Command line tools for vault operators and Kintsugi users

by Kint0Sens

part of the VAULT INTERFACE bounty (https://github.com/interlay/bounties/issues/1)

License

The purpose of this project is to develop a suite of command line executables that will enable the user to interact with the Interbtc vaults. The current release for the AmsterDOT Hackathon contains the following command line tools:

- explore (list vault info - retrieve Liquidation Vault data)
- issue (combines request and btc payment)
- redeem
- transfer (KSM) to Relay Chain
- burn

This project is a fork of the interlay/interbtc-clients. All the modifications are licensed under the Apache License, Version 2.0. See the LICENSE_PREMIUM_BOT file.

Project Repository

https://github.com/Kint0Sens/interbtc-clients-cli The Premium BOT repository is a fork of the interlay/interbtc-clients. The PremiumBot code is available in the branch cli-1.12.0 (https://github.com/Kint0Sens/interbtc-clients-cli/tree/cli-1.12.0)

Video presentation

https://youtu.be/5Iw7rOLqCpQ

Overview

I started the current project as a way to introduce myself to rust programming and the substrate exosystem. When I reviewed the design of the interbtc-clients repository that generates the Interbtc vault program I realized that it is a rust client to the Interbtc Parachain. I started using the functions exposed in the rpc.rs file in the runtime of the package to make calls to the Interbtc parachain. I managed to build the explore command that way. I did the same to desing a basic issue and redeem. To then build the burn and transfer command I had to extend the rpc library with extra calls that have not been included in the original repository because they correspond to extrinsic calls that are not relevant to the vault operations.

For the issue command, my original intent was to produce a command line tool that would both request the issue to a vault and then immediately generate a btc transaction from a wallet. I intitially explored the useage of the bdk crate (bitcoindevkit). That project contains many nice features and does not force you to run a bitcoin core node to hold your wallet. I however stopped using bdk because I ran into issues when I started

running tests. Next option was to reverse engineer how the Interbtc team generates btc transactions to redeem bitcoins from the vault BTC account. By replicating those functions (that are in teh bitcoin crate of the interbtc-clients repository) I achieved my objective. The issue command line will generate or load a wallet similar to how the vault does when it is created. I can then send btcs to that wallet and then execute issues from it.

Explore command

The purpose of the Explore tool is to retrieve information about the vaults. Currently the Explore tool offers two subcommands.

- vaults
- liquidation-vault

The explore command is compatible with the recent introduction of KINT as collateral. See example execution below on Kintsugi where a KINT collateral vault is reported.

The **vaults** subcommand will list all vaults of the network and give the following infos

- status (Active, Liquidated, CommittedTheft),
- accepting issues/redeem
- banned until
- wrapped (current and pending) and Collateral balance
- collateralization (current and pending)
- issuable capacity

The **liquidation-vault** subcommand returns information on the burnable tokens that have been centralised in the "liquidation vault"[^1] when theft or undercollateralization have been identified on some vaults. It also gives an estimate of the reward gained per burnt KSM.

Burn command

The burn command will burn KBTC similar to what can be done in the UI. At the time of the AmsterDOT Hackathon release, there is some burnable amounts in Testnet and no burnable amounts in Kintsuki. So we test both a burn call with burnable amount present or not.

Transfer command

The transfer command issues a transfer call to the XToken palette of the Interbtc chain. It is configured to only transfer from the Interbtc parachain to the Relay Chain and only transfer the Relay Chain tokens. We did not address the transfer from Relay Chain to the Interbtc chain as in the current release we are only interacting with the Interbtc server. A transfer from the Relay Chain requires a call to the Realy Chain (Kusame or Polkadot).

The transfer command executes the transfer with the same sender/ receiver account by default unless a different destination address is provided

The command executes correctly in Testnet, but there seem to be some issue in the Kintsugi network that cause the connection to be closed when the request is sent.

Redeem command

The redeem command will redeem the requested amount from the specified vault. The current release does not allow for a random selection of the vault.

Issue command

The issue command will first request an issue to the specified vault. Next it will emit a BTC payment from a BTC wallet. The BTC wallet needs to reside on a bitcoin core node just like a Vault btc wallet does. See the section below on "interacting with the BTC wallet".

