Ark v0.6.x

Nigiri環境 Ark v0.6.x#1e25a3

非Nigiri環境 Ark v0.6.x#75fd5e

https://github.com/arkade-os/arkd/tree/v0.6.3

2025/07/17

最初はArk v0.7.xを試していたが、動かないのでTelegramに質問したところリリースされたばかりでどうのこうの。

`ark-cli` での受信ができなかったがドキュメントもなくよくわからないものの、一応 v0.6.3 なら動いた。

Nigiri環境

データディレクトリは `~/.nigiri` になる。

やり直したい場合は Nigiri が止まっている間に削除すると良い。

install

```
$ curl https://getnigiri.vulpem.com | bash

$ nigiri --help
NAME:
   nigiri CLI - one-click bitcoin development environment

USAGE:
   nigiri [global options] command [command options] [arguments...]

VERSION:

Version: 0.5.10
Commit: 27bf34c062257a7f283412d02aec098fbf2cc0c9
Date: 2025-05-28T14:03:37Z

COMMANDS:
   start    start nigiri
   stop     stop nigiri
```

```
   logs     check Service logs
   update   check for updates and pull new docker images
   version
   forget   forget previously remembered flags for the start command
   rpc      invoke bitcoin-cli or elements-cli
   faucet   generate and send bitcoin to given address
   mint     liquid only: issue and send a given quantity of an asset
   push     broadcast raw transaction
   ark      invoke ark client commands
   arkd     invoke arkd client commands
   cln      invoke Core Lightning command line
   lnd      invoke LND command line
   tap      invoke tap command line interface
   help, h  Shows a list of commands or help for one command

GLOBAL OPTIONS:
   --datadir value  use different data directory (default: "/home/ueno/.nigiri")
   --ark            enable Ark Network (default: false)
   --help, -h       show help (default: false)
   --version, -v    print the version (default: false)

$ nigiri arkd --version
Arkd CLI version v0.6.2
```

arkありスタート

```
$ nigiri start --ark
......(略)
 ENDPOINTS
✓ chopsticks: localhost:3000
✓ ark: localhost:7070
✓ electrs: localhost:50000
✓ esplora: localhost:5000
✓ bitcoin: localhost:18443
```

新規であれば Bitcoin ブロックが 101 生成された状態になっている。

nigiri arkの使い方

`arkd` はウォレット作成してunlockすると gRPC/REST APIが動くようになる。

wallet

```
$ nigiri arkd wallet create --password aaaaaaaa
```

```
(復元用のニモニックが出力される)
$ nigiri arkd wallet unlock --password aaaaaaaa
wallet unlocked
$ nigiri arkd wallet status
initialized: true
unlocked: true
synced: true
```

`ark` が参加する際、 `arkd` が同額以上のBTCを持っていないと成功しない。

`nigiri faucet` で多めに入金しておく。このコマンドは実施後に 1ブロック生成するようである(generate and sendとはあるが)。

fund

```
$ nigiri arkd wallet address
bcrt1qm7dzq5vrwrcaf2a94p2yk9hskxr63sta8w9mw5
$ nigiri faucet bcrt1qd8pxcfr9gvjm23ck2f4nnlrz5kz5gnyuanxeqs 10
txId: e735556ec4cd3d85b9ceb32369b2f8576b3c960c4e1258e413654f80896f6102
$ nigiri arkd wallet balance
main account
   available: 10.00000000
   locked: 0.00000000
connectors account
   available: 0.00000000
   locked: 0.00000000
```

`ark` コマンドも `nigiri` に入っている。ただデータフォルダのARK_WALLET_DATADIRは使えなさそうだった。

ark

```
$ nigiri ark init --network regtest --password bbbbbbbb --server-url localhost:7070
--explorer http://chopsticks:3000
$ nigiri ark balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 0,
                "details": []
        }
}
```

boarding_addressへ入金する。

ここからがよくわかっていないのだが、入金して 1conf の状況で `settle` コマンドを使うと boarding が完了するように見える。

send BTC

```
$ nigiri ark receive
{
        "boarding_address":
"bcrt1pqltez3muwh5kt0caefxn75qnwu58uy9czxwtlrflmzm8yaphtvwsff2dxp",
        "offchain_address":
"tark1q4t27tpts6j0jf948kn9legaugdu89yfqhsmnj45cte9frqcff3kgycv2eesqyzmy5uq5vqdznc78
e37f6ukcfuqldpgjy4lnhfj0ysw57vuk"
}

$ nigiri faucet bcrt1pqltez3muwh5kt0caefxn75qnwu58uy9czxwtlrflmzm8yaphtvwsff2dxp 1
txId: 9372a44aad0c7315f0c349e360e8f50201a09e8ae1a4c3732c5a1e2d8644b8c3

$ nigiri ark settle --password bbbbbbbb
time="2025-07-17T08:15:34Z" level=info msg="registered inputs and outputs with
request id: 1903ca4e-a76e-413f-b11a-34179fb38f79"
time="2025-07-17T08:15:36Z" level=info msg="a round signing started"
time="2025-07-17T08:15:36Z" level=info msg="round combined nonces generated"
time="2025-07-17T08:15:36Z" level=info msg="a round finalization started"
time="2025-07-17T08:15:36Z" level=info msg="submitting forfeit transactions... "
time="2025-07-17T08:15:36Z" level=info msg=done.
time="2025-07-17T08:15:36Z" level=info msg="waiting for round finalization..."
time="2025-07-17T08:15:36Z" level=info msg="round completed
5780fc75ea92655580e97f878f0613e460085a388d61d1adfc048e3d31c74ef6"
{
        "txid": "5780fc75ea92655580e97f878f0613e460085a388d61d1adfc048e3d31c74ef6"
}

$ nigiri ark balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 100000000,
```

```
                "next_expiration": "2025-07-24T08:13:28Z",
                "details": [
                        {
                                "expiry_time": "2025-07-24T08:13:28Z",
                                "amount": 100000000
                        }
                ]
        }
}
```


