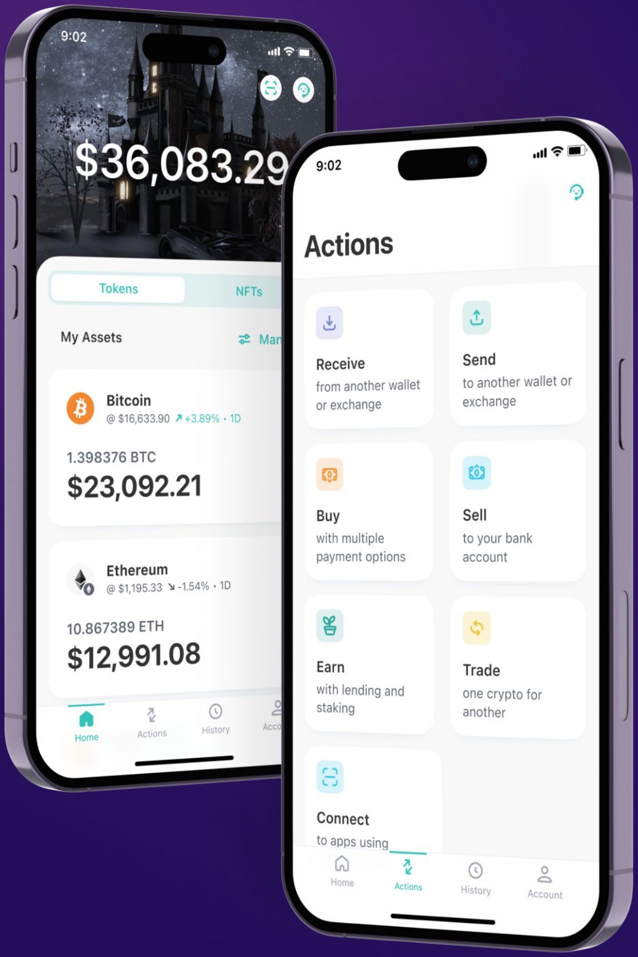


9:02

$36,083.29

Tokens NFTs

My Assets

Bitcoin
@ $16,633.90 ↗ +3.89% • 1D
1.398376 BTC
$23,092.21

Ethereum
@ $1,195.33 ↘ -1.54% • 1D
10.867389 ETH
$12,991.08

9:02

Actions

Receive
from another wallet or exchange

Send
to another wallet or exchange

Buy
with multiple payment options

Sell
to your bank account

Earn
with lending and staking

Trade
one crypto for another

Connect
to apps using

# ZenGo

# The most secure crypto wallet

# Give me some (key) space!

ZenGo

# Agenda

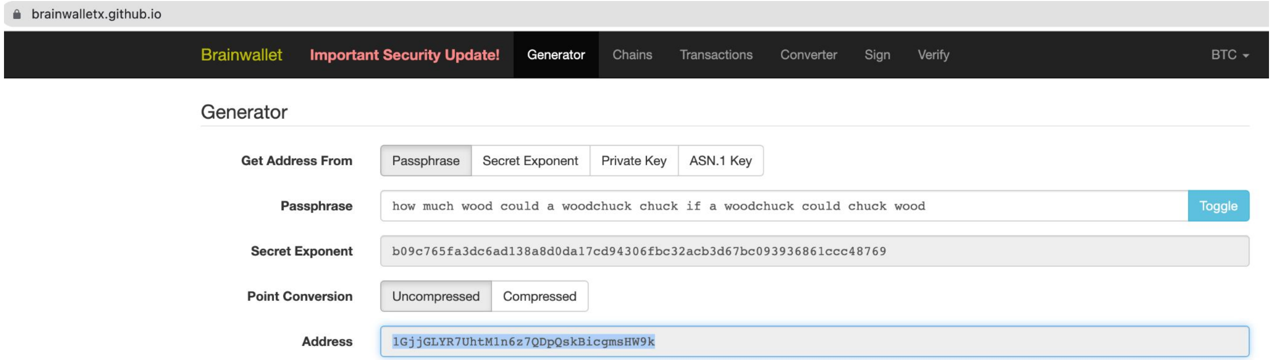
- Pre BIP-39
  - ➔ Brainwallets
- BIP-39 From randomness to Address(es)
  - ➔ Entropy, mnemonic, seed, address
- War stories
  - ➔ ZenGo: The seed saviors
  - ➔ The BTC challenge
  - ➔ Trust wallet extension
  - ➔ Profanity vanity addresses

