

# Krux and miniscript

**an introduction to BIP379 and a little game**

qlrd

# Miniscript

## **Definition 1:** BIP 379.

(...) a language for writing (a subset of) **Bitcoin Scripts** in a structured way, enabling analysis, composition, generic signing and more. [1]

# Back to the basics

# Bitcoin script

## **Definition 2: .**

(...) an unusual stack-based language with many edge cases designed for implementing spending conditions consisting of various combinations of signatures, hash locks, and time locks. [1]

## Bitcoin script

Common transactions from [2] and [3]

Comment	Unlock	Lock
P2PK	<sig> <pk>	OP_CHECKSIG
P2PKH	<sig> <pk>	OP_DUP OP_HASH160 <pkh> OP_EQUALVERIFY OP_CHECKSIG
Multisig 2-of-3	OP_0 <sigA> <sigB>	2 <pkA> <pkB> <pkC> 3 OP_CHECKMULTISIG

## Bitcoin script

Freezing funds until a time in the future from [2]

Unlock	Lock
<sig> <pk>	<expiry time> OP_CHECKLOCKTIMEVERIFY OP_DROP OP_DUP OP_HASH160 <pkh> OP_EQUALVERIFY OP_CHECKSIG

## Bitcoin script

Timelock variable multisignature from [3]: 2-of-3 multisig; after 30 days 1-of-3 with a lawyers's signature; after 90 days the lawyer's signature.

Unlock	Lock
<pre>OP_0 &lt;sigA&gt; &lt;sigB&gt; OP_TRUE       OP_TRUE</pre>	<pre>OP_IF OP_IF 2 OP_ELSE &lt;30 days&gt; OP_CHECKSEQUENCEVERIFY       OP_DROP &lt;sigD&gt; OP_CHECKSIGVERIFY 1 OP_ENDIF       &lt;sigA&gt; &lt;sigB&gt; &lt;sigC&gt; 3 OP_CHECKMULTISIG OP_ELSE &lt;90 days&gt; OP_CHECKSEQUENCEVERIFY       OP_DROP &lt;sigD&gt; OP_CHECKSIG       OP_ENDIF</pre>

# The issue [1]



Given a combination of spending conditions, it is challenging to:

- find the most economical script to implement it;
- implement a composition of their spending conditions;
- find out what spending conditions it permits.

...

# The motivation

**Miniscript** has a structure that allows composition: a representation for **scripts** that makes these type of operations possible.

# Specification [1]

# Specification

Miniscript analyzes scripts to determine properties.

# Specification

**Not expected** to be used with:

- BIP 16 (p2sh);

**Expected** to be used within:

- BIP 382: wsh descriptor;
- BIP 386: tr descriptor.

And together with:

- BIP 380: Key expressions:

[<fingerprint>/<purpose>/<cointype>/<index>]

# Specification

From a user's perspective, Miniscript is not a separate language, but rather a significant expansion of the descriptor language. [1]

# Specification

Liana's simple inheritance wallet [4].

```
wsh(  
  or_d(  
    pk([07fd816d/48'/1'/0'/2']tpub...wd5/<0;1>/*),  
    and_v(  
      v:pkh([da855a1f/48'/1'/0'/2']tpub...Hg5/<0;1>/*),  
      older(36)  
    )  
  )  
)#lz4jfr7g
```



## Specification

Liana's simple inheritance wallet TR [5]. First key expression is a NUMS (“nothing-up-my-sleeves”) point [6].

```
tr(  
  [07fd816d/48'/1'/0'/2']tpub...mwd5/<0;1>/*,  
  and_v(  
    v:pk([da855a1f/48'/1'/0'/2']tpub...Hg5/<0;1>/*),  
    older(36)  
  )  
)#506utvsp
```

## Specification

Liana's variable multisig [7].

```
wsh(  
  or_d(  
    multi(2,  
      [07fd816d/48'/1'/0'/2']tpub...wd5/<0;1>*,  
      [da855a1f/48'/1'/0'/2']tpub...Hg5/<0;1>*  
    ),  
    and_v(  
      v:thresh(2,  
        pkh([07fd816d/48'/1'/0'/2']tpub...mwd5/<2;3>*),  
        a:pkh([da855a1f/48'/1'/0'/2']tpub...Hg5/<2;3>*),  
        a:pkh([cdef7cd9/48'/1'/0'/2']tpub...Ak2/<0;1>*)  
      ),  
      older(36)  
    )  
  ) )#wa74c6se
```

## Specification

Liana's variable multisig TR [8]. First key expression is a NUMS (“nothing-up-my-sleeves”) point [6].

```
tr(tpub...pMN/<0;1>/*, {  
  and_v(  
    v:multi_a(2,  
      [07fd816d/48'/1'/0'/2']tpub...mwd5/<2;3>/*,  
      [da855a1f/48'/1'/0'/2']tpub...DHg5/<2;3>/*,  
      [cdef7cd9/48'/1'/0'/2']tpub...SAk2/<0;1>/*  
    ),  
    older(36)  
  ),  
  multi_a(2,  
    [07fd816d/48'/1'/0'/2']tpub...mwd5/<0;1>/*,  
    [da855a1f/48'/1'/0'/2']tpub...DHg5/<0;1>/*  
  )  
})#tvh3u2lu
```