これで offchain_balance に残高が出てきた。

受信先のarkを用意する。

`ark` バイナリはv0.6.3のarkをダウンロードして使った。

環境変数 `ARK_WALLET_DATADIR` を設定した。 `--explorer` は Nigiri の Docker network ではなく `localhost` にする。

ark2


```
$ export ARK_WALLET_DATADIR=cli
$ ./ark init --network regtest --password cccccccc --server-url localhost:7070 --explorer http://localhost:3000
$ ./ark balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 0,
                "details": []
        }
}
$ ./ark receive
{
        "boarding_address": "bcrt1pmmyccwlzahrygmkflnx2095p78e480yn2l437z8dnx0jcljec8fqnuw7gk",
        "offchain_address": "tark1q4t27tpts6j0jf948kn9legaugdu89yfqhsmnj45cte9frqcff3a74sng0kx3w6mypqw0mdnzdahuqgpadqa083ytf0nl7cp48p34yc9jf8lz"
}
```


`nigiri ark` から送信。

10,000 sats の送信で 10,204 sats 減っている。手数料は 204 satsか。

send

```
$ nigiri ark send --to
tark1q4t27tpts6j0jf948kn9legaugdu89yfqhsmnj45cte9frqcff3a74sng0kx3w6mypqw0mdnzdahuq
gpadqa083ytf0nl7cp48p34yc9jf8lz --amount 10000 --password bbbbbbbb
WARN: failed to parse the redeem tx, returning the full psbt
{
        "redeem_tx":
"92aae12bd0455251d430c62d3f3066f80f13707a17e683f6ee58caf6b366e330"
}

$ nigiri ark balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 99989796,
                "next_expiration": "2025-07-24T08:13:28Z",
                "details": [
                        {
                                "expiry_time": "2025-07-24T08:13:28Z",
                                "amount": 99989796
                        }
                ]
        }
}
```

`./ark balance` で確認。

`expiry_time` は `nigiri ark` 側と同じになっている。

balance

```
$ ./ark balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 10000,
                "next_expiration": "2025-07-24T17:13:28+09:00",
```

```
                "details": [
                        {
                                "expiry_time": "2025-07-24T17:13:28+09:00",
                                "amount": 10000
                        }
                ]
        }
}
```

今度は `./ark` から `nigiri ark` に送信。

500 sats の送金で 704 sats 減っている。こちらも手数料が 204 satsということか。

確認できないが Ark transaction のサイズだろうか？

send to nigiri

```
$ ./ark send --to
tark1q4t27tpts6j0jf948kn9legaugdu89yfqhsmnj45cte9frqcff3kgycv2eesqyzmy5uq5vqdznc78e
37f6ukcfuqldpgjy4lnhfj0ysw57vuk --amount 500 --password cccccccc
WARN: failed to parse the redeem tx, returning the full psbt
{
        "redeem_tx":
"83f9c2f1972b22d9abaecd50afee2035d852fcabc76e1287eeb8780ae5fc0276"
}
$ ./ark balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 9296,
                "next_expiration": "2025-07-24T17:13:28+09:00",
                "details": [
                        {
                                "expiry_time": "2025-07-24T17:13:28+09:00",
                                "amount": 9296
                        }
                ]
        }
}
```

`ark` 自体にはそこまでコマンドが実装されていないので、何かやりたかったらAPIを使うこと

になる。

例えばListVtxosコマンドだとこうなる。これは spendable も spent も出力しているので、指定したアドレスに関係する VTXOなのだろう。

ListVtxos

```
$ curl http://localhost:7070/v1/vtxos/
tark1q4t27tpts6j0jf948kn9legaugdu89yfqhsmnj45cte9frqcff3a74sng0kx3w6mypqw0mdnzdahuq
gpadqa083ytf0nl7cp48p34yc9jf8lz
```