# Brain wallets

ZenGo

# Brain wallets

- Select a passphrase
  - Example: how much wood could a woodchuck chuck if a woodchuck could chuck wood
- Hash it (SHA-256)
- This is the private key
- https://brainwalletx.github.io/



Generator

**Get Address From** [✓ Passphrase] Secret Exponent Private Key ASN.1 Key

**Passphrase** how much wood could a woodchuck chuck if a woodchuck could chuck wood

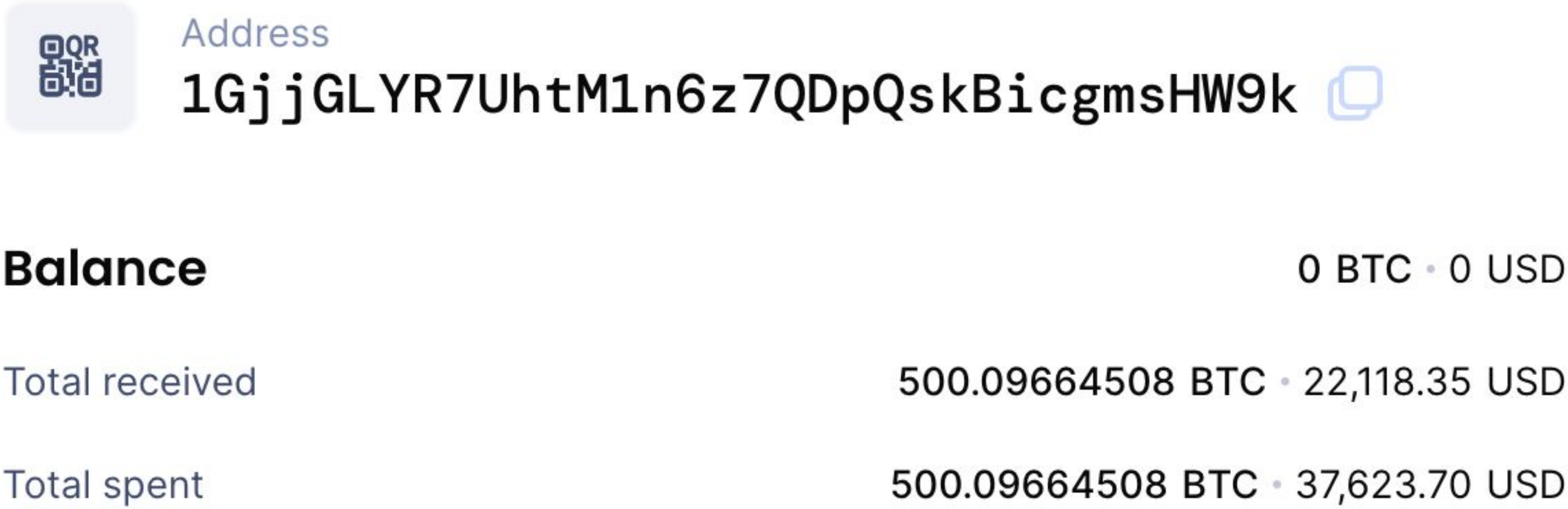
**Secret Exponent** b09c765fa3dc6ad138a8d0da17cd94306fbc32acb3d67bc093936861ccc48769