Technical details

Interbtc account information

For all commands that would need to access a Interbtc parachain account, you will need to provide the seed phrase in a file. By default the file read is keyfile.json in the current directory. The **keyfile** and **keyname** command line options let you change the default settings

-- keyfile

-- keyname

Unless specified in a commandline option, the programs look for a file named **keyfile.json** in the current directory. In the file the entry with the correct keyname is read. By default the keyname is **keyname**. You can override these defaults with a commandline option.

This file should contain the Kintsugi account of the BOT. You should follow the same syntax as the keyfile for vault accounts[^2].

Bitcoin Core RPC information

When using the issue command, you need to connect via RPC to a Bitcoin Core node. This process is identical to the one used by a Vault. In fact we reused the logic of the bitcoin crate of the interbtc-clients package to manage the BTC transcation of the issue command via rust.

As in the instruction to start a vault[^2] you need to specify the rpc url, user and password to connect to the bitcoin core node.

Example connection parameters

```
--bitcoin-rpc-url http://localhost:18332 \
--bitcoin-rpc-user useruseruser \
--bitcoin-rpc-pass passpasspass \
```

Interacting with the BTC wallet

Currently the issue command interacts with a wallet named "InterbtcCLIWallet-master". To manually load the wallet, check its balance or get a new btc address to transfer funds use the following commands

```
curl --data-binary '{"jsonrpc": "1.0", "id":"curltext", "method": "loadwallet",
"params": ["PremiumBotWallet-master"]}' -H 'content-type: text/plain;'
http://useruseruser:passpasspass@127.0.0.1:18332/
curl --data-binary '{"jsonrpc": "1.0", "id":"curltest", "method": "getbalance",
"params": ["*", 6] }' -H 'content-type: text/plain;'
http://useruseruser:passpasspass@127.0.0.1:18332/
curl --data-binary '{"jsonrpc": "1.0", "id":"curltest", "method": "getnewaddress",
"params": [] }' -H 'content-type: text/plain;'
http://useruseruser:passpasspass@127.0.0.1:18332/
```

Issues and possible next steps

- Resolve "connection closed" issue on Kintsugi **transfer** tests.
- **Issue** and **redeem** should select a vault automatically if no vault is specified
- Obfuscate the Bitcoin Core user and password in a file similar to keyfile.json for the Interbtc account details
- Add options to select the name of the BTC wallet on the RPC node in the **issue** command.
- Add to **redeem** Bitcoin Core BTC Wallet info to the redeem command (so as to generate)
- The **redeem** command does not check if the vault specified in the command options is valid or has aenough redeemable tokens. In case of error the Extrinsic error is reported by the program. Add some error management.

How to build

Please read the prerequisites provided in the README_interlay.md file about how to build the interbtc-clients repository. Building the Premium BOT repository uses exactly the same process. You only need to select the correct branch.

At the time of release of the project (20/6/2022) the main branch on Testnet and Kintsugi is 1.12.0 The corresponding Premium BOT branch is premium-bot-1.12.0 (https://github.com/Kint0Sens/interbtc-clients-cli/tree/cli-1.12.0).

In the root of the repository you will find 3 build scripts. They call cargo build --release with the appropriate feature (testnet/kintsugi or interlay) + specify a distinct target directory. For instance the build_kintsugi script is

```
cargo build --release  --features parachain-metadata-kintsugi --target-dir
./target/kintsugi
```

Example executions:

