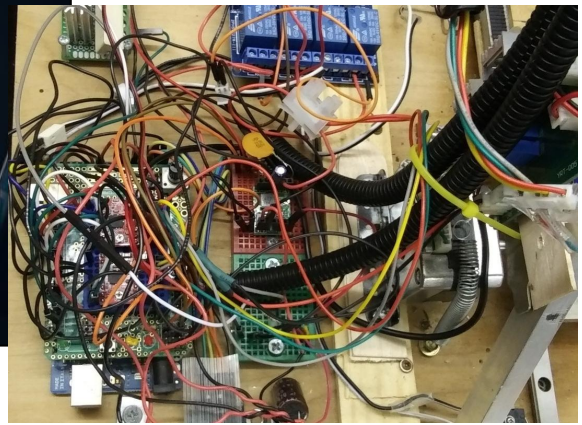
Bitcoin & BigQuery

Introduction to fraud detection

About me

- nam
- city:
- occu
- in th
- lang
- duri
  - family
  - offshore sailing
  - finding solution

ment: **>10 years**



Story of my life

TODOs

Cryptography 101

Bitcoin Data Model

Google Bitcoin Public Dataset Overview

Simple Fraud Detection

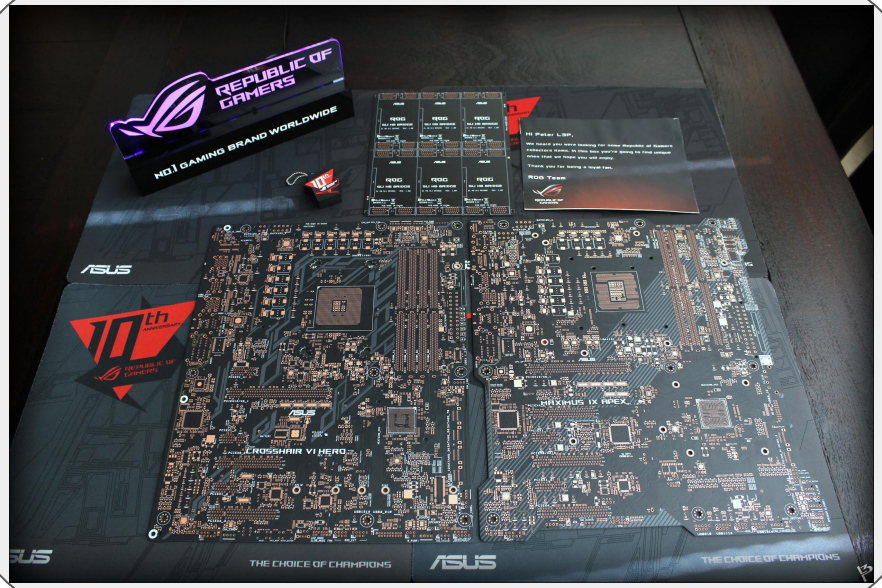
Chapter 0:

Cryptography 101

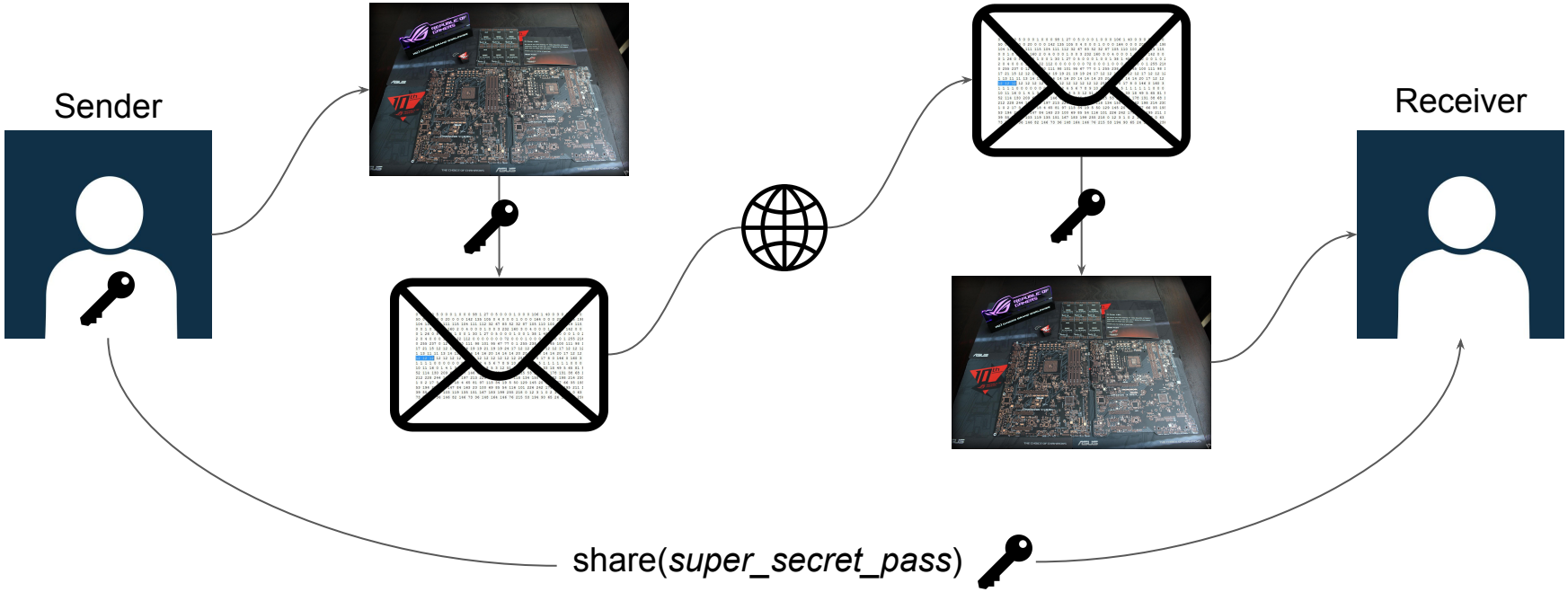
Chapter 0: Cryptography 101 - use cases

Send nudes

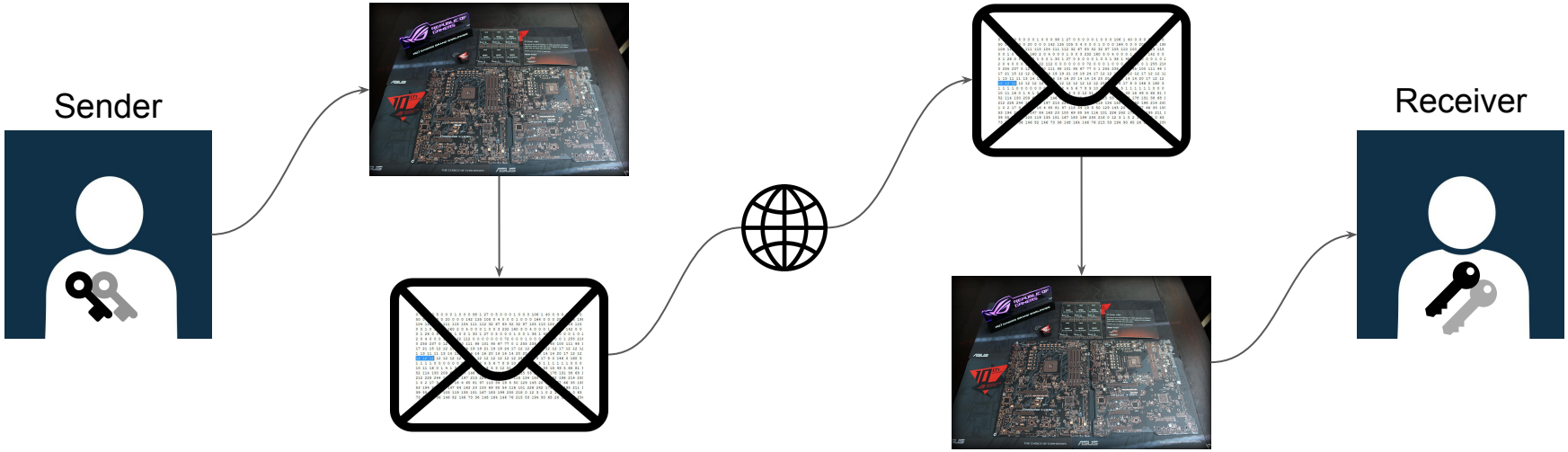
Sure!