**Point Conversion** [✓ Uncompressed] Compressed

**Address** 1GjjGLYR7UhtM1n6z7QDpQskBicgmsHW9k

# What can go wrong?

- https://blockchair.com/bitcoin/address/1GjjGLYR7UhtM1n6z7QDpQskBicgmsHW9k



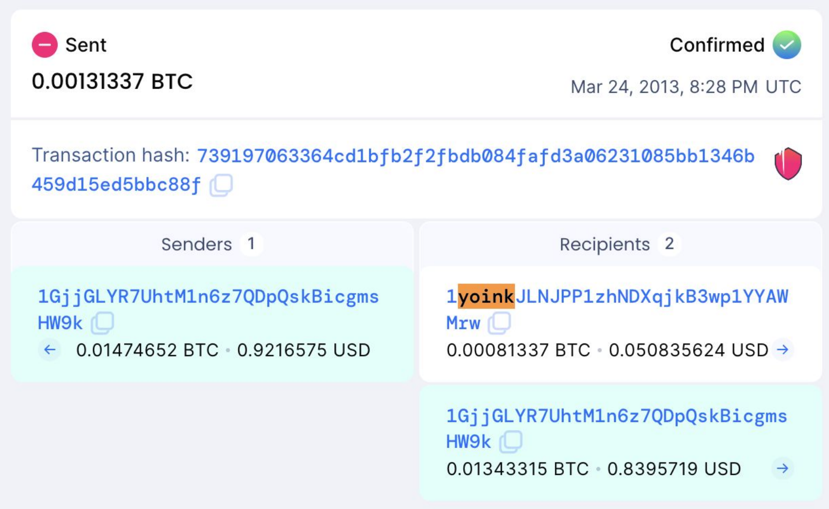
Address

1GjjGLYR7UhtM1n6z7QDpQskBicgmsHW9k

| | |
|---|---|
| **Balance** | 0 BTC · 0 USD |
| Total received | 500.09664508 BTC · 22,118.35 USD |
| Total spent | 500.09664508 BTC · 37,623.70 USD |

# Yoink!

- https://blockchair.com/bitcoin/address/1GjjGLYR7UhtM1n6z7QDpQskBicgmsHW9k



Sent
0.00131337 BTC

Confirmed ✓
Mar 24, 2013, 8:28 PM UTC

Transaction hash: 739197063364cd1bfb2f2fbdb084fafd3a06231085bb1346b459d15ed5bbc88f

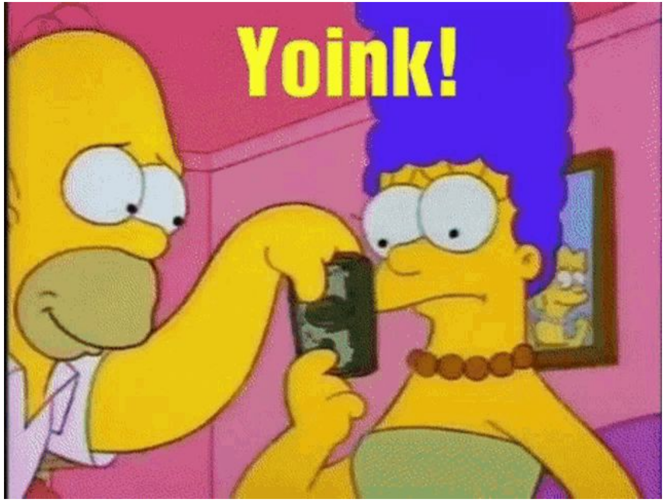
Senders 1

1GjjGLYR7UhtM1n6z7QDpQskBicgmsHW9k
← 0.01474652 BTC · 0.9216575 USD

Recipients 2

1yoinkJLNJPP1zhNDXqjkB3wp1YYAWMrw
0.00081337 BTC · 0.050835624 USD →

1GjjGLYR7UhtM1n6z7QDpQskBicgmsHW9k
0.01343315 BTC · 0.8395719 USD →



# Who is Yoink? Ryan Castellucci

- Cracking Cryptocurrency Brainwallets https://rya.nc/files/cracking_cryptocurrency_brainwallets.pdf
- Found 733 BTC
- Examples
    - “Down the Rabbit-Hole”: held about 85 BTC in July 2012
    - “The Quick Brown Fox Jumped Over The Lazy Dot”: held about 85 BTC in December 2011
    - “”: had 50BTC last week (when the original preso was presented), stolen in seconds

> *An early old-style brainwallet was created by memorization of a passphrase and converting it a private key with a hashing or key derivation algorithm (example: SHA256). That private key is then used to compute a Bitcoin address. This method was found to be very insecure and* ***should not be used****.*
> ***Humans are not a good source of entropy.***