Explore command

Running "explore vaults" on Testnet and Kintsugi

```
> ./target/testnet/release/explore vaults
Connected to Interlay Testnet parachain
-------------------------------
Found 21 vaults. Current block 179173

-------------------------------
5H3ZbAmcyWfw45SKaHMiRFQ5fcXjBqstbHLs1a7eFtLH8vN5
   Active
   Collateral                 16.998000000000 KINT
   Tokens - Current           0.00010000 KBTC
   Tokens - Pending           0.00102050 KBTC
   Tokens - To be issued      0.00102000 KBTC
   Tokens - To be redeemed    0.00009950 KBTC
   Tokens - To be replaced    0.00000000 KBTC
   Collateralization - issued 3258.3400000000000000
   Collateralization - all    290.9232142857142857
   Issuable tokens:           0.00000000 KBTC

-------------------------------
5GBfrVtunvfHNo2R35UwSpUciBApSkFCvwpZ6gPznQeDKFC9
Liquidated

-------------------------------
5H6MkGXDz3f1BYQx3DcfjBs3i4cHixAKzh221aXd7V45ojeN
   Active
   Collateral                 334.331333333332 KINT
   Tokens - Current           0.00100000 KBTC
   Tokens - Pending           0.00000045 KBTC
   Tokens - To be issued      0.00000000 KBTC
   Tokens - To be redeemed    0.00099955 KBTC
   Tokens - To be replaced    0.00000000 KBTC
   Collateralization - issued 6408.7970000000000000
   Collateralization - all    6408.7970000000000000
   Issuable tokens:           0.01502199 KBTC

-------------------------------
...output interrupted...
```

```
> ./target/kintsugi/release/explore vaults
Connected to Kintsugi parachain
-------------------------------
Found 60 vaults. Current block 438045

-------------------------------
a3fudELrRCjuSyYEPkRAKFQyjzo5YyU228LdqinGsnjBUNB8P
   Inactive
   Collateral                 188.046805932678 KSM
   Tokens - Current           0.16203635 KBTC
   Tokens - Pending           0.16203635 KBTC
   Tokens - To be issued      0.00000000 KBTC
   Tokens - To be redeemed    0.00000000 KBTC
```

```
   Tokens - To be replaced    0.00000000 KBTC
   Collateralization - issued 289.6269201324270757
   Collateralization - all    289.6269201324270757
   Issuable tokens:           0.00000000 KBTC
   ------------------------------
a3bAsW2TQb1c7HrcZd2y81SdCVSGEA5pRkfg576G2EyxQupbw
   Inactive
   Collateral                 69.000000000000 KINT
   Tokens - Current           0.00130088 KBTC
   Tokens - Pending           0.00130088 KBTC
   Tokens - To be issued      0.00000000 KBTC
   Tokens - To be redeemed    0.00000000 KBTC
   Tokens - To be replaced    0.00000000 KBTC
   Collateralization - issued 1024.9115983026874115
   Collateralization - all    1024.9115983026874115
   Issuable tokens:           0.00000000 KBTC
```

Running "explore liquidation-vault" on Testnet and Kintsugi. For Kintsugi there are no burnable tokens.

```
> ./target/testnet/release/explore liquidation-vault
Connected to Interlay Testnet parachain
--------------------------------
Liquidation Vault [KSM/KBTC]
--------------------------------
   Collateral                     333.585543933016  KSM
   Burnable                             0.07508604  KBTC - (burnable = issued -
to_be_redeemed)
   Reward per burnt token                1752.4844  KSM  - (collateral /
(issued_tokens + to_be_issued_tokens))
--------------------------------
   to_be_issued_tokens                  0.11506010  KBTC
   issued_tokens                        0.07528999  KBTC
   to_be_redeemed_tokens                0.00020395  KBTC
```

```
> ./target/kintsugi/release/explore liquidation-vault
Connected to Kintsugi parachain
--------------------------------
Liquidation Vault [KSM/KBTC]
--------------------------------
   Collateral                     0.000000000000  KSM
   Burnable                          0.00000000  KBTC - (burnable = issued -
to_be_redeemed)
   Reward per burnt token                   NaN  KSM  - (collateral /
(issued_tokens + to_be_issued_tokens))
--------------------------------
   to_be_issued_tokens              0.00000000  KBTC
   issued_tokens                    0.00000000  KBTC
   to_be_redeemed_tokens            0.00000000  KBTC
```