GetVtxosはクエリーで出力をいじることができる。

GetVtxos

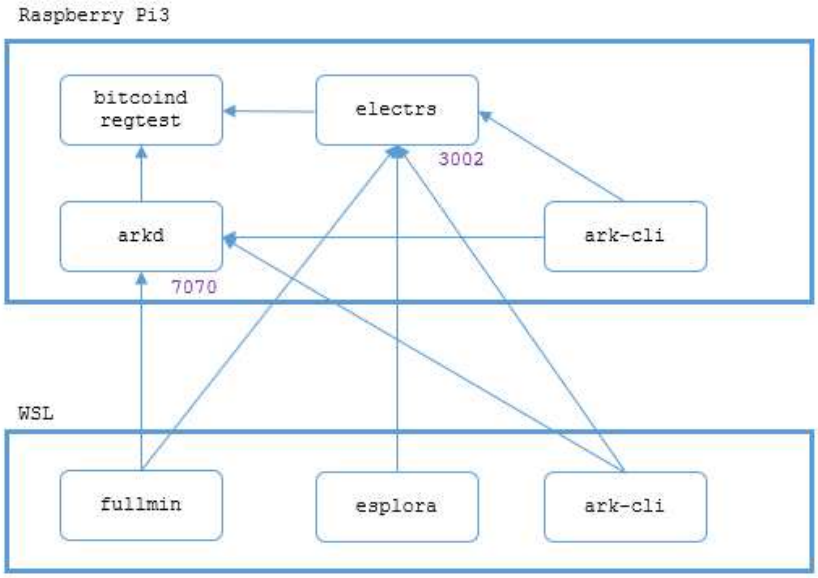
```
$ curl http://localhost:7070/v1/getVtxos/
tark1q4t27tpts6j0jf948kn9legaugdu89yfqhsmnj45cte9frqcff3a74sng0kx3w6mypqw0mdnzdahuq
gpadqa083ytf0nl7cp48p34yc9jf8lz
{"vtxos":[{"outpoint":
{"txid":"83f9c2f1972b22d9abaecd50afee2035d852fcabc76e1287eeb8780ae5fc0276","vout":1
},"createdAt":"1752741193","expiresAt":"1753344808","amount":"9296","script":"df561
343ec68bb5b2040e7edb3137b7e0101eb41d79e245a5f3ffb01a9c31a93","isLeaf":false,"isSwep
t":false,"isSpent":false,"spentBy":"","commitmentTxid":"5780fc75ea92655580e97f878f0
613e460085a388d61d1adfc048e3d31c74ef6"},{"outpoint":
{"txid":"92aae12bd0455251d430c62d3f3066f80f13707a17e683f6ee58caf6b366e330","vout":0
},"createdAt":"1752740464","expiresAt":"1753344808","amount":"10000","script":"df56
1343ec68bb5b2040e7edb3137b7e0101eb41d79e245a5f3ffb01a9c31a93","isLeaf":false,"isSwe
pt":false,"isSpent":true,"spentBy":"83f9c2f1972b22d9abaecd50afee2035d852fcabc76e128
7eeb8780ae5fc0276","commitmentTxid":"5780fc75ea92655580e97f878f0613e460085a388d61d1
adfc048e3d31c74ef6"}],"page":null}

$ curl http://localhost:7070/v1/getVtxos/
tark1q4t27tpts6j0jf948kn9legaugdu89yfqhsmnj45cte9frqcff3a74sng0kx3w6mypqw0mdnzdahuq
gpadqa083ytf0nl7cp48p34yc9jf8lz?spendableOnly=true
{"vtxos":[{"outpoint":
{"txid":"83f9c2f1972b22d9abaecd50afee2035d852fcabc76e1287eeb8780ae5fc0276","vout":1
},"createdAt":"1752741193","expiresAt":"1753344808","amount":"9296","script":"df561
343ec68bb5b2040e7edb3137b7e0101eb41d79e245a5f3ffb01a9c31a93","isLeaf":false,"isSwep
t":false,"isSpent":false,"spentBy":"","commitmentTxid":"5780fc75ea92655580e97f878f0
613e460085a388d61d1adfc048e3d31c74ef6"}],"page":null}
```

非Nigiri環境

全部自分で立てる場合。

Nigiri は dockerコンテナなのでパラメータを変化させたいときのやり方がよくわからない。全部自分で立てた方がやりやすいと思った。



bitcoind

~/.bitcoin/bitcoin.conf

```
server=1
txindex=1
regtest=1

zmqpubrawblock=tcp://127.0.0.1:28332
zmqpubrawtx=tcp://127.0.0.1:28333

[regtest]
rpcuser=testuser
rpcpassword=testpass
fallbackfee=0.000001

# SPV
blockfilterindex=1
peerblockfilters=1
```

起動bitcoind

```
bitcoind -daemon
```

ウォレットの作成や最初の状態までブロック生成したりする手順をスクリプトにしておくと楽だろう。

renew-bitcoind.sh

```
#!/bin/bash

bitcoin-cli stop
rm -rf ~/.bitcoin/regtest
bitcoind -daemon

while :
do
        dl=`bitcoin-cli getblockchaininfo | jq -r .initialblockdownload`
        if [ "$dl" == "false" ]; then
                break
        fi
        sleep 1
done

bitcoin-cli createwallet ""
./generate.sh 101
```

generate.sh はブロック生成用。"auto"で自動実行する。30秒にしている。短いと arkd 側でエラーが出るかもしれないが、正直なところ関連性は不明である。

generate.sh

```
#!/bin/bash

PERIOD=30
NUM=$1
if [ "$NUM" == "" ]; then
        NUM=1
elif [ "$NUM" == "auto" ]; then
        while :
        do
                date
                bitcoin-cli generatetoaddress 1 `bitcoin-cli getnewaddress`
                sleep $PERIOD
        done
fi
bitcoin-cli generatetoaddress $NUM `bitcoin-cli getnewaddress`
```

electrs

Installing & indexing

build

```
$ git clone https://github.com/blockstream/electrs && cd electrs
$ git checkout new-index
$ cargo build --release --bin electrs

# 適当に実行ファイルをコピー
$ cp target/release/electrs ~/.local/bin/
```