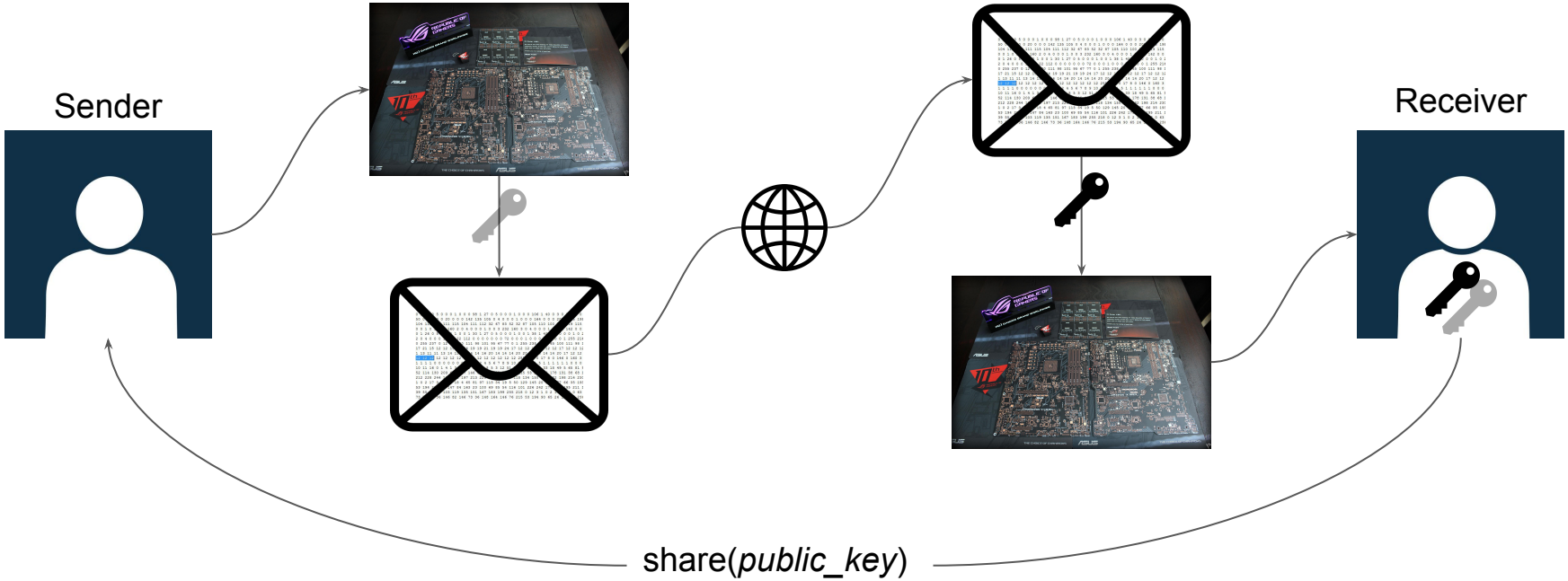
Chapter 0: Cryptography 101 - symmetric



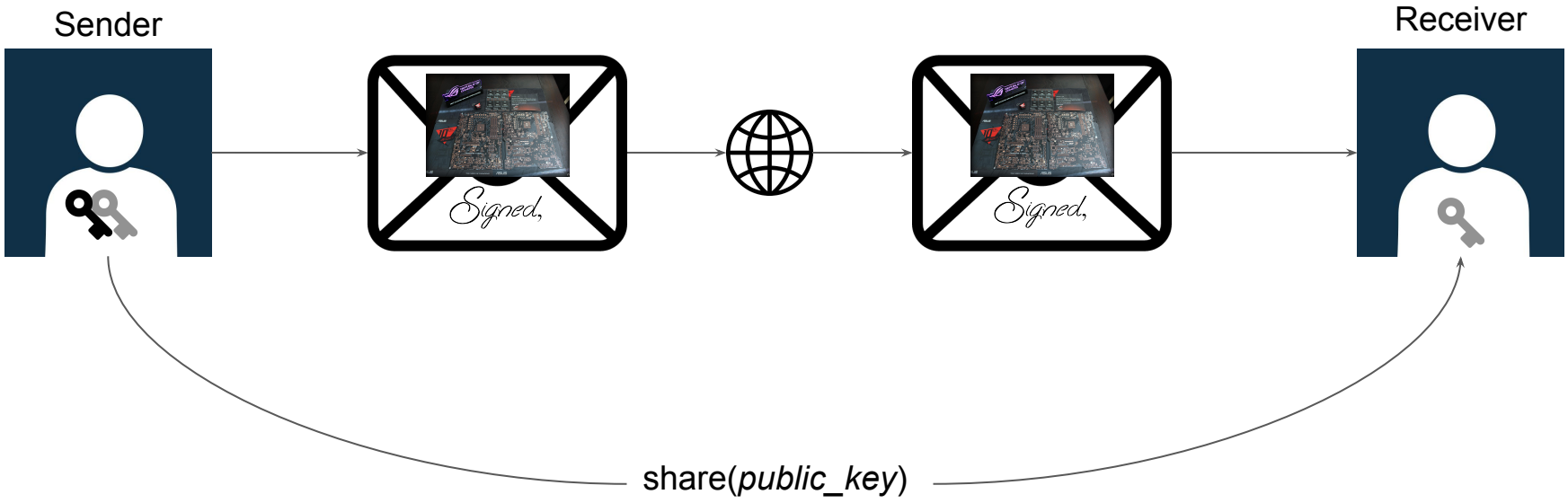
Chapter 0: Cryptography 101 - test question



Chapter 0: Cryptography 101 - asymmetric



Chapter 0: Cryptography 101 - signing



Chapter 0: Cryptography 101 - summary

- Symmetric cryptography

- Asymmetric cryptography
  - Encrypting & Decrypting
  - **Signing**

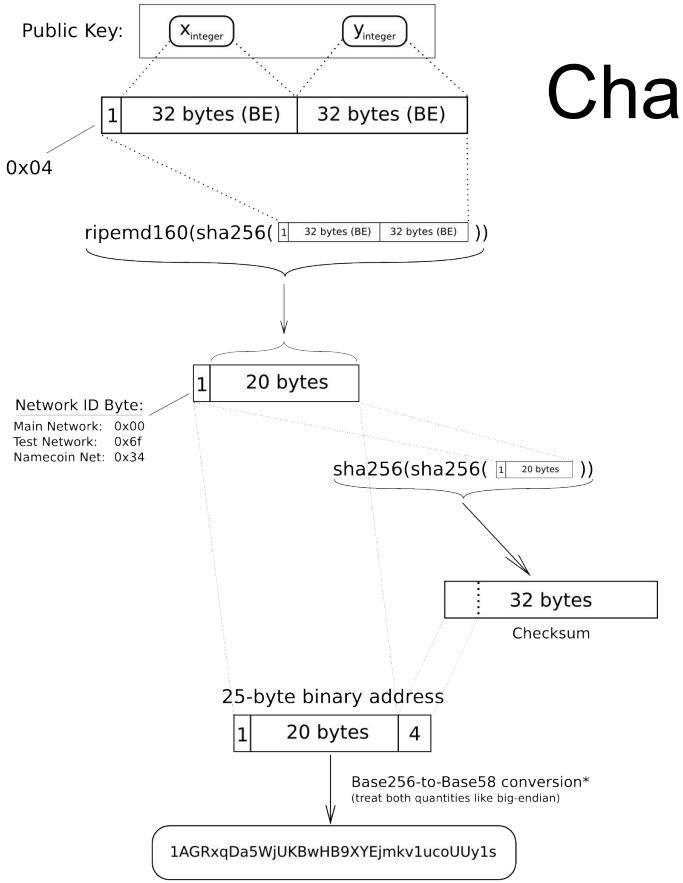
Chapter 1:

