHex":
"123fbf5acd4d96edf68b6fc4d385dad56b5ef6234d24d4203544ce1ebec38f0290fc6511c3863d
a26bd8f80ede5787cece797d7146efe8a3a617c8b24ee1df03007f525f",
 "description": "Withdrawal from the Exchange",
}
```

The response returns the transaction hash and status. A status of pending means the transaction was successfully attached to the DAG and is waiting for confirmation by the network.

```
{
  "transactionHash":
"bcd3efa5a0b25840f681fbf8e8197172bb3e6fd9f46445115399e1ca6eab0c45",
  “transactionStatus”:"Pending”,
  "Status": "Success",
}
```

At this stage your exchange application client accepted the transaction and likely broadcast it to the broader network.

By receiving the webhook updates you will be informed when your transaction is confirmed. In practice, we expect you will maintain private database state to track the transaction process. It can be useful to embed an exchange-specific transaction for the transactionrequest in order to map to your private database state. Another approach is to simply use the transaction ID for your mapping.

Transaction Status Call

The POST query will be in the following format

```
POST /transaction/status {
  "transactionHash":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6ba47ffdf
c41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407"
}
```

The response should return the transaction status:

```
{
  "transactionHash":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6ba47ffdf
c41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
  "transactionStatus": "Confirmed",
  “Status”: “Success”
}
```

```
Possible transactionStatus responses: Confirmed, Pending.

In case of error:

{  “status”: “Error”,
   “errorMessage”:”Transaction not valid”
  }
```

Get Address Balance Call

The POST query will be in the following format

```
POST /address/balance
{
  "addressHex":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407"
}
```

The response should return the address’s balance and preBalance (confirmed and unconfirmed balances)

```
{
  "addressHex":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
  "balance": "1544",
  "prebalance”:”1000",
  "status": "Success"
}
```

Get Total Wallet Balance

The POST query will be the following format

```
GET /balance
```

The response should return the wallet balance and preBalance (confirmed and unconfirmed balances)

```
{
  "balance": "350000",
  "prebalance”:”365000",
  "status": "Success"
}
```

Get Address Transactions Call

It is possible for the exchange to show the list of all transactions associated with a certain address in the following way:

The POST query will be the following format

```
POST /address/transactions
{
  "addressHex":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407"
 }
```

The response should return the list of

```
{
  "addressHex":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
  "transactions": [{"hash":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
             "createTime": 1557746085.610000000,
             ”type”:”sent”,
             "amount": "100",
              "transactionStatus":"Confirmed"},
            {"hash":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
             "createTime": 1557746092.857000000,
              ”type”:”received”,
             "amount": "500",
              "transactionStatus":"Confirmed"}]
"status": "Success"

}
```

Get Wallet transactions

It is possible for the exchange to request the list of all wallet transactions, along with an optional filter parameter:

The POST query will be in the following format

```
POST /transactions
{
  "filter":{
      updatedRecentSeconds: 120
      }
 }
```

The response should return the list of filtered transactions

```
{"status": "Success",
    "transactions": [
        {
            "addressHex":
"f68967ea61d19cb9079f633f232f503d8114cdc906ad4a9470c56bfae1ad8954110187b82d663e
d0db10143ca80dbea7595c8fc221f340c7f5f02f6f3e3fbffe613b2e91",
            "hash":
"a0d830f8f5cb64e07ec0f9a242aa924827a9d6728a4f96d5eb14b1f5245838c2",
            "amount": "100.10999",
            "type": "Sent",
            "transactionStatus": "Confirmed"
        },
        {
            "addressHex":
"63675fb4b6e136e6711f0e290d94ade9a5ebdd255155405471970f9d1927a7101dbba5c003ede6
ba637d0c73a046e6237d94a34575e646d8a22549b923e8abab2aba3fe8",
            "hash":
"a0d830f8f5cb64e07ec0f9a242aa924827a9d6728a4f96d5eb14b1f5245838c2",
            "amount": "100",
            "type": "Received",
            "transactionStatus": "Confirmed"
        },

   ]
}
```

Is Address Valid

To check if a given address is valid send a POST query in the following format

```
POST /address/valid
{
"address":f68967ea61d19cb9079f633f232f503d8114cdc906ad4a9470c56bfae1ad895411018
7b82d663ed0db10143ca80dbea7595c8fc221f340c7f5f02f6f3e3fbffe613b2e91
 }
```

The response returns isValid (a boolean flag) and validationDetails giving additional information.

Example of a valid address response:

```
{
    "status": "Success",
    "isValid": true,
    "validationDetails": "Address found"
}
```

Example of an invalid address response:

```
{
    "status": "Success",
    "isValid": false,
    "validationDetails": "Invalid address format"
}
```

Webhook notifications

The addresses generated through the exchange application wallet are automatically monitored for changes in the COTI Mainnet, allowing the exchange to track deposit and withdrawal statuses.

The following notification types are supported:

- Address balance updates
- Address transactions updates (any new transactions, transactions status changes)

Address balance update callback:

```
POST yourBalanceCallbackUrl
{
  "addressHex":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
  “balance”:1500,
  “prebalance”:1000
 }
```

Address transactions update callback:

```
POST yourTransactionCallbackUrl
{
  "addressHex":
"8e34b1e8221f20c8b79acf331e271c320053b05d14899ea3563a791c0124e7744b3b09183c7c6b
a47ffdfc41e68486601bb2662109687c9a355d5f5e68aff6cd7931a407",
  “transactions”:[
{"hash": "6fd702afe7eb9abd2676d2aad3e1d06268e8686ae93ff6adc9a70fcea2c7be1e",
             "createTime": 1557746092.857000000,

             “type”:”sent”,
             "amount": "500",
              transactionStatus":"Confirmed"},

{"hash": "6fd702afe7eb9abd2676d2aad3e1d06268e8686ae93ff6adc9a70fcea2c7be1e",
             "createTime": 1557746092.857000000,

             “type”:”received”,
             "amount": "1000",
              transactionStatus":"Pending"}
 ]
```

The webhook requests expect to receive a response with status code 200 (OK) and will otherwise retry the request every X minutes.

Best practices

- For deposits, wait for at least several minutes on the send to the deposit address and one confirmation for the send to the holding address.
- Keep the private key for the holding address secret and safe.
- Keep the bulk of your exchange’s funds in cold storage.