`--cors` はesploraから呼び出すために設定している。

start-electrs.sh

```
#!/bin/bash

DATADIR="./electrs-data"
NETWORK="regtest"
RPCUSER="testuser"
RPCPASS="testpass"
ELECTRUM_URL="localhost:50001"
REST_URL="localhost:3002"
ESPLORA_URL="*"

electrs --db-dir="$DATADIR" \
    --network="$NETWORK" \
    --cookie="${RPCUSER}:${RPCPASS}"  \
    --electrum-rpc-addr="$ELECTRUM_URL" \
    --http-addr="$REST_URL" \
    --cors="$ESPLORA_URL"
```

esplora

必須ではない。トランザクションの確認に便利である。

Developing

開発用の場合は localhost でしかちゃんと動かない。

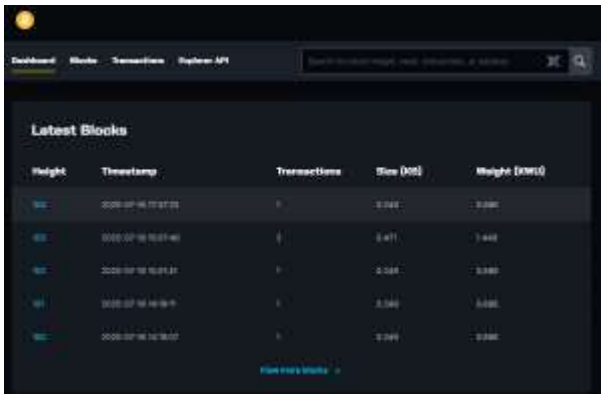
`STATIC_ROOT` はブラウザでアクセスするときのURL、 `API_URL` はelectrsの `REST_URL` である。

起動esplora

```
$ git clone https://github.com/Blockstream/esplora && cd esplora
```

```
$ npm install
$ export STATIC_ROOT=http://localhost:5000/
$ export API_URL=http://localhost:3002/
$ npm run dev-server
```

ブラウザから http://localhost:5000/ にアクセスすると見える。エラーになる場合はelectrsの `--cors` 設定を疑うとよいかもしれない(ブラウザの開発モードにするとわかりやすい)。



arkd

バイナリファイルはhttps://github.com/arkade-os/arkd/releases/tag/v0.6.3からダウンロードした。

`arkd` はdaemonでもあり、daemon CLIでもある。

スクリプトを用意した。

`ARK_VTXO_TREE_EXPIRY` などは秒数なのだが 512の倍数という縛りがある。

デフォルト値は `604672` で7日間となっているが、 $int(7 * 24 * 60 * 60/512) * 512$ のような感じだ。

`ARK_ALLOW_ZERO_FEES=true` にしている。設定の有無で何の手数料が変わるのかは分からない。

arkd.sh

```
#!/bin/bash

ARKD=./bin/arkd
PASSWD=aaaaaaaa

export ARK_PORT="7070"
export ARK_DATADIR="./data/arkd"
export ARK_LOG_LEVEL="5"

# 512x
export ARK_VTXO_TREE_EXPIRY="512"
```

```
export ARK_UNILATERAL_EXIT_DELAY="512"
export ARK_BOARDING_EXIT_DELAY="1024"

export ARK_ROUND_INTERVAL="15"
export ARK_NO_TLS="true"
export ARK_NO_MACAROONS="true"

export ARK_EVENT_DB_TYPE="badger"
export ARK_DB_TYPE="sqlite"
export ARK_SCHEDULER_TYPE="gocron"
export ARK_TX_BUILDER_TYPE="covenantless"
export ARK_UNLOCKER_TYPE="env"
export ARK_UNLOCKER_PASSWORD="$PASSWD"
export ARK_ALLOW_ZERO_FEES="true"

# UTXO
export ARK_UTXO_MAX_AMOUNT=-1
export ARK_UTXO_MIN_AMOUNT=-1
# VTXO
export ARK_VTXO_MAX_AMOUNT=-1
export ARK_VTXO_MIN_AMOUNT=1

export ARK_NETWORK="regtest"
# export ARK_NEUTRINO_PEER="localhost:18444"
# export ARK_ESPLORA_URL="http://localhost:3002"
export ARK_BITCOIND_RPC_USER="testuser"
export ARK_BITCOIND_RPC_PASS="testpass"
export ARK_BITCOIND_RPC_HOST="localhost:18443"
export ARK_BITCOIND_ZMQ_BLOCK="tcp://localhost:28332"
export ARK_BITCOIND_ZMQ_TX="tcp://localhost:28333"

if [ "$1" == "init" ]; then
        $ARKD wallet create --password $PASSWD
        $ARKD wallet unlock --password $PASSWD
        sleep 10
        killall ./bin/arkd
elif [ "$1" == "unlock" ]; then
        $ARKD wallet unlock --password $PASSWD
else
```

```
        $ARKD $@
    fi
```

初回

引数無しで実行すると常駐する。

```
$ ./arkd.sh
```

ウォレット作成

ウォレットがないと何も動かない(APIも呼び出せない)。

```
$ ./arkd.sh init
```

`ARK_DATADIR` にデータディレクトリが作られる。

(なお、バージョンv0.7.0から環境変数の名前が変わる)

`ARK_UNLOCKER_PASSWORD` を設定しているので自動的にunlockされるかもしれない。

少なくとも一度データディレクトリを作成した後は `./arkd.sh` で起動するとunlockされていた。

理由はよくわからないが、ウォレット作成したままの `arkd` でクライアントへの入金作業を進めると、 `arkd` のウォレットがブロックの追跡に失敗するようだった。

一度正起動すればよさそうだったので `init` 後は `killall` で終了させるようにしている。10秒スリープさせているのは、多少バックグラウンドで何かしているようだったからである。別の待ち方の方が格好良いだろう。

例

```
    $ ./arkd.sh wallet address
    $ bitcoin-cli sendtoaddress <アドレス> 10
    $ ~/.generate.sh
    $ ./arkd.sh wallet balance
```