Bitcoin Model

Chapter 1: Bitcoin Model - address

https://www.blockchain.com/btc/address/1DBqpfptUMizLDACvMVsrJFPdMtbgmZCk1

Chapter 1: Bitcoin Model - address



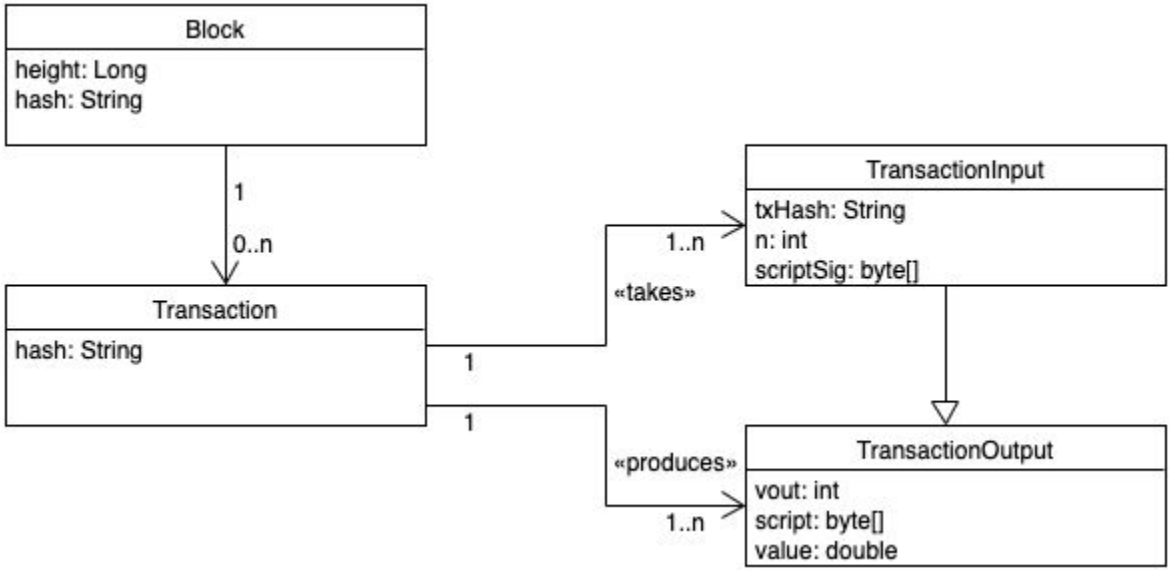
https://en.bitcoin.it/wiki/Technical_background_of_version_1_Bitcoin_addresses

Chapter 1: Bitcoin Model - value

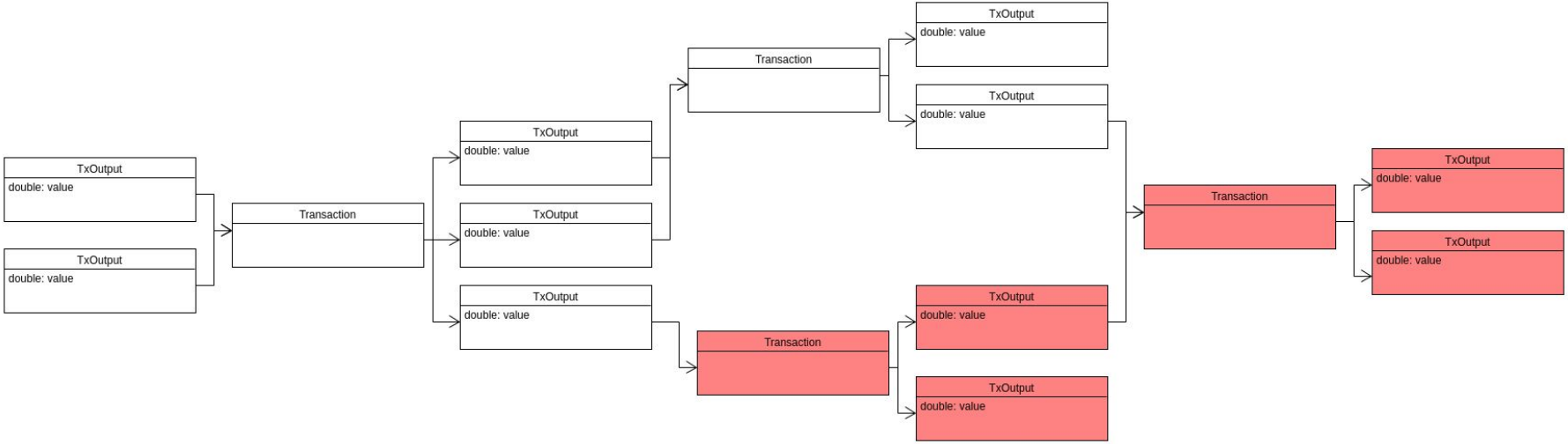
1 BTC = 100,000,000 Satoshi ~ 6,940.00$ ~ 26,364.85zł

TxOutput!

Chapter 1: Bitcoin Model - UML like a pr0; part: I

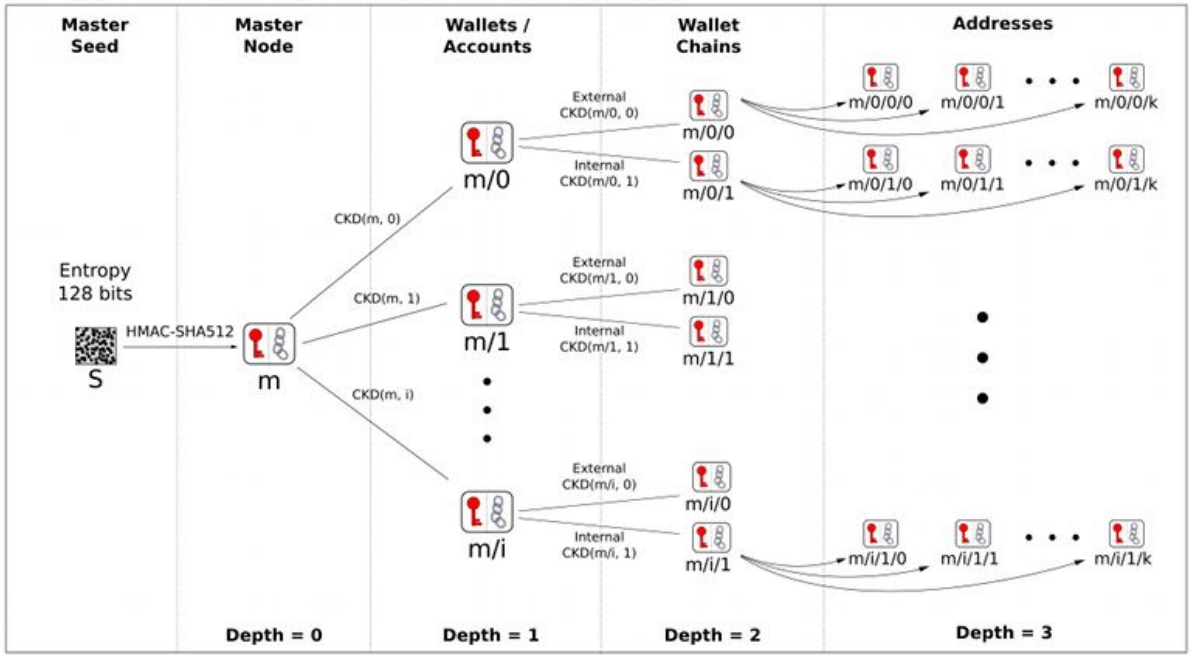


Chapter 1: Bitcoin Model - UML like a pr0; part: II



Chapter 1: Bitcoin Model - hd wallets

BIP 32 - Hierarchical Deterministic Wallets



Child Key Derivation Function ~ $CKD(x,n) = HMAC\text{-}SHA512(x_{Chain}, x_{PubKey} || n)$

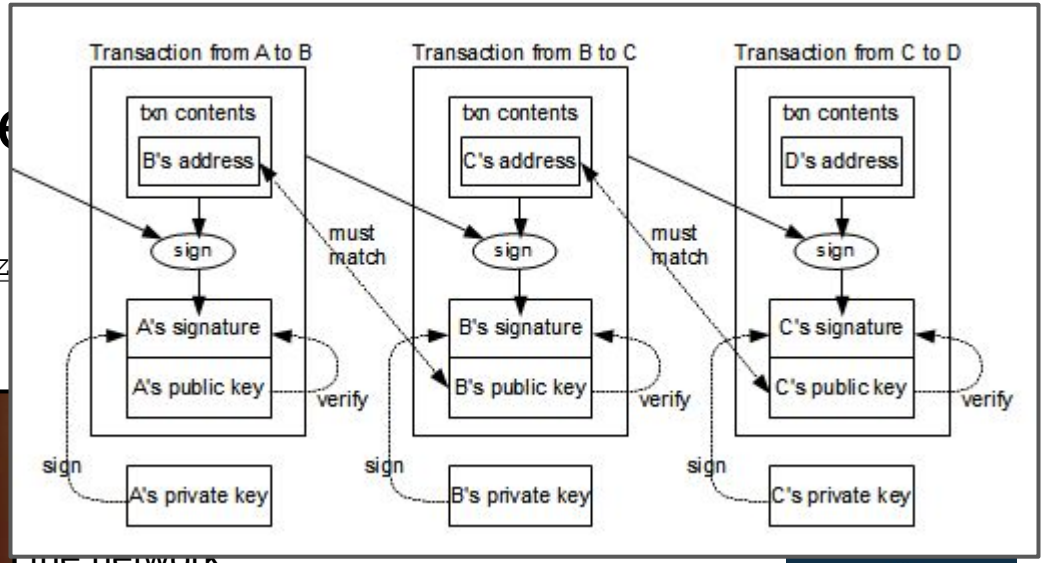
https://medium.com/@harshagoli/hd-wallets-explained-from-high-level-to-nuts-and-bolts-9a41545f5b0

https://iancoleman.io/bip39/

Chapter 1: Bitcoin Mode

1DBqpfptUMiz

- takes tx x out



the network

Owns: **1DBqpfptUM[...]**

Chapter 1: Bitcoin Model - dawn of the devs

It turns out that actually using the Bitcoin protocol is harder than I expected. As you will see, the protocol is a bit of a jumble: it uses big-endian numbers, little-endian numbers, fixed-length numbers, variable-length numbers, custom encodings, DER encoding, and a variety of cryptographic algorithms, seemingly arbitrarily. As a result, there's a lot of annoying manipulation to get data into the right format. [7]