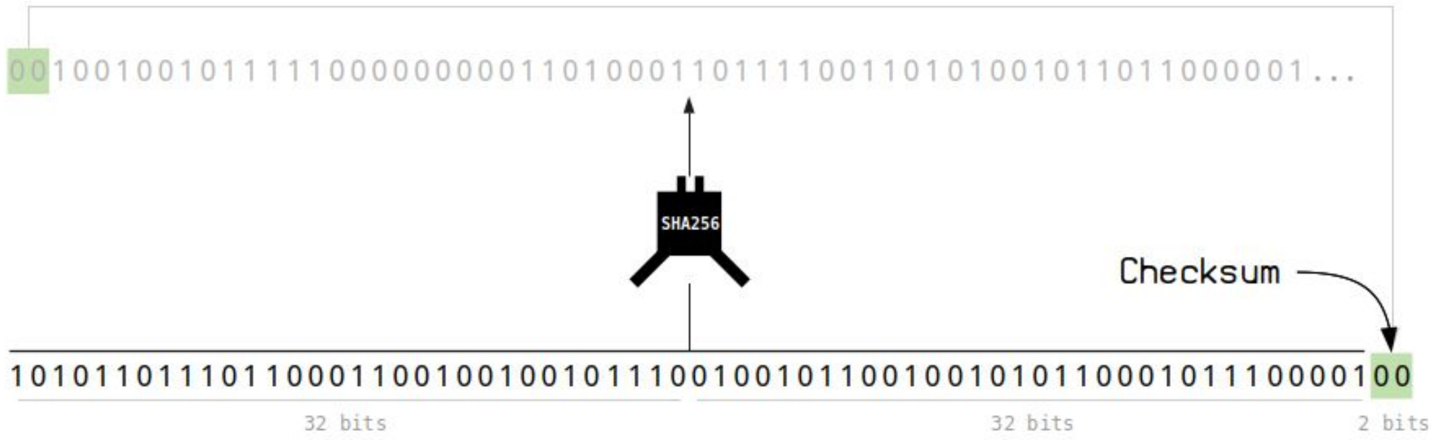
- The Bitcoin wiki

# BIP-39

# Step 1: Random ➔ Random + CheckSum

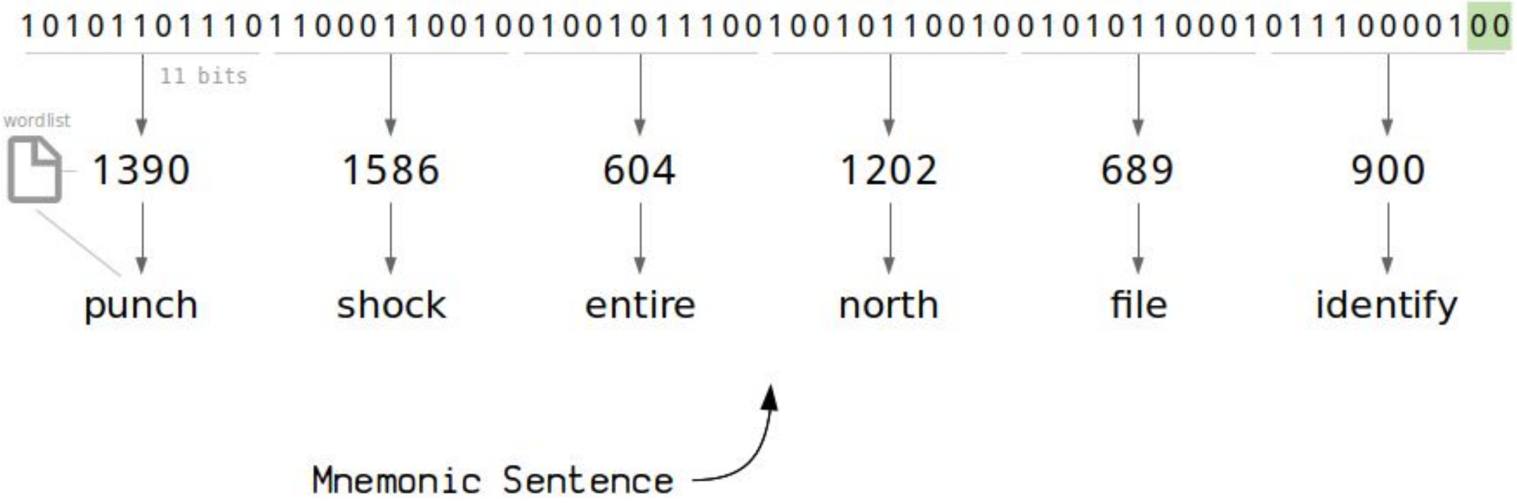
- Randomness: 128/256 bit
- Adding 1 bit of checksum for each 32bit (33 is divisible by 11)
  - 128 ➔ 132
  - 256 ➔ 264

Take 1 bit for every 32 bits of entropy.