以降

それ以降は Nigiri 環境を使った場合と同じである。

ユーザ側の `ark` を操作するためのスクリプトを用意した。

`ark send` の代わりに `pay` を用意した。

1.sh

```
    #!/bin/bash
```

```
export ARK_WALLET_DATADIR=./data/cli1
export PASSWD=bbbbbbbb

./cli.sh $@
```

2.sh

```
#!/bin/bash

export ARK_WALLET_DATADIR=./data/cli2
export PASSWD=cccccccc

./cli.sh $@
```

cli.sh

```
#!/bin/bash

ARK="./bin/ark"

CLI="$ARK"
ASP_SERVER="localhost:7070"
HTTP_API="http://localhost:3002"

if [ "$1" == "init" ]; then
       $CLI init --network regtest --password $PASSWD --server-url $ASP_SERVER --
explorer $HTTP_API
elif [ "$1" == "settle" ]; then
       $CLI settle --password $PASSWD
elif [ "$1" == "pay" ]; then
       $CLI send --to $2 --amount $3 --password $PASSWD
else
       $CLI $@
fi
```

VTXOのタイムアウトはスクリプトにLockTimeを見ているので、ブロック生成は定期的にさせておいた方が期待した動作になる。

例

```
$ ./1.sh init
$ ./1.sh receive
$ bitcoin-cli sendtoaddress <onchainアドレス> 1
```

```
$ ~/generate.sh
$ ./1.sh settle
$ ./1.sh balance
```

TX `fb14...` はsettle時のトランザクションで、INPUTはオペレータとユーザがreceiveアドレスに送金したものである。
ユーザは1BTCのVTXOを作っている。オペレーターの残高が9BTCくらいになったけど、同じ額が必要とされる？

settle時のログ

```
[2025-07-19 16:20:31.252] DEBU[3386] building tx for round f2260a43-da4a-48c0-8046-dc41c243f4cf
[2025-07-19 16:20:31.334] DEBU[3386] round tx created for round f2260a43-da4a-48c0-8046-dc41c243f4cf
[2025-07-19 16:20:31.334] DEBU[3386] signing session created for round f2260a43-da4a-48c0-8046-dc41c243f4cf with 1 signers
[2025-07-19 16:20:31.335] DEBU[3386] forwarding event to 1 listeners
[2025-07-19 16:20:31.342] DEBU[3386] gRPC method: /ark.v1.ArkService/SubmitTreeNonces
[2025-07-19 16:20:31.343] DEBU[3386] nonces collected for round f2260a43-da4a-48c0-8046-dc41c243f4cf
[2025-07-19 16:20:31.344] DEBU[3386] nonces aggregated for round f2260a43-da4a-48c0-8046-dc41c243f4cf
[2025-07-19 16:20:31.346] DEBU[3386] forwarding event to 1 listeners
[2025-07-19 16:20:31.351] DEBU[3386] tree signed by us for round f2260a43-da4a-48c0-8046-dc41c243f4cf
[2025-07-19 16:20:31.352] DEBU[3386] waiting for cosigners to sign the tree
[2025-07-19 16:20:31.357] DEBU[3386] gRPC method: /ark.v1.ArkService/SubmitTreeSignatures
[2025-07-19 16:20:31.358] DEBU[3386] signatures collected for round f2260a43-da4a-48c0-8046-dc41c243f4cf
[2025-07-19 16:20:31.366] DEBU[3386] vtxo tree signed for round f2260a43-da4a-48c0-8046-dc41c243f4cf
[2025-07-19 16:20:31.366] DEBU[3386] started finalization stage for round: f2260a43-da4a-48c0-8046-dc41c243f4cf
[2025-07-19 16:20:31.368] DEBU[3386] forwarding event to 1 listeners
[2025-07-19 16:20:31.378] DEBU[3386] gRPC method: /ark.v1.ArkService/SubmitSignedForfeitTxs
[2025-07-19 16:20:31.422] DEBU[3386] all forfeit txs and boarding inputs signatures
```

```
have been sent
[2025-07-19 16:20:31.423] DEBU[3386] signing transaction f2260a43-da4a-48c0-8046-
dc41c243f4cf
[2025-07-19 16:20:31.436] DEBU[3386] finalized round f2260a43-da4a-48c0-8046-
dc41c243f4cf with round tx
fb1411f612a814506bec5c3df3c9c6631036525714d59053f3c8a9035d25b652
[2025-07-19 16:20:31.437] DEBU[3386] forwarding event to 0 listeners
[2025-07-19 16:20:31.438] DEBU[3386] forwarding event to 1 listeners
[2025-07-19 16:20:31.481] DEBU[3387] started registration stage for new round:
b78d8ce1-147e-44f7-8d93-0afa18ea5d15
[2025-07-19 16:20:31.523] DEBU[3387] added 1 new vtxos
[2025-07-19 16:20:31.523] DEBU[3387] scheduled sweep for round
fb1411f612a814506bec5c3df3c9c6631036525714d59053f3c8a9035d25b652 at 2025-07-19
16:29:02
[2025-07-19 16:20:31.523] DEBU[3387] started watching 1 vtxos
```

`generate.sh auto` でブロックを作らせているとVTXOタイムアウト時には sweeperが動作していた。

`5ab8...` はエクスプローラで見えなかったのでVTXOか。 `9bff...` はsettleしたトランザクションをINPUTにしている。

VTXOタイムアウト時のarkdログ

```
[2025-07-19 16:29:32.580] DEBU[3928] sweeper:
5ab83fdeac5449eb738eddcd3d07c91f2a65c567049eb41f0393131b8264c8c1
[2025-07-19 16:29:32.669] DEBU[3928] sweep tx broadcasted:
9bffdf94df1ad1d6bc0c509d3573747d89f55df572edb647a046272a10cbe8a3
[2025-07-19 16:29:32.713] DEBU[3928] 1 vtxos swept
```