FFFF

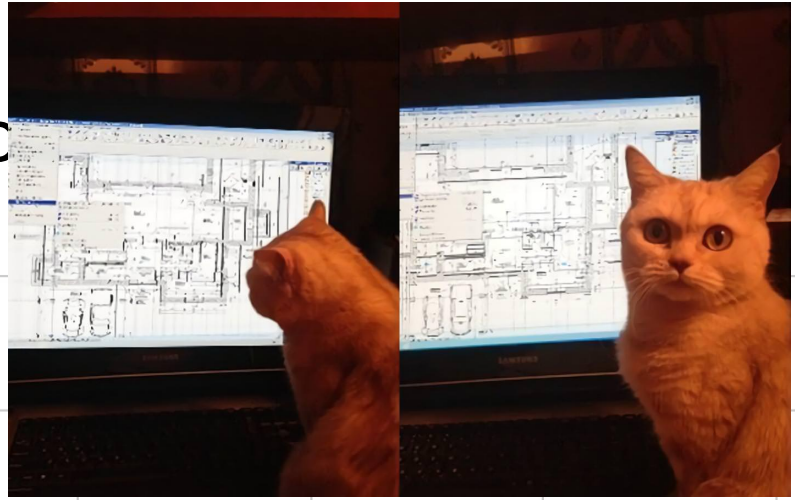
Chapter 1: Bitcoin Model - getting dirty

How to check where the coins came from?

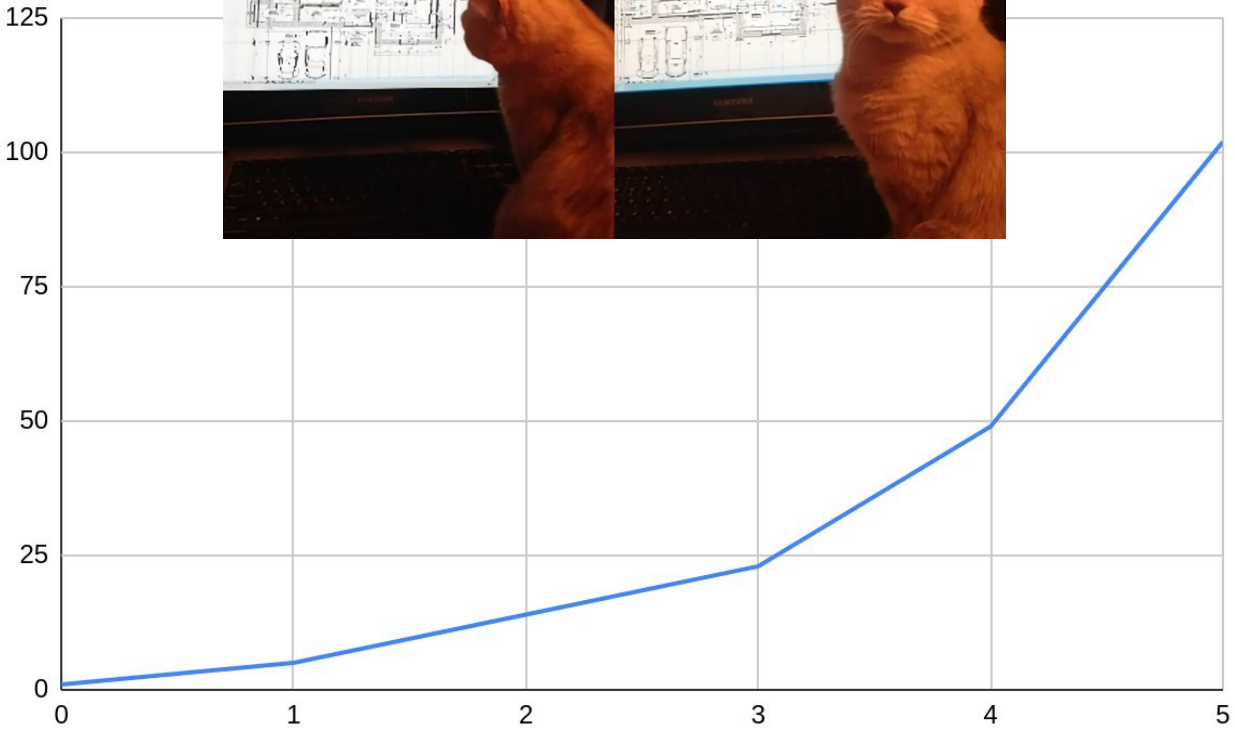
Chapter 1: Bitcoin Model - querying public apis

```
import requests
import time

desired_depth = 5

def deeper(txhash, depth):
    isFraud(txhash)
    print('Going to: ' + txhash + ' at depth: ' + str(depth))
    if depth < desired_depth:
        r = requests.get("https://api.blockcypher.com/v1/btc/main/txs/" + txhash)
        inputs  = r.json()['inputs']
        for inp in inputs:
            try:
                deeper(inp['prev_hash'], depth+1)
            except:
                return

def isFraud(txhash):
    if txhash == '3d0b134570101fbb1d61370c0c122c219d130ae3760fb072309f6f2b2eeaec7b':
        print('FRAUD!!!!')


deeper("f854aebae95150b379cc1187d848d58225f3c4157fe992bcd166f58bd5063449?token=9d56627bbe214a19a82141a0c645768d", 0)
```

Chapter 1: Bitcoin Mo is



Depth	No Of Txs
0	1
1	5
2	14
3	23
4	49
5	102



Chapter 1: Bitcoin Model - standard approach summary

- It’s slow (50reqs in 12s)
- It’s expensive (1.500.000reqs cost 650$)
- It’s not secure

Chapter 1: Bitcoin Model - alternatives for traversals

- Real time
  - Self hosted bitcoin-core node
  - PostgreSQL (with indexed addresses)
  - Mongo
  - JanusGraph