# Specification

## Definition 3: .

**Miniscript** consists of a set of **script** fragments which are designed to be safely and correctly composable (...) targeted by spending policy compilers

# Implementations

- Peter Wuile's reference implementation
- C++:
  - Bitcoin-core
- Rust:
  - rust-miniscript
  - Liana
- Go:
  - Tutorial: Understanding Bitcoin Miniscript - Part III
- Python:
  - Embit's miniscript.py
  - Krux (branch p2wsh\_miniscript)
  - Krux (branch tr\_miniscript)

## Hands on: setup a single-inheritance scheme



Figure 1: Before start, download a Krux demo android app [https://github.com/odudex/krux\\_binaries/blob/main/Android/Krux\\_25.01.beta8\\_Android\\_0.2.apk](https://github.com/odudex/krux_binaries/blob/main/Android/Krux_25.01.beta8_Android_0.2.apk)

## Hands on: setup a single-inheritance scheme (ii)



Figure 2: Before start, download Liana coordinator in your computer <https://wizardsardine.com/liana/>

## Hands on: setup a single-inheritance scheme (iii)

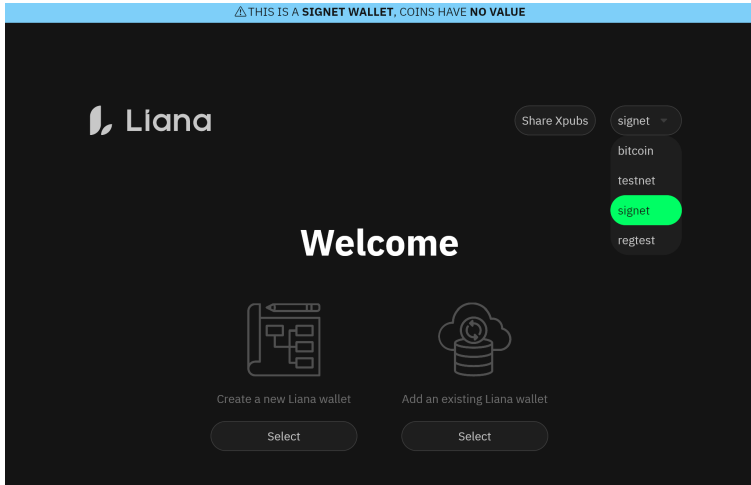


Figure 3: We will select **signet** to not risk our beloved sats.



## Hands on: setup a single-inheritance scheme (iv)

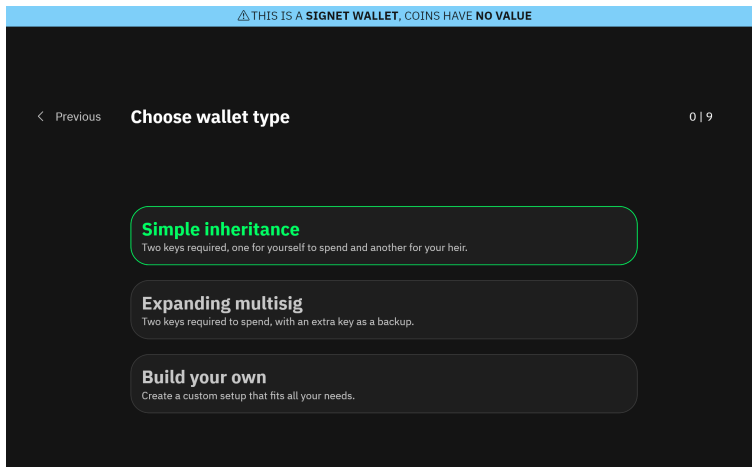


Figure 4: Selecting **Simple inheritance** scheme.



# Hands on: setup a single-inheritance scheme (v)

THIS IS A SIGNET WALLET, COINS HAVE NO VALUE

< Previous **Introduction** 1 | 9


## Simple inheritance wallet

For this setup you will need 2 Keys: Your Primary Key (for yourself) and an Inheritance Key (for your heir). For security reasons, we suggest you use a separate Hardware Wallet for each key.


 **Primary key**  **Inheritance key**

You will always be able to spend using your Primary Key.  
After a period of inactivity (but not before that) your Inheritance Key will become able to recover your funds.

Primary spending policy:

 **Primary Key**

Recovery spending policy:

 **Inheritance Key**

Receipt of funds

After some time\* of wallet inactivity

**CAN SPEND**

**CAN'T SPEND (TIMELOCKED)**

**CAN SPEND**

\*The time range (timelock) for the activation of keys can be configured in the next step.

Next

Figure 5: An explanation how **Simple inheritance** scheme works.