この時点でユーザの残高はゼロである。しかし、一度も使っていないので取り戻せるはずだ。

ユーザ残高

```
$ ./1.sh balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 0,
```

```
                "details": []
        }
}
```

リカバー

```
$ ./1.sh recover --password bbbbbbbb
INFO[0000] registered inputs and outputs with request id:
8c8566e9-0df2-44d8-85e2-3cc71f697f37
INFO[0000] a round signing started
INFO[0000] round combined nonces generated
INFO[0000] a round finalization started
INFO[0000] submitting forfeit transactions...
INFO[0000] done.
INFO[0000] waiting for round finalization...
INFO[0000] round completed
05b221ee27cbe088798fc6869a22d63227352d14fb2bd4dcdf418191c5988839
{
        "txid": "05b221ee27cbe088798fc6869a22d63227352d14fb2bd4dcdf418191c5988839"
}

$ ./1.sh balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 100000000,
                "next_expiration": "7 minutes",
                "details": [
                        {
                                "expiry_time": "2025-07-19T17:06:37+09:00",
                                "amount": 100000000
                        }
                ]
        }
}
```

省略しているが、RECOVER時のVOUT#0はさらに送金してSETTLE時と同じようなトランザクションを展開している。

`arkd` のログにアドレスが出てこないので、どれが誰のものかよくわからない。ただ、この時

点で `arkd wallet balance` は9BTCくらいに減っている。時間がかかるだけか？



`arkd` 側がお金を払っている。ユーザ側の残高が変わっていないのは `ARK_ALLOW_ZERO_FEES` だからか？ これを無駄に繰り返すだけでオペレータはかなり損害を受けるだろうから、何か考えてはあるだろう。

`arkd`

```
[2025-07-19 16:58:04.981] DEBU[5640] gRPC method: /ark.v1.ExplorerService/ListVtxos
[2025-07-19 16:58:05.166] DEBU[5640] gRPC method: /ark.v1.ArkService/RegisterIntent
[2025-07-19 16:58:05.172] DEBU[5640] gRPC method: /ark.v1.ExplorerService/GetRound
[2025-07-19 16:58:05.176] DEBU[5640] gRPC method: /ark.v1.ArkService/GetEventStream
[2025-07-19 16:58:05.177] DEBU[5640] gRPC method: /ark.v1.ArkService/Ping
[2025-07-19 16:58:05.464] DEBU[5640] started finalization stage for round:
f8f94f3b-28c2-4dff-b8e9-946cb6d3404e
[2025-07-19 16:58:05.468] DEBU[5640] building tx for round f8f94f3b-28c2-4dff-
b8e9-946cb6d3404e
[2025-07-19 16:58:05.507] DEBU[5641] round tx created for round f8f94f3b-28c2-4dff-
b8e9-946cb6d3404e
[2025-07-19 16:58:05.508] DEBU[5641] signing session created for round
f8f94f3b-28c2-4dff-b8e9-946cb6d3404e with 1 signers
[2025-07-19 16:58:05.510] DEBU[5641] forwarding event to 1 listeners
[2025-07-19 16:58:05.514] DEBU[5641] gRPC method: /ark.v1.ArkService/
SubmitTreeNonces
[2025-07-19 16:58:05.516] DEBU[5641] nonces collected for round f8f94f3b-28c2-4dff-
b8e9-946cb6d3404e
[2025-07-19 16:58:05.516] DEBU[5641] nonces aggregated for round
f8f94f3b-28c2-4dff-b8e9-946cb6d3404e
[2025-07-19 16:58:05.517] DEBU[5641] forwarding event to 1 listeners
[2025-07-19 16:58:05.524] DEBU[5641] tree signed by us for round
f8f94f3b-28c2-4dff-b8e9-946cb6d3404e
[2025-07-19 16:58:05.525] DEBU[5641] waiting for cosigners to sign the tree
[2025-07-19 16:58:05.526] DEBU[5641] gRPC method: /ark.v1.ArkService/
```

```
SubmitTreeSignatures
[2025-07-19 16:58:05.527] DEBU[5641] signatures collected for round
f8f94f3b-28c2-4dff-b8e9-946cb6d3404e
[2025-07-19 16:58:05.536] DEBU[5641] vtxo tree signed for round f8f94f3b-28c2-4dff-
b8e9-946cb6d3404e
[2025-07-19 16:58:05.536] DEBU[5641] started finalization stage for round:
f8f94f3b-28c2-4dff-b8e9-946cb6d3404e
[2025-07-19 16:58:05.536] DEBU[5641] forwarding event to 1 listeners
[2025-07-19 16:58:05.540] DEBU[5641] gRPC method: /ark.v1.ArkService/
SubmitSignedForfeitTxs
[2025-07-19 16:58:05.585] DEBU[5641] signing transaction f8f94f3b-28c2-4dff-
b8e9-946cb6d3404e
[2025-07-19 16:58:05.620] DEBU[5641] finalized round f8f94f3b-28c2-4dff-
b8e9-946cb6d3404e with round tx
05b221ee27cbe088798fc6869a22d63227352d14fb2bd4dcdf418191c5988839
[2025-07-19 16:58:05.621] DEBU[5641] forwarding event to 0 listeners
[2025-07-19 16:58:05.622] DEBU[5641] forwarding event to 1 listeners
[2025-07-19 16:58:05.681] DEBU[5641] started registration stage for new round:
31cf48e1-544c-43e3-92f7-40cce06bd0db
[2025-07-19 16:58:05.721] DEBU[5641] added 1 new vtxos
[2025-07-19 16:58:05.721] DEBU[5641] scheduled sweep for round
05b221ee27cbe088798fc6869a22d63227352d14fb2bd4dcdf418191c5988839 at 2025-07-19
17:06:37
[2025-07-19 16:58:05.722] DEBU[5641] started watching 1 vtxos
```