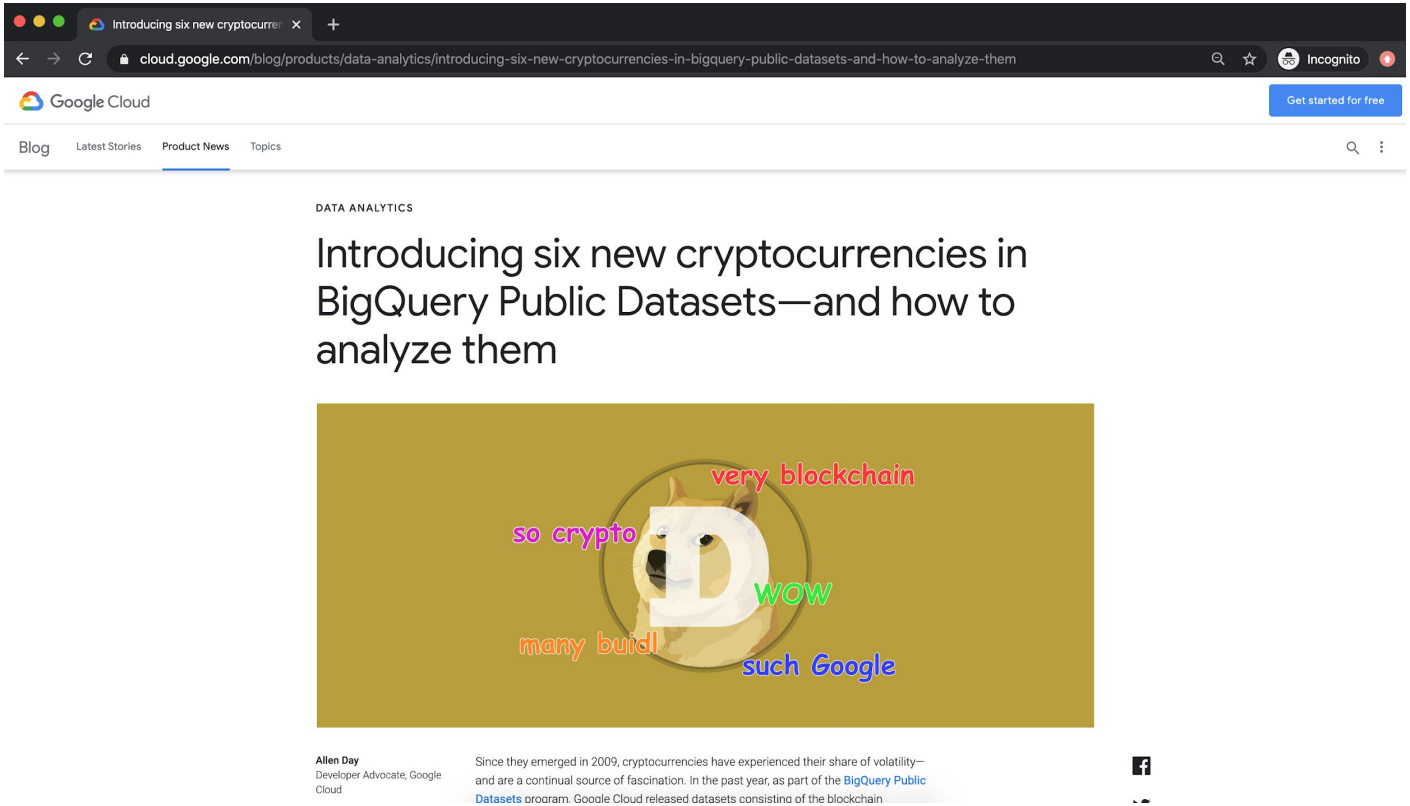
- Batch
  - Dataflow
  - **BigQuery!**

Chapter 2: BigQuery to the rescue

Chapter 2: BigQuery to the rescue

A public dataset is any dataset that is stored in BigQuery and made available to the general public through the Google Cloud Public Dataset Program. The public datasets are datasets that BigQuery hosts for you to access and integrate into your applications. Google pays for the storage of these datasets and provides public access to the data via a project. You pay only for the queries that you perform on the data. The first 1 TB per month is free, subject to query pricing details.

Chapter 3: BigQuery to the rescue



https://cloud.google.com/blog/products/data-analytics/introducing-six-new-cryptocurrencies-in-bigquery-public-datasets-and-how-to-analyze-them

Chapter 2: Bitcoin Public Dataset Schema

https://console.cloud.google.com/bigquery?project=telemetry-233915&folder&organizationId&p=bigquery-public-data&d=crypto_bitcoin&t=transactions&page=table

- Tables: blocks, **transactions**
- Views: inputs, outputs

Schema Details Preview

Field name	Type	Mode	Description
hash	STRING	REQUIRED	The hash of this transaction
size	INTEGER	NULLABLE	The size of this transaction in bytes
virtual_size	INTEGER	NULLABLE	The virtual transaction size (differs from size for witness transactions)
version	INTEGER	NULLABLE	Protocol version specified in block which contained this transaction
lock_time	INTEGER	NULLABLE	Earliest time that miners can include the transaction in their hashing of the Merkle root to attach it in the latest block of the blockchain
block_hash	STRING	REQUIRED	Hash of the block which contains this transaction
block_number	INTEGER	REQUIRED	Number of the block which contains this transaction
block_timestamp	TIMESTAMP	REQUIRED	Timestamp of the block which contains this transaction
block_timestamp_month	DATE	REQUIRED	Month of the block which contains this transaction
input_count	INTEGER	NULLABLE	The number of inputs in the transaction
output_count	INTEGER	NULLABLE	The number of outputs in the transaction
input_value	NUMERIC	NULLABLE	Total value of inputs in the transaction
output_value	NUMERIC	NULLABLE	Total value of outputs in the transaction
is_coinbase	BOOLEAN	NULLABLE	true if this transaction is a coinbase transaction
fee	NUMERIC	NULLABLE	The fee paid by this transaction
inputs	RECORD	REPEATED	Transaction inputs
inputs. **index**	INTEGER	REQUIRED	0-indexed number of an input within a transaction
inputs. **spent_transaction_hash**	STRING	NULLABLE	The hash of the transaction which contains the output that this input spends
inputs. **spent_output_index**	INTEGER	NULLABLE	The index of the output this input spends
inputs. **script_asm**	STRING	NULLABLE	Symbolic representation of the bitcoin's script language op-codes
inputs. **script_hex**	STRING	NULLABLE	Hexadecimal representation of the bitcoin's script language op-codes
inputs. **sequence**	INTEGER	NULLABLE	A number intended to allow unconfirmed time-locked transactions to be updated before being finalized; not currently used except to disable locktime in a transaction
inputs. **required_signatures**	INTEGER	NULLABLE	The number of signatures required to authorize the spent output
inputs. **type**	STRING	NULLABLE	The address type of the spent output
inputs. **addresses**	STRING	REPEATED	Addresses which own the spent output
inputs. **value**	NUMERIC	NULLABLE	The value in base currency attached to the spent output
outputs	RECORD	REPEATED	Transaction outputs
outputs. **index**	INTEGER	REQUIRED	0-indexed number of an output within a transaction used by a later transaction to refer to that specific output
outputs. **script_asm**	STRING	NULLABLE	Symbolic representation of the bitcoin's script language op-codes
outputs. **script_hex**	STRING	NULLABLE	Hexadecimal representation of the bitcoin's script language op-codes
outputs. **required_signatures**	INTEGER	NULLABLE	The number of signatures required to authorize spending of this output
outputs. **type**	STRING	NULLABLE	The address type of the output
outputs. **addresses**	STRING	REPEATED	Addresses which own this output
outputs. **value**	NUMERIC	NULLABLE	The value in base currency attached to this output

Chapter 3:

Chapter 3: BigQuery - Bitcoin’s unique IDs

```
select t.hash, count(*)
  from `bigquery-public-data.crypto_bitcoin.transactions` t
  group by t.hash
  having count(*) > 1
```

Chapter 3: BigQuery - Bitcoin’s unique IDs

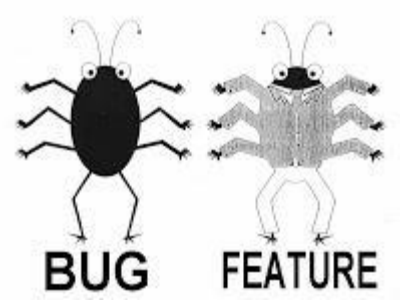
```
select t.hash, count(*)
  from `bigquery-public-data.crypto_bitcoin.transactions` t
  group by t.hash
  having count(*) > 1
```

Chapter 3: Unique IDs

```
select t.hash, count(*)
  from `bigquery-public-data.crypto_bitcoin.transactions` t
  group by t.hash
  having count(*) > 1
```

Job information | Results | JSON | Execution details

Row	hash
1	d5d27987d2a3dfc724e359870c6644b40e497bdc0589a033220fe15429d88599
2	e3bf3d07d4b0375638d5f1db5255fe07ba2c4cb067cd81b84ee974b6585fb468



Chapter 3: How much coins I have?

```
WITH
ins AS (
  SELECT SUM(outs.value) AS value FROM `bigquery-public-data.crypto_bitcoin.transactions` t
  JOIN UNNEST (t.outputs) AS outs
  JOIN UNNEST(outs.addresses) AS addrs
  WHERE addrs = '1DBqpfptUMizLDACvMVsrJFPdMtbgmZCk1'),
outs AS (
  SELECT SUM(ins.value) AS value FROM `bigquery-public-data.crypto_bitcoin.transactions` t
  JOIN UNNEST (t.inputs) AS ins
  JOIN UNNEST(ins.addresses) AS addrs
  WHERE addrs = '1DBqpfptUMizLDACvMVsrJFPdMtbgmZCk1')
SELECT
  (ins.value - outs.value)/100000000.0*7445.87
FROM
  ins,outs
```