## Hands on: setup a single-inheritance scheme (vi)

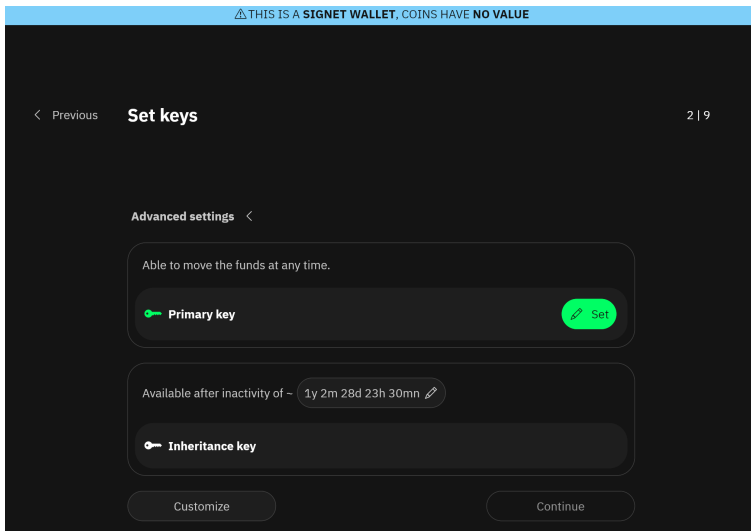


Figure 6: Liana's menu to setup **Simple inheritance**.

## Hands on: setup a single-inheritance scheme (vii)

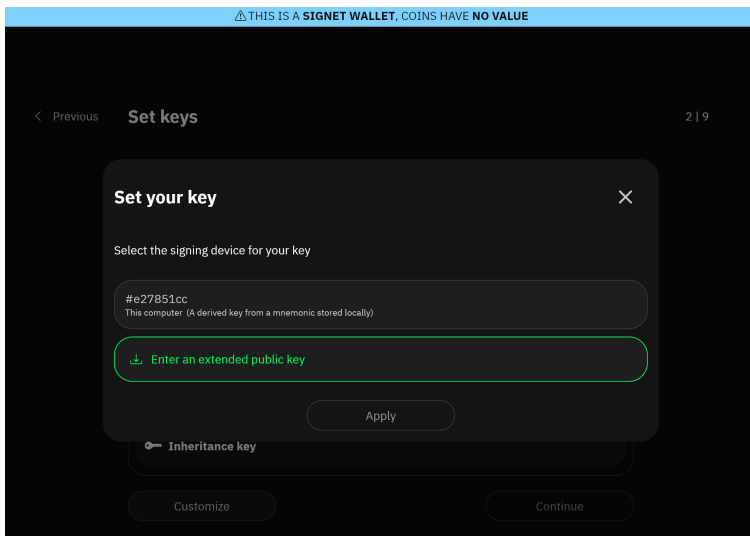


Figure 7: Liana's menu to setup **Simple inheritance**.

## Hands on: setup a single-inheritance scheme (viii)

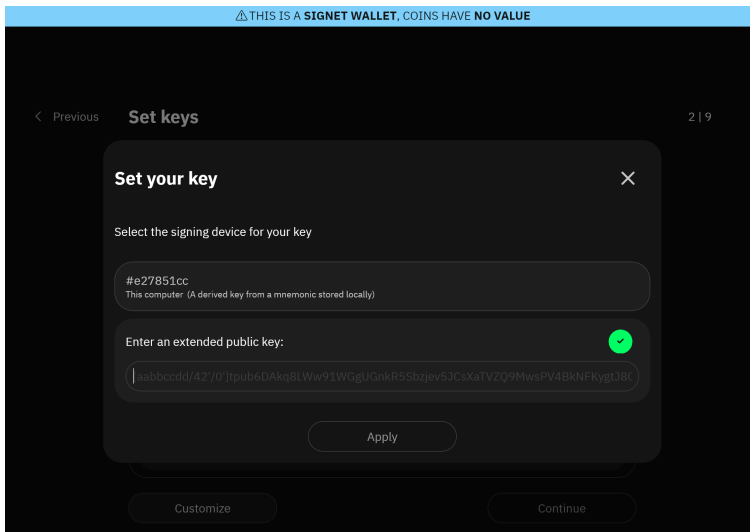


Figure 8: Liana's waiting for the first (key-expression + xpub) to setup **Simple inheritance**.

## Hands on: setup a single-inheritance scheme (ix)

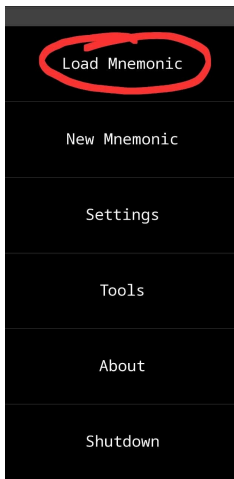


Figure 9: Load a previous created wallet on **Krux**.

## Hands on: setup a single-inheritance scheme (x)

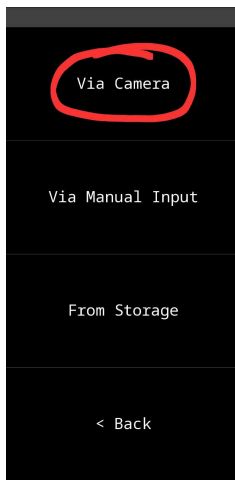


Figure 10: Select **Via Camera** to Load.

## Hands on: setup a single-inheritance scheme (xi)

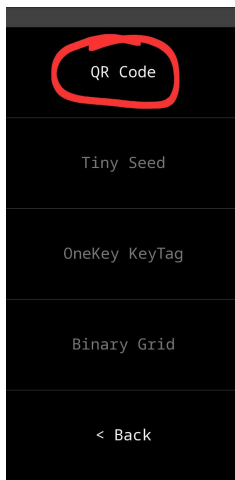


Figure 11: Select **QR Code** to scan.