Burn command

We will burn 2500 KBTC sat on Testnet

```
> ./target/testnet/release/burn
error: The following required arguments were not provided:
    --amount <AMOUNT>

USAGE:
    burn [OPTIONS] --amount <AMOUNT>

For more information try --help
> ./target/testnet/release/burn --amount 2500
[2022-06-18T12:58:42Z INFO  burn] Connected to Interlay Testnet parachain
[2022-06-18T12:58:42Z INFO  burn] ------------------------------
[2022-06-18T12:58:42Z INFO  burn] Signer:
5GTH76cE3vQyehe5w2Q9si4nxZ1izyvQzB6WezCSJGTKxwCU
[2022-06-18T12:58:42Z INFO  burn] Burn amount:        2500 KBTC sat
[2022-06-18T12:58:42Z INFO  burn] Wrapped balance:    7145368 KBTC sat
[2022-06-18T12:58:42Z INFO  burn] Collateral balance: 43095926098 KSM planck
[2022-06-18T12:58:42Z INFO  burn] Sending burn request to parachain
[2022-06-18T12:59:24Z INFO  burn] Burn of 2500 KBTC sat into KSM successful
[2022-06-18T12:59:24Z INFO  burn] Balance:            7142868 KBTC sat
[2022-06-18T12:59:24Z INFO  burn] Balance:            86955026252 KSM planck
```

Trying to burn 2500 KBTC sat on Kintsugi where burnable amount is 0 (see explore output above)

```
 ./target/kintsugi/release/burn \
--amount 2500 \
--keyfile ../kintsugi_test_keyfile/keyfile.json
[2022-06-18T13:05:29Z INFO  burn] Connected to Kintsugi parachain
[2022-06-18T13:05:29Z INFO  burn] ------------------------------
[2022-06-18T13:05:29Z INFO  burn] Signer:
5DkmTr3NeLPkQqefuU7QMJHJsjoso8vbDQbiTEFjCae2ecpK
[2022-06-18T13:05:29Z INFO  burn] Burn amount:        2500 KBTC sat
[2022-06-18T13:05:29Z INFO  burn] Wrapped balance:    21971 KBTC sat
[2022-06-18T13:05:29Z INFO  burn] Collateral balance: 0 KSM planck
[2022-06-18T13:05:29Z INFO  burn] Sending burn request to parachain
Error: RuntimeError(SubxtRuntimeError(Module(ModuleError { pallet:
"VaultRegistry", error: "InsufficientTokensCommitted", description: ["The
requested amount of tokens exceeds the amount available to this vault."],
error_data: ModuleErrorData { pallet_index: 20, error: [2, 0, 0, 0] } })))
```

Transfer command

Testnet transfer

```
> ./target/testnet/release/transfer --amount 2500
[2022-06-18T13:25:47Z INFO  transfer] Connected to Interlay Testnet parachain
[2022-06-18T13:25:47Z INFO  transfer] ------------------------------
[2022-06-18T13:25:47Z INFO  transfer] Signer:
5GTH76cE3vQyehe5w2Q9si4nxZ1izyvQzB6WezCSJGTKxwCU
[2022-06-18T13:25:47Z INFO  transfer] Transfer amount:  2500 KSM planck
[2022-06-18T13:25:47Z INFO  transfer] Balance:          86955026252 KSM planck
[2022-06-18T13:26:23Z INFO  transfer] Transfer of 2500 KSM sat to account  on
Relay Chain successful
[2022-06-18T13:26:23Z INFO  transfer] Balance:        86955023752 KSM planck
```

Kintsugi transfer

```
 ./target/kintsugi/release/transfer \
--amount 2500 \
--keyfile ../kintsugi_test_keyfile/keyfile.json \
--keyname keyname2
[2022-06-18T13:42:06Z INFO  transfer] Connected to Kintsugi parachain
[2022-06-18T13:42:06Z INFO  transfer] ------------------------------
[2022-06-18T13:42:06Z INFO  transfer] Signer:
5EKr3rfb2Pz2tZm4GtaYqguGK2mHkGBm5Eb3gshWsfvigp4u
[2022-06-18T13:42:06Z INFO  transfer] Transfer amount:  2500 KSM planck
[2022-06-18T13:42:06Z INFO  transfer] Balance:          449872000000 KSM planck
Error: RuntimeError(SubxtRuntimeError(Rpc(RestartNeeded("Networking or low-level
protocol error: WebSocket connection error: connection closed"))))
```