Chapter 3: How much coins I have?

```
WITH
ins AS (
  SELECT SUM(outs.value) AS value FROM `bigquery-public-data.crypto_bitcoin.transactions` t
  JOIN UNNEST (t.outputs) AS outs
  JOIN UNNEST(outs.addresses) AS addrs
  WHERE addrs = '1DBqpfptUMizLDACvMVsrJFPdMtbgmZCk1'),
outs AS (
  SELECT SUM(ins.value) AS value FROM `bigquery-pub
  JOIN UNNEST (t.inputs) AS ins
  JOIN UNNEST(ins.addresses) AS addrs
  WHERE addrs = '1DBqpfptUMizLDACvMVsrJFPdMtbgmZ
SELECT
  (ins.value - outs.value)/100000000.0*7445.87
FROM
  ins,outs
```

Row	f0_
1	20.978738725

Chapter 3: How much coins I have?

```
WITH
active_addrs as (
  SELECT distinct(addrs) AS addr FROM `bigquery-public-data.crypto_bitcoin.transactions` t
  JOIN UNNEST (t.outputs) AS outs
  JOIN UNNEST(outs.addresses) AS addrs where t.block_timestamp_month = '2019-12-01' ),
ins AS (
  SELECT addrs, SUM(outs.value) AS value FROM `bigquery-public-data.crypto_bitcoin.transactions` t
  JOIN UNNEST (t.outputs) AS outs
  JOIN UNNEST(outs.addresses) AS addrs
  group by addrs  ),
outs AS (
  SELECT addrs, SUM(ins.value) AS value FROM `bigquery-public-data.crypto_bitcoin.transactions` t
  JOIN UNNEST (t.inputs) AS ins
  JOIN UNNEST(ins.addresses) AS addrs
  group by addrs )
SELECT
  active_addrs.addr, (ins.value - outs.value)/100000000.0*7445.87 as val
FROM
  active_addrs join ins on ( active_addrs.addr = ins.addrs )
  join outs on (active_addrs.addr = outs.addrs) order by val desc
```

Row	addr	val
1	35hK24tcLEWcgNA4JxpvbkNkoAcDGqQPsP	1902435728.968829836
2	3JZq4atUahhuA9rLhXLMhhTo133J9rF97j	933995195.286528686
3	3CgKHXR17eh2xCj2RGnHJHTDjPpqaNDgyT	832881958.685483666
4	3M6UcBNGZAW1HRjiFDMRcY5aXFrQ4F9E1y	667777685.169873268
5	3D2oetdNuZUqQHPJmcMDDHYoqkyNVsFk9r	572536905.077642759
6	34xp4vRoCGJym3xR7yCVPFHoCNxv4Twseo	545495104.140982197
7	1HQ3Go3ggs8pFnXuHVHRytPCq5fGG8Hbhx	516520976.980904968
8	385cR5DM96n1HvBDMzLHPYcw89fZAXULJP	488701521.174904326
9	38UmuUqPCrFmQo4khkomQwZ4VbY2nZMJ67	406167510.858920128
10	3FDWNdFxrJxXoF4hWXF4dkc1FFxsNyKAHh	403286288.515758849

Chapter 3: Am I the bad guy?

```
haystack as (
  SELECT * FROM `bigquery-public-data.crypto_bitcoin.transactions` t
  WHERE t.block_timestamp_month >= '2018-01-01' ),
```

```
my_addresses as (
  SELECT distinct addrs AS addr FROM `bigquery-public-data.crypto_bitcoin.transactions` t
  JOIN UNNEST (t.outputs) AS outs
  JOIN UNNEST(outs.addresses) AS addrs WHERE t.block_timestamp_month = '2020-01-01'
```

```
received_transactions as (
  SELECT distinct h.hash, addrs FROM haystack h
  JOIN unnest(h.outputs) AS outs JOIN unnest(outs.addresses) AS addrs
  RIGHT JOIN my_addresses ON addrs = my_addresses.addr LIMIT 10000),
```

Chapter 3: Am I the bad guy?

```
adjacency_list AS (
  SELECT h.hash AS root, inputs.spent_transaction_hash AS prev from haystack h
  JOIN unnest(h.inputs) AS inputs),
```

```
graph AS (
  SELECT l0.root AS l0, l1.root AS l1, l2.root AS l2, l3.root AS l3, l4.root AS l4
    FROM adjacency_list l0
    JOIN adjacency_list l1 ON l0.prev = l1.root
    JOIN adjacency_list l2 ON l1.prev = l2.root
    JOIN adjacency_list l3 ON l2.prev = l3.root
    JOIN adjacency_list l4 ON l3.prev = l4.root)
```

```
SELECT rt.addrs, g.* FROM received_transactions rt JOIN graph g ON (rt.hash = g.l0 )
  WHERE
    g.l0 IN (SELECT * FROM potential_fraud ) OR
    g.l1 IN (SELECT * FROM potential_fraud ) OR
    g.l2 IN (SELECT * FROM potential_fraud ) OR
    g.l3 IN (SELECT * FROM potential_fraud ) OR
    g.l4 IN (SELECT * FROM potential_fraud )
```