## Hands on: setup a single-inheritance scheme (xii)

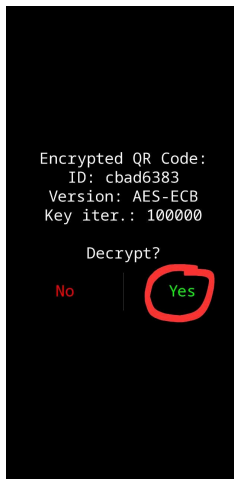


Figure 12: Decrypt a encrypted mnemonic.

## Hands on: setup a single-inheritance scheme (xiii)

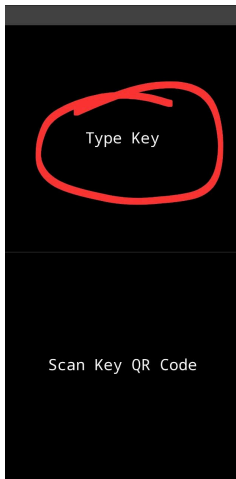


Figure 13: Select **Type Key** to type a decrypt key. Alternatively, you can scan a previous created QRCode key.

## Hands on: setup a single-inheritance scheme (xiv)

Key				
a	b	c	d	e
f	g	h	i	j
k	l	m	n	o
p	q	r	s	t
u	v	w	x	y
z	ABC	<	Esc	Go

Figure 14: **Type Key** keyboard. Try something like 'test' or another one.

## Hands on: setup a single-inheritance scheme (xv)

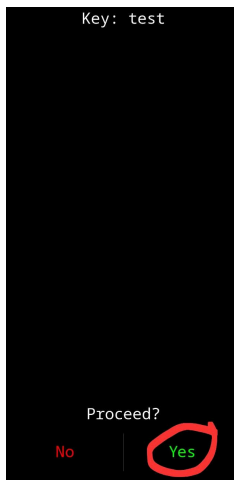


Figure 15: 'test' was typed as the decrypt key.

## Hands on: setup a single-inheritance scheme (xvi)

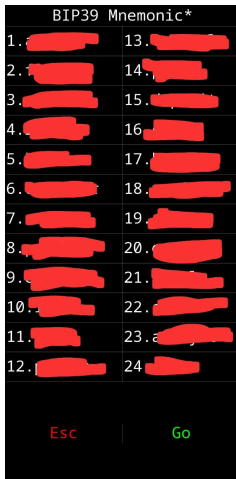


Figure 16: Decrypted a double-mnemonic (see \*, the first 12 words are a valid mnemonic; the last 12 are a valid mnemonic and the 24 words are another valid mnemonic).

## Hands on: setup a single-inheritance scheme (xvii)

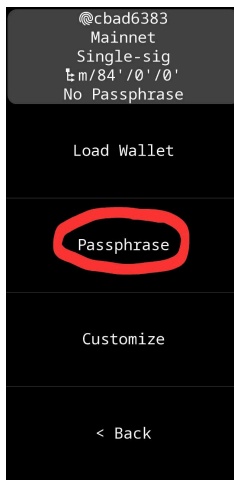


Figure 17: Loaded wallet secured by an optional BIP39 passphrase.

## Hands on: setup a single-inheritance scheme (xviii)

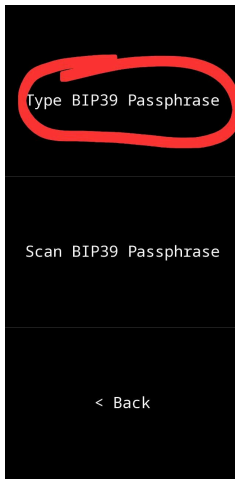


Figure 18: Select **Type BIP39 Passphrase** to access keyboard. Alternatively you can scan an QRCode encoded one.

## Hands on: setup a single-inheritance scheme (xix)

Passphrase				
a	b	c	d	e
f	g	h	i	j
k	l	m	n	o
p	q	r	s	t
u	v	w	x	y
z	ABC	<	Esc	Go

Figure 19: Select **Type BIP39 Passphrase** to access keyboard. Alternatively you can scan an QRCode encoded one.



## Hands on: setup a single-inheritance scheme (xx)

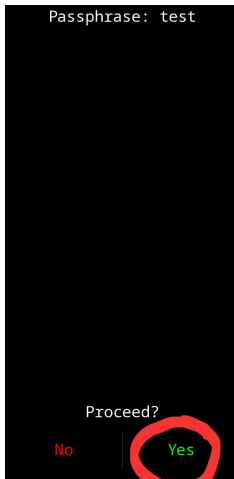


Figure 20: Krux ask for passphrase confirmation. Remember that different passphrases leads to different wallets!

## Hands on: setup a single-inheritance scheme (xxi)

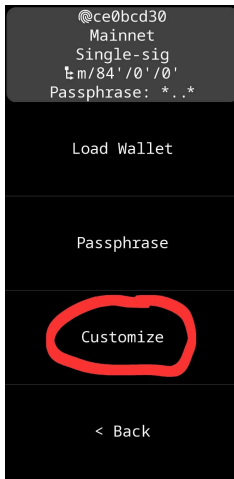


Figure 21: A different wallet was loaded (verify the upper checksum). But we still need to customize some stuffs.