もう1回タイムアウトしてリカバーしたが、同じ動作をした。

別のユーザを追加して 6BTCの VTXOを作った。オペレーターの available balanceは 16BTCくらいになっている。単純に考えれば、最初に10BTC、ユーザAが1BTC、ユーザBが6BTCなので17BTCになりそうだがそうはなっていない。
この16BTCは2ユーザのVTXOがタイムアウトした後である。

VTXOがタイムアウトしたのでユーザ側で unilateral exit しようとしたが、残高がゼロだけにどうしようもなかった。リカバリーしてVTXOになったところをやればよいのか。

redeem

```
$ ./1.sh redeem --address
bcrt1pz6nyd9yjxcgm8el9wen4659akt0w3xw7sya29z7c8wsy8uv0v29sszqyc0 --amount 100000000
--password bbbbbbbb
INFO[0000] registered inputs and outputs with request id: 57c67102-
```

```
ce9f-4999-9de4-46b7de334902
INFO[0000] a round finalization started
INFO[0000] submitting forfeit transactions...
INFO[0000] done.
INFO[0000] waiting for round finalization...
WARN[0000] round failed, retrying...                     error="round failed:
failed to broadcast round tx: bad txns inputs missingorspent"
INFO[0000] registered inputs and outputs with request id: 53431617-8962-494f-b41e-
d0c1ac1e2a52
INFO[0002] a round finalization started
INFO[0002] submitting forfeit transactions...
INFO[0003] done.
INFO[0003] waiting for round finalization...
INFO[0003] round completed
039d9b269d633294439fbe315a9812d4386d24376dc918a8cd5aee4eb22fa21c
{
        "txid": "039d9b269d633294439fbe315a9812d4386d24376dc918a8cd5aee4eb22fa21c"
}
```



問題1

`./1.sh receive` のアドレスに送金してgenerateすると `arkd` 側のログに generate した block-hash のデータが取得できないというエラーが出る。この `2a8d...` はブロック高 `118` なのでそれも合ってない。

`err`

```
[2025-07-19 14:12:51.835] DEBU[0212] started registration stage for new round:
f5cb47f3-9a44-4a57-8d0b-f1a11020617f
[2025-07-19 14:12:53.439] INFO[0213] 2025-07-19 14:12:52.996 [ERR] wallet: Unable
to process chain backend block connected notification: failed to store sync
information 2a8d975d7550cded0f179542930a904f7c0cf564b4ca573375b5fac2598ea361:
failed to fetch block hash for height 117: block not found
```

よくわからないが、 `arkd` を起動したあとか、ウォレットを作成した後かわからないが、そこら辺で一度 `arkd` を再起動するとよさそうに見える。
少なくともウォレット作成後に再起動するとこのエラーは出なくなった。

VTXOの期限切れ

`expiry_time` を過ぎたのだが、違いが見えない。
この後 `send` もしたが送受金できている。

期限切れ

```
$ ./1.sh balance
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 100000000,
                "next_expiration": "8 minutes",
                "details": [
                        {
                                "expiry_time": "2025-07-18T18:32:23+09:00",
                                "amount": 100000000
                        }
                ]
        }
}
#### しばらく放置 ####
$ date ; ./1.sh balance
Fri Jul 18 18:36:03 JST 2025
{
        "onchain_balance": {
                "spendable_amount": 0
        },
        "offchain_balance": {
                "total": 100000000,
                "next_expiration": "3 minutes",
                "details": [
                        {
                                "expiry_time": "2025-07-18T18:32:23+09:00",
                                "amount": 100000000
                        }
```

```
                ]
            }
    }
```

expiry_time には `arkd` のログに

`arkd` を再起動すると sweep txなるものが展開されて、それはオペレータへの入金になった。
入金のINPUTはクライアントが settle したときのトランザクションである。

arkd起動ログ

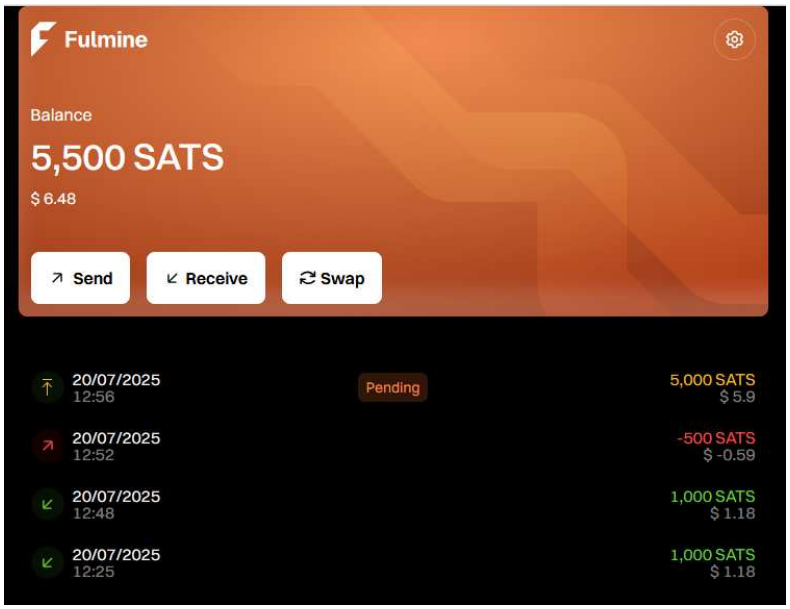
```
DEBU[0011] restored watching 3 vtxos
DEBU[0011] starting sweeper service...
DEBU[0011] sweeper:
493e6f87936b5232c7c9db83cdd3f9bca9f2bf570f9bd62c5bccadd2c851aecc
DEBU[0011] sweep tx broadcasted:
b5b2170f94ed439584c8105dbcc55b587f43350592536a8a37e9346e7eeae43d
DEBU[0011] 1 vtxos swept
```