Row	addrs	l0	l1	l2	l3	l4
1	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	df5339e7d03a4653a3b43278f29a1d779c628f276a66dd5df6702b7f2e6a03d	d640e7ea82d16a4711fcf3ca878c09e5055ff3ec6990f19d3b8adb448183f5f11	042f3409fce0070addaf1bd57d2ac501f1aa301fada9a6ae92f6ee03447255b6	339fe0979f13b5dca354bc4f3daea8d6399f86ba36357c0e491d4494e6bf6f5a
2	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	43e3ac5c7e82af1700e8aabb93374ec26d9acdfac8981505eee0bf29505f6b08	da5cb64abbe793a215171657f250821321b4bdb4ad2dde083e62d71f39cc728e	dc3a32950dcb0e353a6c5ce7a10b4de3ec33078ca0d844caaa6f45639ef1145e	6071b056d05d2a82223216668580413fddb4cd95fcab204ebfafa48a9df87fc9	f80dfef7be86b14ccf4e18087fd49cba75cb874ee9485d7de1a02f0b944e63a1
3	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	43e3ac5c7e82af1700e8aabb93374ec26d9acdfac8981505eee0bf29505f6b08	da5cb64abbe793a215171657f250821321b4bdb4ad2dde083e62d71f39cc728e	dc3a32950dcb0e353a6c5ce7a10b4de3ec33078ca0d844caaa6f45639ef1145e	6071b056d05d2a82223216668580413fddb4cd95fcab204ebfafa48a9df87fc9	f80dfef7be86b14ccf4e18087fd49cba75cb874ee9485d7de1a02f0b944e63a1
4	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	7d9a8b0f387e24e5d3f83e3d29a3b5f11eabb52c938d4f0f8f050f22dffcac923d	a496a53fae06ee985c4f1f554be6edc3ab225d396915756b02a3c6066264c0d7	49bcae579891b90df5b6478cf5b3eea46df54ea28e384659fe776ce51dc9c40b	d1ec070c272d240ad879fd9bb4e0fde13e38842d624e7a1d7a1f7194d3fc34df
5	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	aa8d948a70f66f98ba01ba347dc828f172f0076da8a0f911e3a25441ffe985b	99d318ccc26e2b822b5bc963dfde5b568daf443f9ed1b5f3e57866e500b
6	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	a30f144f79db7a734ba4adac61fd2a213b4f7eb0091f499874c9fa84b984e583	5273288bfa19e32b5a91c878563629fe83cc96fe0d0393f313c37d4a943a0f1b	92a2ca92708e0fceb2633791a14348717c6b1ee2501ac550203df8a6eefdb245	b4f3f864c3130611b3aeb522604a2cd50def69d8b99bfcd59037b49577ff973a	613845747a13105491109b0aaddb1c3e33ba63785138c34202cbca86d6f50d69
7	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	b7e0b2974d1906b9ef42d4ee9a314ac0217792be9bc22f4212e621afe61da43d	fef37382a21af70655abb612738d25375c03d5994c4f5ebe4d8de004b147beffe	02ce7a4054dd2a1f8e8ab9040032fb35eaddca9cf674fc37bfe9bf69a89d233d	528ef4af6021f0b2877a9cc36a22da73cf965a3b7dfa28908421914b3fa01030
8	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	9ab01f387e24e5d3f83e3d29a3b5f11eabb52c938d4f0f8f050f22dffcac923d	a496a53fae06ee985c4f1f554be6edc3ab225d396915756b02a3c6066264c0d7	49bcae579891b90df5b6478cf5b3eea46df54ea28e384659fe776ce51dc9c40b	d1ec070c272d240ad879fd9bb4e0fde13e38842d624e7a1d7a1f7194d3fc34df
9	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	aa8d948a70f66f98ba01ba347dc828f172f0076da8a0f911e3a25441ffe985b	99d318ccc26e2b822b5bc963dfde5b568daf443f9ed1b5f3e57866e500b
10	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	0759371c2ffabe3bd688f914f8285d5f01429202b42e22a27faabc4aeeff973d	2848fa3c69f82bc3e818c9709edbf3c367e6ee9c566ec3d07bd3d6fec5871cd5	6fcedca461d4a2f8d3815089b95e305dd8acf61bb17bcdeb233d853ee12d7fe1	2851eb2999c2a35c268de8f94ba203d50e8eff90a92961b56910496 1b4f52da5
11	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	d6950bc43078f95c7513e93cb7de4a986dfb1b8185ebd4820f934e1b9644e42	f59a3506182fa95b221425dbf8bdbab1fc3da3aaa9252707d6ec8154e1469785	5b6838512 6aece9533039c1d346251e877d15219c06800fa22208beb241a9448	9b45f8bf14687c0c1ab36a8054172fdd4a0eeb5ec13231979159d99f79710a54
12	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	d7fc53f271618dc6b64c3a4c373dfc0e5891c214591feb51ffcf760461f7973d	0f1334029eebbd1ff3ad1989a46fd0ad0976cd53c742de138650397291f10977	a54f3b4d44adeb1df9c01a01d8b78bba61cc12e3adf2afefe53bd81eb6f6bb544	cca23a80d86dc50532422d0e280cf820a529f92b6c5c3a217c91f8de6c26a95c
13	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	0869c3da01984db0aaa25347d1ae37148fe8152ae266388773a25f10daaa903d	ce7526db2cca8619612c189327849fbd64aa0d348cee70f449243cfe10fdee2f	6217f7586b98714b8eab47b5229a59cd62bbe05ff3669984801275 68a45e51c9	e47b6f0136f8bb6ed16495c49cbeea4d79e66dbce8971cc09382e7ea6b70dd4c
14	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	0869c3da01984db0aaa25347d1ae37148fe8152ae266388773a25f10daaa903d	ce7526db2cca8619612c189327849fbd64aa0d348cee70f449243cfe10fdee2f	6217f7586b98714b8eab47b5229a59cd62bbe05ff3669984801275 68a45e51c9	e47b6f0136f8bb6ed16495c49cbeea4d79e66dbce8971cc09382e7ea6b70dd4c
15	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	7d4faebe5387e8c84798a0795a960f3e6e43d28cb898a0f5f0e76b840d8d5842	a35032b1eb3daa131819b07b40bab2c50b3e031d3329a7a57b7e2613354c304c	b6c7e31feb7fc9ca74eb82743b425839c63cd6b78bd6e452bacd9dbff111fef2a	db0373f6ac953555415a742817d5a17980ae2f2e60f1db8ceaf159df26c25d58
16	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	2e69bef594bdf7dce9df307df586e26e096f83f344beec2d0dd9f38eb641e50	23f99aaa6030a1546295 4e78ee5ae96599df4eb73fcd59ab9fc81da8366c14c
17	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	7d4faebe5387e8c84798a0795a960f3e6e43d28cb898a0f5f0e76b840d8d5842	58bd924a210f055d1d8309339040dc6e80552ffe3754879e9b5dac179752434c	cb1ade9bf49b66a295bc3acc12e1e614ce820b68aead3b2355b165414102f21f	2c53af24649ee8f7553aa21ce9d1ff4524496 8ba2eecdf230d7c9567cea2ef32
18	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	2e69bef594bdf7dce9df307df586e26e096f83f344beec2d0dd9f38eb641e50	23f99aaa6030a1546295 4e78ee5ae96599df4eb73fcd59ab9fc81da8366c14c
19	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	7d4faebe5387e8c84798a0795a960f3e6e43d28cb898a0f5f0e76b840d8d5842	6712097e54a0d2bef08c4e1048b2bd0e2bc4bc8c59a5efa3d339e78ee4d4447c	79df48722a8a3140223ac294198b924aefa007bd03b46e882e9d215dd4c5ee69	f298ba7fe4e58589dec435e8d544f8ca27df2f79c71e3d6f1f9d6c26a95c
20	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	5a45e5b1e7db578f0410133098817 73d1fc0bc42c819a0c70b76f00f6932ec9	2ebb7cb8d7fcc9c99975c9dd0f8d1af05771df2ab12e3a82ebe6fc1d43b65079	b6fe1963af1719183f938df52a015608240d597f91ef6016906a83ecacca5976	87d4f890d556b2d180d81fec5205242c640c229566057c4dabc8b577b7356ae7	31a59ceaa058aba1ba428a240ccc7b8111ba0fb9d9b217b8817b658913f434d
21	1G47mSr3oANXMafVrR8UC4pzV7FEAzo3r9	cd6070cdb6aaf32cd9f9a417d9d0a810ee6e183cbfa396554177597458463e77	9cb9e3ff7b81f7780e3bb2df6b762f1c0894bd5f87188c0a3e9b0c1d03bb8f19	4b73d458b5e9c5f88090673c587764045988b3f19b508a4da9f92b3188dff0579	988e3e3f0736fa80fdab41b4ef9169060233346 09a3917e5136c4d39508f52dc	66517ae012ed5206d7f7b65e062d6a74f49f0d1f11ca28854f2c7217e75e94eb1
22	1G47mSr3oANXMafVrR8UC4pzV7FEAzo3r9	2a97c5689edab6f988bac22132fb45fc447b255bc433ed762edce4bb0f81e1d0	8d75abaaee7cd71d7c67b0511e9e0d11ef7decfe430c243c06861f34f015ac28	01606085c16e47937b028e1160118ba8a35f79b1117026a5bc28327c599f22e3	503ad37e152e5e1c0de379d1f827649157e2000094d835a8a55f42b54fa56e50	dcbb45876996c1a21f40ea1f19ba2419bab7308bbca1fa1f7b42d8665b0e505e
23	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	7ff604b9992a3637915b32a930a042fb4fdc6121cd925570f1a2a20120c5ae87	b4d0c68cf3a7dbb959cafbf90ab614f48f3d06083079bd2ec4d5a8a199c77aa8
24	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	a7dfdf02c01ee52fccd32a4a88657c3f442af086e6900717ddb45f5fde31a03d	3b73ba9b489f2c9262af004521cfac8c3bfbb554abb8146279079051 6ed40f6bc	b1efc9686cd896de619218dc20a18b0f49036d0c5d391f6858cede10116d0d8c	aea13af2a53e320b5471b78534e6ac3508bc363d36cbf898b13b2bd15e2682aa