Issue command

```
> ./target/testnet/release/issue \
  --bitcoin-rpc-url http://localhost:18332 \
  --bitcoin-rpc-user useruseruser \
  --bitcoin-rpc-pass passpasspass \
  --amount 2500 \
  --vault 5H6MkGXDz3f1BYQx3DcfjBs3i4cHixAKzh221aXd7V45ojeN
[2022-06-18T14:56:54Z INFO  issue] Connected to Interlay Testnet parachain
[2022-06-18T14:56:54Z INFO  issue] ------------------------------
[2022-06-18T14:56:54Z INFO  issue] Connected to bitcoin Testnet network
[2022-06-18T14:56:54Z INFO  issue] ------------------------------
[2022-06-18T14:56:54Z INFO  issue] Signer:
5GTH76cE3vQyehe5w2Q9si4nxZ1izyvQzB6WezCSJGTKxwCU
[2022-06-18T14:56:54Z INFO  issue] Vault:
5H6MkGXDz3f1BYQx3DcfjBs3i4cHixAKzh221aXd7V45ojeN
[2022-06-18T14:56:54Z INFO  issue] Issue amount:     2500 KBTC sat
[2022-06-18T14:56:54Z INFO  issue] Issuable amount:  49651293 KBTC sat
[2022-06-18T14:56:54Z INFO  issue] BTC balance:      10000 BTC sat
[2022-06-18T14:56:54Z INFO  issue] Sending issue request to parachain
[2022-06-18T14:57:30Z INFO  issue] Issue request accepted
[2022-06-18T14:57:30Z INFO  issue] Issue BTC address:
```

```
tb1q4watk73an3kn5glgy7088nrrdy3lx2weda7ugm
[2022-06-18T14:57:30Z INFO  issue] Issue amount:     2496 KBTC sat
[2022-06-18T14:57:30Z INFO  issue] Issue fee:        4 KBTC sat
[2022-06-18T14:57:30Z INFO  issue] Building BTC transaction
[2022-06-18T14:57:30Z INFO  issue] Transaction sent. TxID:
41668bc0ec941a4dd5d28110627306e2218c4989759c3e9a29da9584bcee5b46
```

Redeem command

I ran the **redeem** command on the Kintsugi network. Testnet vaults were how of capacity at the time of writing.

```
> ./target/kintsugi/release/redeem \
  --btc-address bc1qstz4znvydj5mypqnu34je6x0umfugen039849s \
  --amount 12500 \
  --vault a3b5jAhU7CKnBKA3y92HkiCqRTbSKRMK5kN3RVPXcKoH4SDaC \
  --keyfile ../kintsugi_test_keyfile/keyfile.json
[2022-06-18T15:26:21Z INFO  redeem] Connected to Kintsugi parachain
[2022-06-18T15:26:21Z INFO  redeem] ------------------------------
[2022-06-18T15:26:21Z INFO  redeem] Signer:
5DkmTr3NeLPkQqefuU7QMJHJsjoso8vbDQbiTEFjCae2ecpK
[2022-06-18T15:26:21Z INFO  redeem] Vault:
5CwH4ELoC9Bjt8T1tsaXX1h6XngZWoBJT6JywPSz7LuWVq7J
[2022-06-18T15:26:21Z INFO  redeem] BTC Address
bc1qstz4znvydj5mypqnu34je6x0umfugen039849s
[2022-06-18T15:26:21Z INFO  redeem] Redeem amount:  12500 KBTC sat
[2022-06-18T15:26:21Z INFO  redeem] Balance:        21971 KBTC sat
[2022-06-18T15:27:08Z INFO  redeem] Vault
5CwH4ELoC9Bjt8T1tsaXX1h6XngZWoBJT6JywPSz7LuWVq7J confirmed redeem request of 12500
KBTC sat to BTC address bc1qstz4znvydj5mypqnu34je6x0umfugen039849s
```

[^2]: See https://docs.interlay.io/#/vault/installation?id=prerequisites