---

fulmine

Fulmine - Docs - Ark Labs



Bitcoin / Lightning Network / Ark のウォレットであるそうだが、今のところ Ark としてのみ使っている。
ウォレットというよりは wallet daemon で、ブラウザでできることは最低限というところか。

esplora が WSL側にあるのは、単にRaspberry Pi3側で起動できなかったせいである。

start-fulmine.sh

```
#!/bin/bash

export FULMINE_DATADIR="./data/fulmine"
export FULMINE_HTTP_PORT=7001
export FULMINE_GRPC_PORT=7000
export FULMINE_ARK_SERVER="http://192.168.0.85:7070"
export FULMINE_ESPLORA_URL="http://192.168.0.85:3002"
export FULMINE_UNLOCKER_TYPE="env"
export FULMINE_UNLOCKER_PASSWORD="aaaaaaaa"
export FULMINE_NO_MACAROONS=true
export FULMINE_LOG_LEVEL=5

./bin/fulmine
```

`arkd` のバージョンとの組み合わせがあるようで、v0.1.14 が `arkd` のv0.6.3に対応しそうだった。
ArkLabsHQ/fulmine at v0.1.14

Goなどをインストールしてビルドすることもできるが、バイナリをダウンロードしても良い。
今回はそうした。
実行すると `./data/fulmine/` 以下に設定を保存している。やり直したいときは削除すると良い。
環境設定をし忘れて立ち上げたときもデフォルト値が書き込まれているため削除することになる(エラーで起動しないから気付くだろう)。

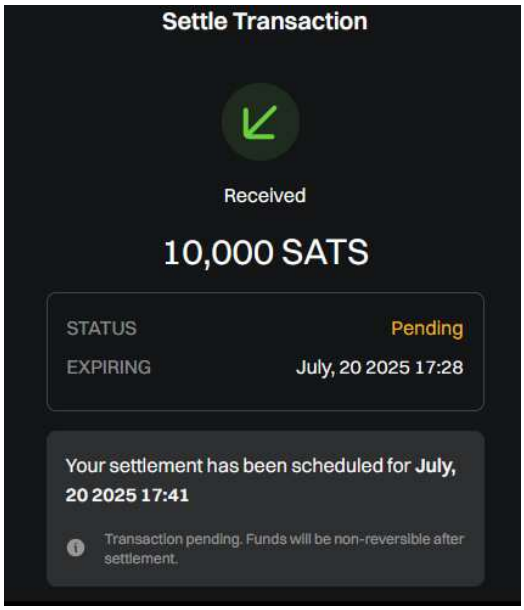
ブラウザでは http://localhost:7001/ にアクセスする。
ブラウザ以外では REST と gRPC をサポートしている。

api

```
$ curl -s http://localhost:7001/api/v1/address | jq .
{
  "address":
"bitcoin:bcrt1p2j3f4xcj5mcwmgqvfuczftcxgnjwp3avl5u6nxcxx03ndemzspnsxpepwp?
ark=tark193aazakxpmf78720wk6yazgx89tkcch0p339djmfwlahhvnzqltjhhy7829jd255gnuetv40gx
ystvm899tezl5473cgpspvyel4c5g5c53h8",
  "pubkey": "0252672e568a6a308220d88f6accc670944092de0649b8d78d8787f90268d2910a"
}
```

fullminで最初に受け取った1000satsが、しばらくすると 0sats になって現れないままである。
fullminの最初に貼った画像がそうなのだが合計があわない。

受金するとなぜか Settle Transaction になることがある。



これはメニューに「Settle Now」があり、選択するとトランザクションが展開された。


34b60a4769413ba504346dbb0cc55f35684e7c090384902fd2dc2367a52f47bc
Details +
#0 cff37c0ac46c730d4f8c5476c4a9edc18533e3704d90 8b2e72f8320e0d13aec5:1 0.00000575 BTC
#1 cff37c0ac46c730d4f8c5476c4a9edc18533e3704d90 8b2e72f8320e0d13aec5:2 4.99997787 BTC
#0 bcrt1pejg28ty7v48kta7kt8q6a8ldp04tqupteczp932gkk04z0dwr5rq0985gj 0.00015155 BTC
#1 bcrt1prq35dda6zkn6r9wgnrg99qelmm77dtkwj3qnw5phls3fnpzps2fsgfy6ql 0.00001392 BTC
#2 bcrt1qlq5uxvy2t7eglr43xyc3lq6cjf55qugggfkhmk 4.99981561 BTC
1 CONFIRMATION
4.99998108 BTC