25	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	e53fb5c9f0e42723b044373eb4beab25ff9d470e2a4463da096a6980b27bc226	d6d442ac80ce6f05648a5fd2aaccf4d2328ac557259504 89eec89a78c10417c	f71954bda152d45bacafefe3978a209373 0c85648a5e5ea33c468d70306297d6	399576711224f77a6398bde230db90956641d289e3ec2d4c25881de24a0493098	33465b5873d9340a968dfe000075c2402b93e698e1d55c8aa136f72811ee6f5
26	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	0869c3da01984db0aaa25347d1ae37148fe8152ae266388773a25f10daaa903d	ce7526db2cca8619612c189327849fbd64aa0d348cee70f449243cfe10fdee2f	6217f7586b98714b8eab47b5229a59cd62bbe05ff3669984801275 68a45e51c9	bb115b47a473d748e3cc756cc1fe14c9293c3d316902b4e72f4b916b254a967d
27	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	5a45e5b1e7db578f0410133098817 73d1fc0bc42c819a0c70b76f00f6932ec9	2ebb7cb8d7fcc9c99975c9dd0f8d1af05771df2ab12e3a82ebe6fc1d43b65079	9db591cb1687c98c4390ea0d57bf7c59ab230ec2368df794663402cecf1757f6	539c18067729fbaf042bfb3b09fc48e541edc0e9b412ed8d433cee77634f1d98	b7a0eac58402f61361a2f41322243cbdcdbe688aee69e6ed7ccaaebc73b593f40
28	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	6bb373188fcb70c4309bb79803bc743bded1d5731f22041b34ffddb21f6bb364c	aeee9dafd3b38537337386d5a5f363fe02187e4a0b9f7b6f4093023c7f89	46635a749d03f1a735ffebe8c4a4c29c1c70b31a10cc620aee6f1ef8622023bf
29	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	7d4faebe5387e8c84798a0795a960f3e6e43d28cb898a0f5f0e76b840d8d5842	c0060754ecc87aafd339463691317c5128bd4a5fa09e8fccaea13e646ac93c4c	e830b5e0ac8daa80eec12c485976859d3afeb92e4410f2fbe87b1454e67043a8	2315fa43ed4e5bf100efdfb7fbe53503405740f8f411c981c6b42656f6d96439
30	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	29557f59703e260d1fd398a4d95d4adfb950c2edeb1b19534ad8e0d38daa9702	56b269b71bfc4031b15b2aa6ee200872283a0bacb6e5782eb3f3e7ca648ebe0a
31	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	3edb652dfdeae6cb62802607e51d60d3b8b39703f9bf3452535f731dba7afe15	e6bb5933c08766c00dab34dadf7d05bbb56522b87eb61f69033c73efc4592d4f
32	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	bcc517ca81b6272e80a6a131d25a9047c3ae99295211d8aef84c8edb7b9e7b53	f863b4b517b753522bf3d56e8ef657c6d8df8d2d7bd8e7b79d53e747894b8dd2
33	1G47mSr3oANXMafVrR8UC4pzV7FEAzo3r9	d0b7c4a5b53ef614d5ef3bae635bc7f27ffe89c658d3db3fb268646395438632	27ec629eb8b33dbddeadaf5a8ed5134d0f61dffd45cec6ff2f1dcbebdb564b83	68ec26fa92d234ad6f8a7031d10e3203208504e316e945bcbf124bfddc996074	22ee7581575f8aa0801 1e85e6451d21f45b108586473818b4b627798e8c9de65	8321507d777ec643a50b908e5e93d1533d026dd01dbed34eb9cb5fc9bdd1f20d
34	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	d6950bc43078f95c7513e93cb7de4a986dfb1b8185ebd4820f934e1b9644e42	c7d5b934e227e01b004ad16d3c91d7e44e7c5ca0ecb5c5e1e3ab2738eb8c8885	0016bde044a2c1f6ce6ec6f913fc58289be916af0479173 90c0542fe5fc18dda	f16b6246d2617dc0770357fbc522d5d79c24d84de05c64f6e36f31d36a94c727
35	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	e34a2df84212a64fbe2c32a1cae0fcc203814f1c7b7c2cbc13d0f1c7c9e8a23d	478df6b692bd26b5715839a09de392e690faa0e16f01275ec763b7b0857cfb	fb6ed57fbaa90961772cd7b03e84aad26ecd15639973d96dc5796cd388af2bcc	b3bf03f5bb57cd0e5e0ee80af42edc23377d6b48
36	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	7d4faebe5387e8c84798a0795a960f3e6e43d28cb898a0f5f0e76b840d8d5842	c0060754ecc87aafd339463691317c5128bd4a5fa09e8fccaea13e646ac93c4c	e830b5e0ac8daa80eec12c485976859d3afeb92e4410f2fbe87b1454e67043a8	2efc73d18f3ea9f989638f1a254063d06943ba38f0c618dd6cad2f60217e3b78
37	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	0869c3da01984db0aaa25347d1ae37148fe8152ae266388773a25f10daaa903d	ce7526db2cca8619612c189327849fbd64aa0d348cee70f449243cfe10fdee2f	6217f7586b98714b8eab47b5229a59cd62bbe05ff3669984801275 68a45e51c9	bb115b47a473d748e3cc756cc1fe14c9293c3d316902b4e72f4b916b254a967d
38	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	0869c3da01984db0aaa25347d1ae37148fe8152ae266388773a25f10daaa903d	ce7526db2cca8619612c189327849fbd64aa0d348cee70f449243cfe10fdee2f	6217f7586b98714b8eab47b5229a59cd62bbe05ff3669984801275 68a45e51c9	bb115b47a473d748e3cc756cc1fe14c9293c3d316902b4e72f4b916b254a967d
39	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	d6950bc43078f95c7513e93cb7de4a986dfb1b8185ebd4820f934e1b9644e42	0bd0ef507717a1b44c3a407ab422bb5fcd09990e41f0685eee9bda5c35db8c85	af06031ff9dedebe2fc7ed340b2c05300789676 3f02da5ed09f77756ea930144	a7de8e952b14f512783a4674dd4815f39b9f063bc64309a4443637b6aafee6b6
40	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	7d4faebe5387e8c84798a0795a960f3e6e43d28cb898a0f5f0e76b840d8d5842	472c3f7e493d9ec16194eb89662028ff24c3df071bf43b2d621cac8014f8394c	f248b048580a58b5adee982e3cce98ff1b16ee34ae983cf68502c0768a9e1025	bfc9adb17f9516f308f0a5ca8e12a42325b0d92c4df978426169 7b45f1c43a2e
41	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	57d028c77c97f01f4fdedad18c1000be5464988cacba8002b0cc777c31a9ab6442	43ca1bdb9800624469 11e72da489c797f3b4d738076e204c76a6cd80220dc0c42e	27ea32d8fe5e51673891a9a8b7a74432cd2ee20249255f83e9f78c30199bb4369	b4f27cae1f2f0d1f1f2446da107e568ad66eb397c7e
42	1FH6NBSJyuAEML3oCMkizn1N4U1vX4Lrv9	deca2b7625669b37a2608c95ccdae1de5a906a72bcab945 14b64d6e54861e05b	e7614451d71642e9ab369999d8039679fc96d345824367 3bdeb90a8a8a38f9d8	0f87d58f2164ae2d1d2fbb0232f76d94566e8a5a88a78c40571e9bedddae3731	d2334e2f6e5d67d1b2194043 9d3a1ea7bc3fd97631e0cb053d4d212924fd69de	f7fc232d02a1d2a24154087d7c883dcff68b82b87586755f7ff2280bf909db13
43	1G47mSr3oANXMafVrR8UC4pzV7FEAzo3r9	2aedb3ee766539aaa1b417e234c6bb55024056e686e2c165733cbd05e421af35	6bc5aa30e055dfff96520dc558d947092baba3bcd3ea3803b504368efa6f1569	5144f58126dfea06e2962117b6dfa69d0434e55fc80aae369047af262fd87b86	7fd97d075db58e5feb2ec1363b38aac57156e6f616461d44c0ceb506a7c1aea0	9f9818fe98a8e4e0d48fd146fea8ed5b1cb9a79a9748aa8c17a31839c53da07
44	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	c8f30f87035ae5d15332e79ba4ce8a752bbf46df286f428eee0561452ad6515d	d7fc53f271618dc6b64c3a4c373dfc0e5891c214591feb51ffcf760461f7973d	0f1334029eebbd1ff3ad1989a46fd0ad0976cd53c742de138650397291f10977	a54f3b4d44adeb1df9c01a01d8b78bba61cc12e3adf2afefe53bd81eb6f6bb544	6c6433974ab92b6b1196f6816b0e39ef2ad86f53a202d9c1e13056bcf5a60b5b5
45	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	1a2038dcccbde5c5c46bc2902c0b26c8191d0b4e9d247b24b5b204a459f60a3e	d6950bc43078f95c7513e93cb7de4a986dfb1b8185ebd4820f934e1b9644e42	e5aae02b6e0877831e46abad33571dcead51b614fcb35a4845129 0530e59a85	8a8be42e7232957a30b5131adda75bac4000c472953084d502149df2c8723704	02e260c30d4937c7a4a22869058cba9a99830dec1b2268945cc5761bd2dcc1e
46	1FoWyxxPXuj4C6abqwhjDWdz6D4PZgYRjA	2b98863032e65e2174927acab23af54078bc9015cab52a43d6b4cc9cbbcdc6d	2d293480b6a6bc5fa56e927cdffe2ba6f4c808a14c22cc0976fd91fa4d366e99	0b24bdf522a90bcf46fa791a3c3b0427df54a1464eca2cfba7c5dde798d5d62b	2e69bef594bdf7dce9df307df586e26e096f83f344beec2d0dd9f38eb641e50	23f99aaa6030a1546295 4e78ee5ae96599df4eb73fcd59ab9fc81da8366c14c

Chapter 3: BigQuery graph traversal - summary

+ It’s super cheap (60Gb/query)
+ Requires no maintenance

- Dataset is updated only once per day :(
- Could be faster

Chapter 3: Ideas to improve

How to achieve deeper traversals

- Address/Transaction clustering
- Fraudulent transactions/addresses preprocessing

How to achieve real-time validation

- Key-value result storages
- Hybrid solutions

Chapter 4: Questions

Chapter 5: Thank you!

Some papers

https://www.ifca.ai/fc14/papers/fc14_submission_11.pdf

https://github.com/mikebocian/talks/tree/master/20200108-warsaw_meetup