## Hands on: setup a single-inheritance scheme (xxii)

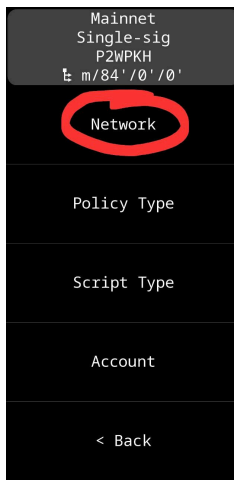


Figure 22: Let's change the network, since we're testing!

## Hands on: setup a single-inheritance scheme (xxiii)

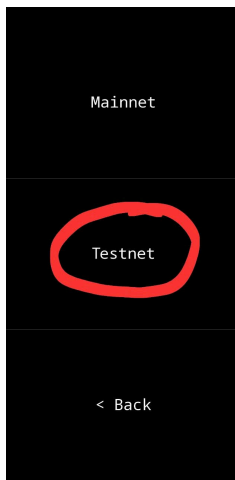


Figure 23: For signet wallet in Liana, we can use a testnet on Krux.

## Hands on: setup a single-inheritance scheme (xxiv)

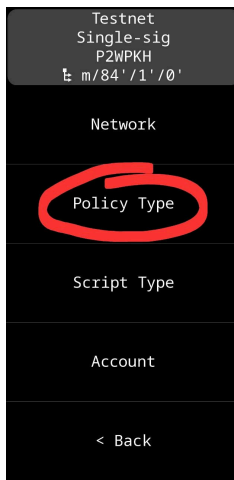


Figure 24: Let's change policy to be able to do a inheritance scheme.

## Hands on: setup a single-inheritance scheme (xxv)



Figure 25: Select miniscript policy.

## Hands on: setup a single-inheritance scheme (xxvi)

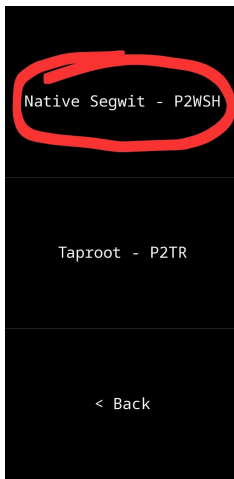


Figure 26: Then select the between BIP382 (wsh) or BIP386 (tr) descriptors.

## Hands on: setup a single-inheritance scheme (xxvii)

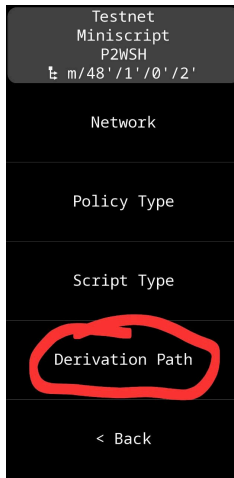


Figure 27: **Optional** you can edit your derivation path for the inheritance scheme.



## Hands on: setup a single-inheritance scheme (xxviii)

Derivation Path		
m/48' / 1' / 0' / 2'		
1	2	3
4	5	6
7	8	9
/	0	'
<	Esc	Go

Figure 28: **Optional** For educational purposes, let be the default  
m/48' / 1' / 0' / 2'.

## Hands on: setup a single-inheritance scheme (xxix)

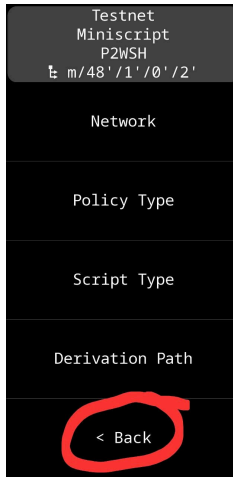


Figure 29: Now we can back to main logged menu.

## Hands on: setup a single-inheritance scheme (xxx)

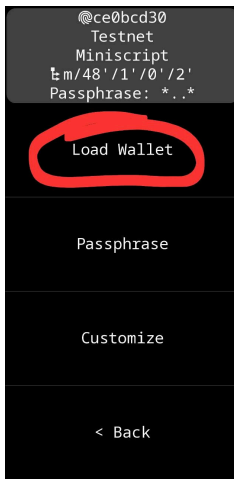


Figure 30: All done! 🎉 And load properly the wallet

## Hands on: setup a single-inheritance scheme (xxxi)

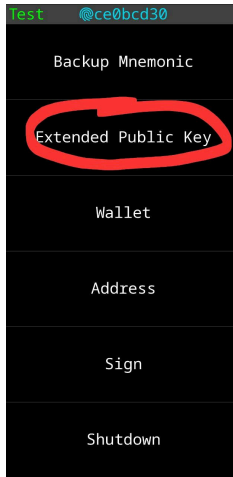


Figure 31: We need to load the key expression + tpub.

## Hands on: setup a single-inheritance scheme (xxxii)

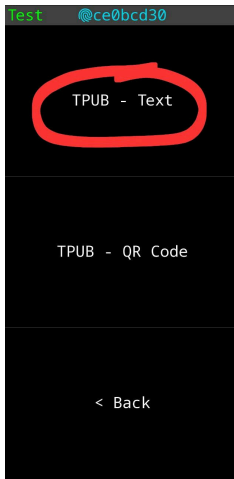


Figure 32: If you have a real krux device, it's recommended to load **TPUB - text**.

