# How To Lock Funds and then Unlock Them

This document describes procedure how to lock and unlock funds with programs locked\_tx\_pusher.cpp and locked\_tx\_spender.cpp, version 0.001.

The procedure assumes we have some funds at some address, and the input comes from a single funding transaction output.

## Prerequisites

1. Electrum installed and runnable with an active wallet.
2. Bitcoin node running? TBD – check if works without it

## Input

We have some funds available at address A, for which we have private key P. Funds are coming from a single funding transaction FT, from output index (vout) FI.

We want to lock the funds until some point in time, LOCK\_TIME.

We need to specify AMOUNT, which should be all the money from FT/FI.

We need to specify FEE.

## Output

Assuming that real time is now beyond the LOCK\_TIME, we have funds available at target address TA (minus fees).

Say, we have 0.01 BTC (testnet) available at address

**mxBcb6aCmwcPHfjyd4ePf3UWyDXqJGw3Ki**

Make sure the funds are available:

**bx fetch-balance** **mxBcb6aCmwcPHfjyd4ePf3UWyDXqJGw3Ki**

balance

{

address mxBcb6aCmwcPHfjyd4ePf3UWyDXqJGw3Ki

received 1000000

spent 0

}

**bx fetch-utxo 1000000**  **mxBcb6aCmwcPHfjyd4ePf3UWyDXqJGw3Ki**

points

{

point

{

hash a0043c3e6080ff7867e609ecfe9349c36e37618b47441750a9aa1093d9d059ec

index 1

value 1000000

}

}

We need A, P, FT, FI, LOCK\_TIME.

A = **mxBcb6aCmwcPHfjyd4ePf3UWyDXqJGw3Ki**

P = **cPMQ45cg5irwpPdhUEJ565mRwQTYN2TRczwffoALBohvyM84Jmgu**

FT = a0043c3e6080ff7867e609ecfe9349c36e37618b47441750a9aa1093d9d059ec

FI = 1

LOCK\_TIME = 1615323600

AMOUNG = 1000000

FEE = 50000

P is obtained from Electrum, go to “addresses”, find address mxBcb6a… , righ-click, click provate key, enter password, grab the private key.

FT, FI, AMOUNT are taken from the output of bx fetch-utxo as above.

LOCK\_TIME is an epoch time of some point in the future.

FEE needs to be determined via internet lookup or some other means.

With A,P, FT, FI, LOCK\_TIME, AMOUNT, and FEE we open **locked\_tx\_pusher.cpp**.

There you set:

privKeyWIF = P

srcTxId = FT

srcTxOutputIndex = FI

satoshisToTransfer = AMOUNT – FEE (i.e. 950000)

lockUntil = LOCK\_TIME

Note that A is not needed, but it is good to take a note of it, as it is a source of a private key on Electrum.

Make sure that Makefile.am is as follows:

bin\_PROGRAMS = bing  
bing\_SOURCES = src/locked\_tx\_pusher.cpp  
  
bing\_LDADD= @bitcoin\_client\_LIBS@

cd to the directory where the project is located, e.g. /Users/miloszm/cpp/bingdiyi

make

./bing

Output should be:

Raw Transaction with frozen output until 1615323600:

0000000001ec59d0d99310aaa9501744478b61376ec34993feec09e66778ff80603e3c04a0010000006a4730440220789c831fff5f13922ef1f04f97f0d43dd2082fdc3939fde846b3216740c860450220625c657b33a4a0a8c6e2e0b43a802a2d458dcd6842b85393408859b8b17daa6001210248648a8cc529219e777fb7e5791aa002c23d3abe55ea78eb3011f949401b8782feffffff01f07e0e000000000017a9147965f1f33228354cb0ce53abda62a92946b4394a8700000000

Copy the highlighted transaction hex and issue command:

bx send-tx <highlighted hex>

This will broadcast a new transaction that should be visible in Electrum’s history panel. Your 950000 satoshis of Bitcoin funds are now frozen until 1615323600.

## You can now try to unfreeze the funds and see that it cannot be done

Wait until your “freezing” transaction gets confirmed.

Note the new TX ID: 03c3ef4afd4d44a84be8f279e69dbac4a0353b1debadb5372dc17d5aa25c9445

Open **locked\_tx\_spender.cpp**

In it fill out the following:

privKeyWIF = P // same as before

srcTxId = 03c3ef4afd4d44a84be8f279e69dbac4a0353b1debadb5372dc17d5aa25c9445

scrTxOutputIndex = 0 // always, you don’t need to look it up

satoshisToTransfer = 900000 //950000 – 50000 fee

srcLockUntil = LOCK\_TIME // same as before

targetAddr = TA // some target address for your fund

Make sure Makefile.am looks as follows:

bin\_PROGRAMS = bing  
bing\_SOURCES = src/locked\_tx\_spender.cpp  
  
bing\_LDADD= @bitcoin\_client\_LIBS@

make

./bing

Output should be:

Raw Transaction:

010000000145945ca25a7dc12d37b5adeb1d3b35a0c4ba9de679f2e84ba8444dfd4aefc303000000007347304402204099cf459742640479e4563a1365635f52077d56565a792b4135b3b6f19a255202203f23b997f8edb3f3d837ae27e871ab0003e17ef45ba22a580dd017dd778f8802012a04d0e14760b175210248648a8cc529219e777fb7e5791aa002c23d3abe55ea78eb3011f949401b8782ac0000000001a0bb0d00000000001976a9143558bb54951f06a9b63fd91e8ea8a727a36fe79388acd0e14760

Copy the highlighted transaction hex and issue command:

bx send-tx <highlighted hex>

You should see the message:

**transaction currently non-final for next block**

This is because your LOCK\_TIME has not elapsed yet. You need to wait until the LOCK\_TIME time, and then still wait ca 11 blocks, and then try again. Your locks should become available at your target address TA: mkP2QQqQYsReSpt3JBoRQ5zVdw3ra1jenh

So let’s wait until 21:00 and see what happens…….

You can do bx fetch-heigh to see the current block number.

If your time (21:00) has arrived, note the block number and wait for another 11 blocks.

Then try your transaction again (you may try sooner as well).

Here is what happened in this particular case:

TIME\_LOCK was set to 1615323600 which is 22:00, March 9th, 2021, Warsaw time.

time block message from bx send-tx <transaction hex>

22:00 1939651 transaction currently non-final for next block

22:06 1939652 transaction currently non-final for next block

22:18 1939653 transaction currently non-final for next block

22:33 1939654 transaction currently non-final for next block

22:47 1939655 transaction currently non-final for next block

23:04 1939656 transaction currently non-final for next block

23:10 1939658 Sent transaction.

It looks like after about 7 blocks the transaction was deemed as not locked.

This is related to Bitcoin measure of time using ‘median time past’, MTP, which is described in BIP 113. Timestamp-based locks use the median timestamp of previous 11 blocks. For more info refer to:

<https://medium.com/summa-technology/bitcoins-time-locks-27e0c362d7a1>

<https://github.com/bitcoin/bips/blob/master/bip-0113.mediawiki>