## Hands on: setup a single-inheritance scheme (xxxiii)

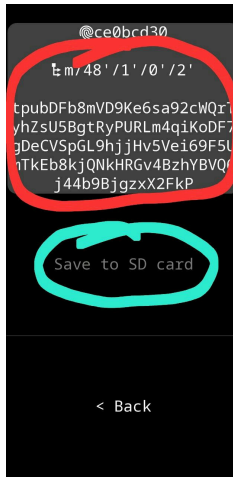


Figure 33: For real devices, save the key expression + tpub into a SDCard.

## Hands on: setup a single-inheritance scheme (xxxiv)

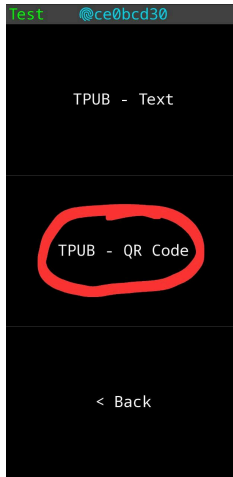


Figure 34: For demo app, press **TPUB - QRCode**.

## Hands on: setup a single-inheritance scheme (xxxv)

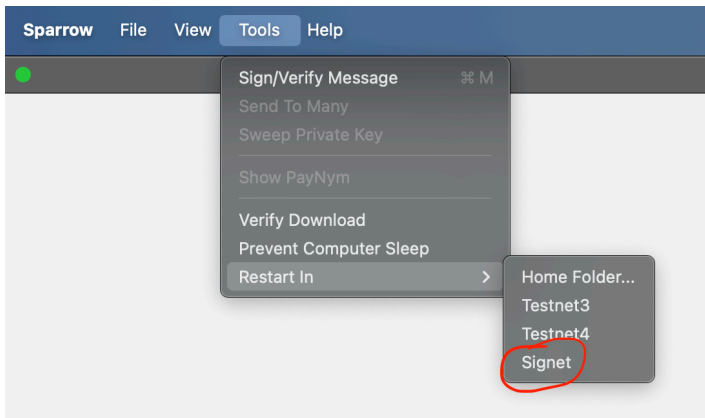


Figure 35: Open another coordinator like Sparrow and start it on signet.



## Hands on: setup a single-inheritance scheme (xxxvi)

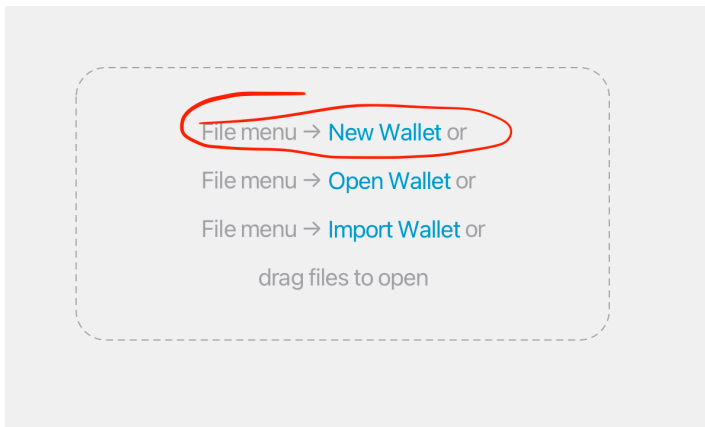


Figure 36: “Create” a new wallet.

# Hands on: setup a single-inheritance scheme (xxxvii)

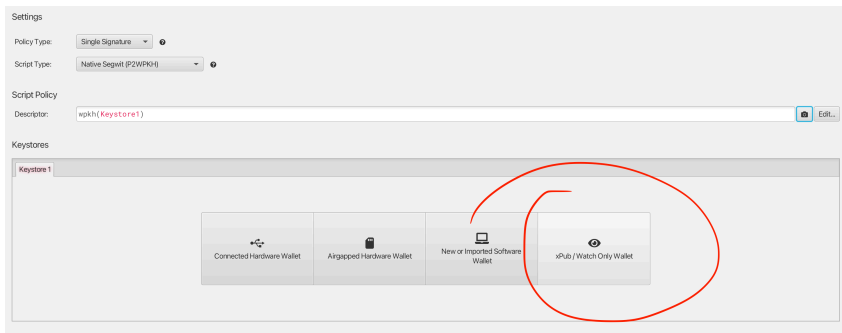


Figure 37: Select xPub/watch Only Wallet

# Hands on: setup a single-inheritance scheme (xxxxiii)


Keystores

Keystore 1

Type: ☒ Watch Only Wallet Import...

Label:

Master fingerprint:   

Derivation:  

tpub:





Figure 38: Click on the  icon to start the scanning procedure.

Keystores

Keystore 1

Type: ☒ Watch Only Wallet Import...

Label:

Master fingerprint:   

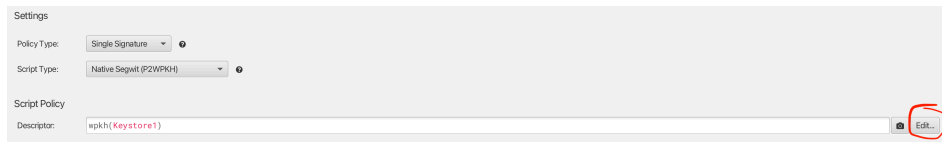
Derivation:  

tpub / vpub:

Figure 39: Once scanned, you can see the derivation-path and the tpub.

## Hands on: setup a single-inheritance scheme (xxxix)



The screenshot shows a settings interface with the following elements:

- Settings** header.
- Policy Type:** A dropdown menu set to "Single Signature" with an information icon to its right.
- Script Type:** A dropdown menu set to "Native Segwit (P2WPKH)" with an information icon to its right.
- Script Policy** section containing:
  - Descriptor:** A text input field containing the value `wpkh(Keystore1)`.
  - Edit...** button: A button with a camera icon and the text "Edit...", which is circled in red.

Figure 40: In upper section, you'll click in the “Next” button near to 📷 icon.

## Hands on: setup a single-inheritance scheme (xl)

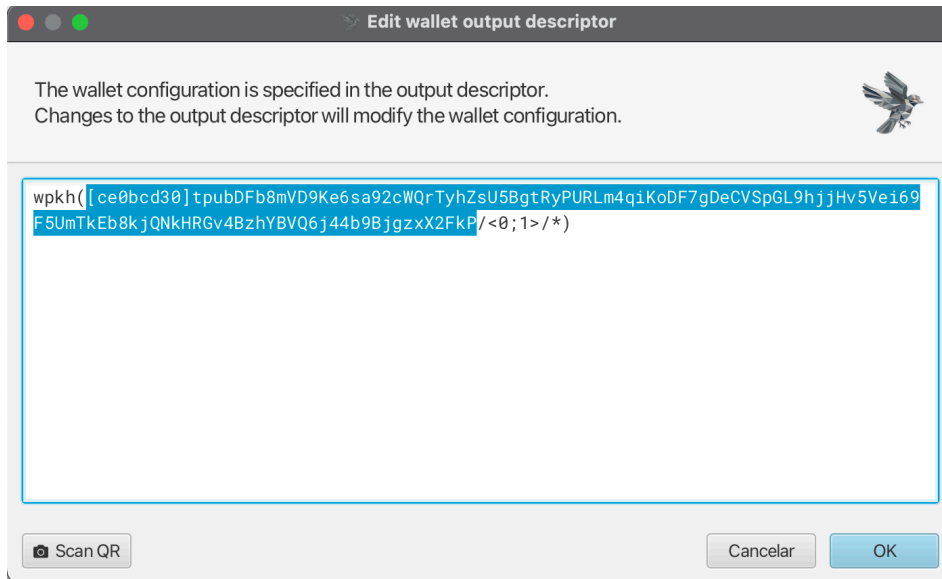



Figure 41: Copy the fingerprint + tpub (the blue part).

## Hands on: setup a single-inheritance scheme (xli)

Set your key

Select the signing device for your key

#2d6702ac  
This computer (A derived key from a mnemonic stored locally)

Enter an extended public key: 

[ce0bcd30]tpubDFb8mVD9Ke6sa92cWQrTyhZsU5BgRyPURLm4qiKoDF7gDeCVSpGL9hjjHv5

**Key name:** ⓘ  
Give this key a friendly name. It helps you identify it later

The bitcoiner me

Figure 42: Key fingerprint + tpub copied to Liana and an Alias for it.

# Exercise 1

## Exercise 1

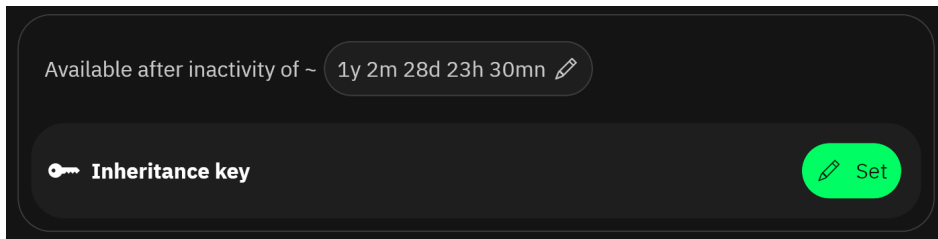


Figure 43: Set a timelock to your second key.



## Exercise 2

## Exercise 2

### Set your key ×

Select the signing device for your key

**The bitcoiner me** #ce0bcd30

#2d6702ac  
This computer (A derived key from a mnemonic stored locally)

↓ Enter an extended public key

Apply

Figure 44: Now you will repeat all previous procedures to a heir key.

**Backup your descriptor**

# Backup your descriptor

Advanced settings <

Able to move the funds at any time.

Primary key **The bitcoiner me** ✓ Edit

Available after inactivity of ~ 1h Edit

Inheritance key **the bitcoiner heir** ✓ Edit

Customize Continue

Figure 45: Check if all is ok.

## Backup your descriptor (ii)

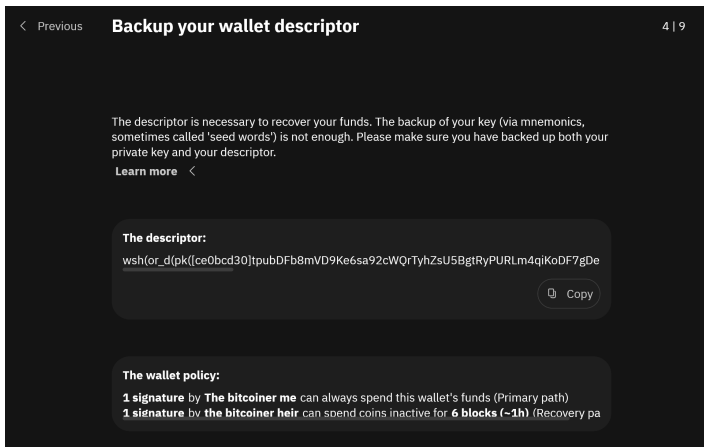


Figure 46: Check the policy and backup the miniscript descriptor on a SDCard to load it with Krux.

**Select a new node**

# Select a new node

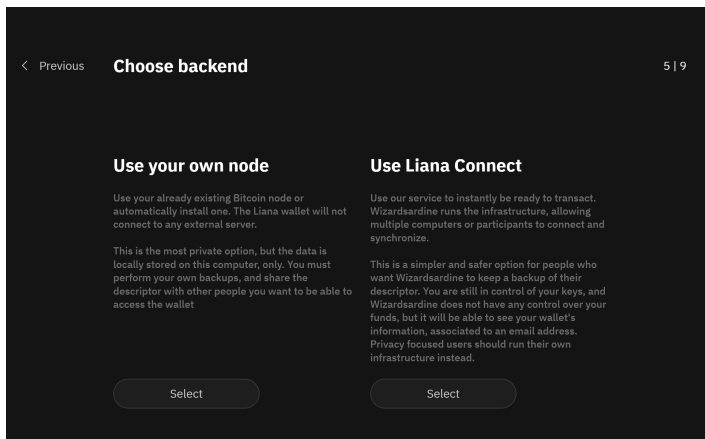
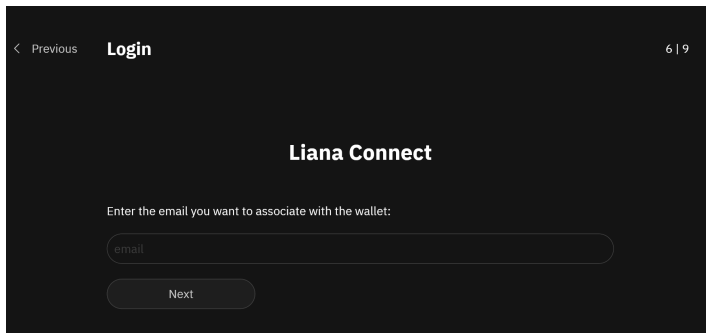


Figure 47: Select a proper node. For the workshop purpose, select Liana Connect.

## Select a new node (ii)



The screenshot shows a dark-themed login interface for 'Liana Connect'. At the top left, there is a back arrow and the text 'Previous'. In the top center, the word 'Login' is displayed in a bold, white font. At the top right, the page number '6 | 9' is visible. The main heading 'Liana Connect' is centered in a bold, white font. Below this, a prompt reads 'Enter the email you want to associate with the wallet:'. Underneath the prompt is a wide, rounded rectangular input field with a light gray border and the placeholder text 'email'. Below the input field is a rounded rectangular button with the text 'Next' in a light gray font.

Figure 48: Put your email to receive an OTP.



**Thanks!**

# Bibliography

- [1] Bitcoin Improvement Proposals, “BIP 379: Miniscript Policy.” [Online]. Available: <https://github.com/bitcoin/bips/blob/master/bip-0379.md>
- [2] Bitcoin FAQ, “Script.” [Online]. Available: <https://en.bitcoin.it/wiki/Script>
- [3] A. M. Antonopoulos and D. A. Harding, “Mastering Bitcoin: Programming the Open Blockchain (Third Edition).” [Online]. Available: <https://github.com/bitcoinbook/bitcoinbook>
- [4] jdlcdl, “Bitcoin Core Watch-Only: Liana Simple-Inheritance WSH.” [Online]. Available: <https://gist.github.com/jdlcdl/b0dea22a8a6caf0fd7c40b244357d8d2>
- [5] jdlcdl, “Bitcoin Core Watch-Only: Liana Simple-Inheritance TR.” [Online]. Available: <https://gist.github.com/jdlcdl/b17c6b551839adb5b7b7d4ef9574e48e>

- [6] jaonoctus, “NUMS secp256k1 .” [Online]. Available: <https://nums-secp256k1.jaonoctus.dev/>
- [7] jdlcdl, “Bitcoin Core Watch-Only Liana Expanding-Multi WSH.” [Online]. Available: <https://gist.github.com/jdlcdl/d83a83ec7d47d98888a8647b636d567d>
- [8] jdlcdl, “Bitcoin Core Watch-Only Liana Expanding-Multi TR.” [Online]. Available: <https://gist.github.com/jdlcdl/c38e1b80cd814e48e1d158a98cf704f6>