# Step 2: Random + CheckSum → Seed Phrase

- Each group of 11 bits is assigned with a BIP-39 word
- Word list(s)
  - https://github.com/bitcoin/bips/blob/master/bip-0039/english.txt



101011011101100011001001001011100100101100100101011000101110000100

11 bits

wordlist

1390 → punch

1586 → shock

604 → entire

1202 → north

689 → file

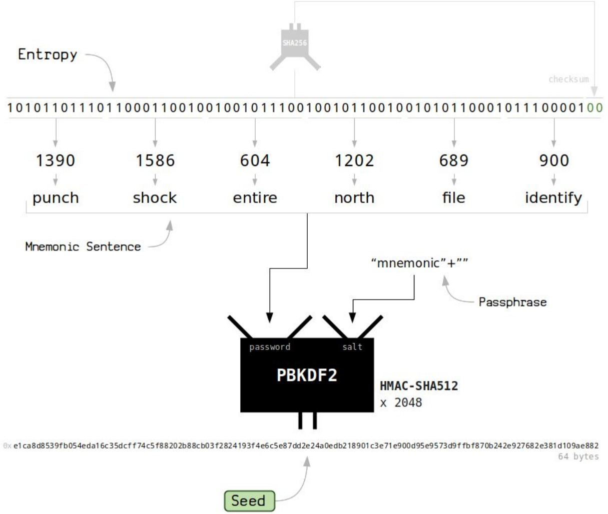
900 → identify

Mnemonic Sentence

ZenGo

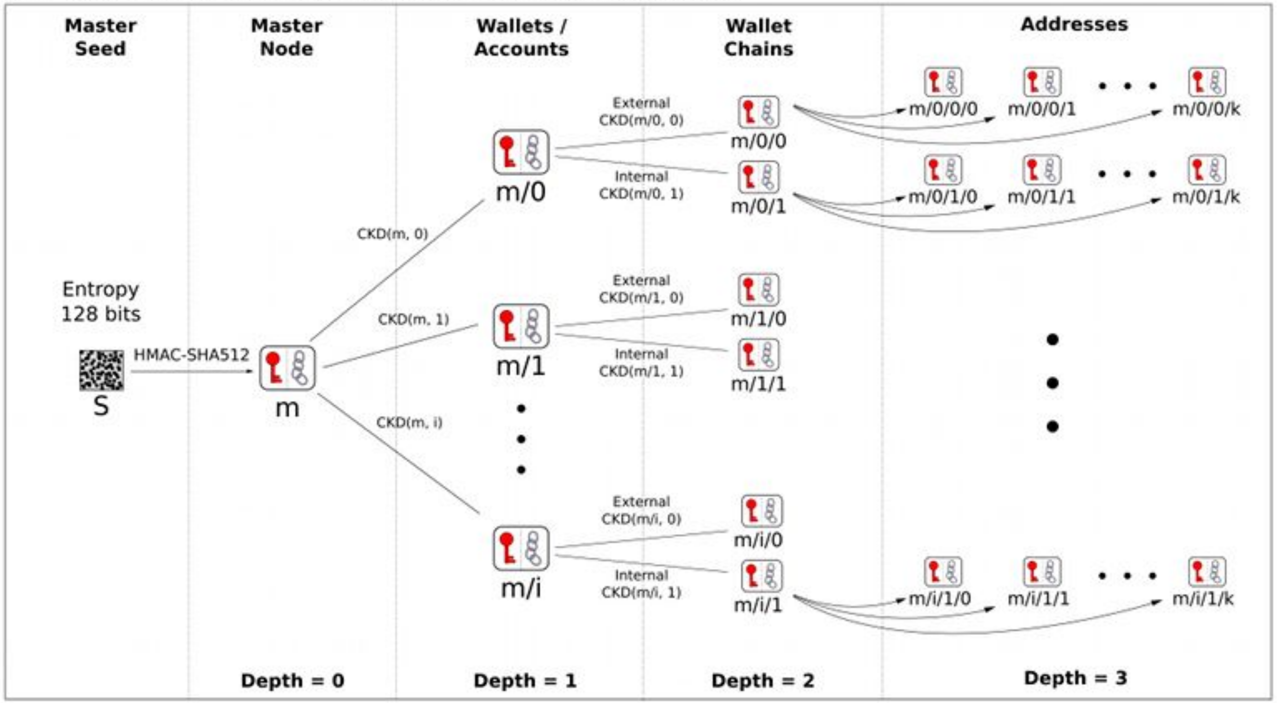
# Step 3: Seed Phrase ➔ Seed

- Key Derivation Function: PBKDF2: 2048 HMAC-SHA512
  - Adding performance “penalty” to make bruteforce harder
  - Potential passphrase addition



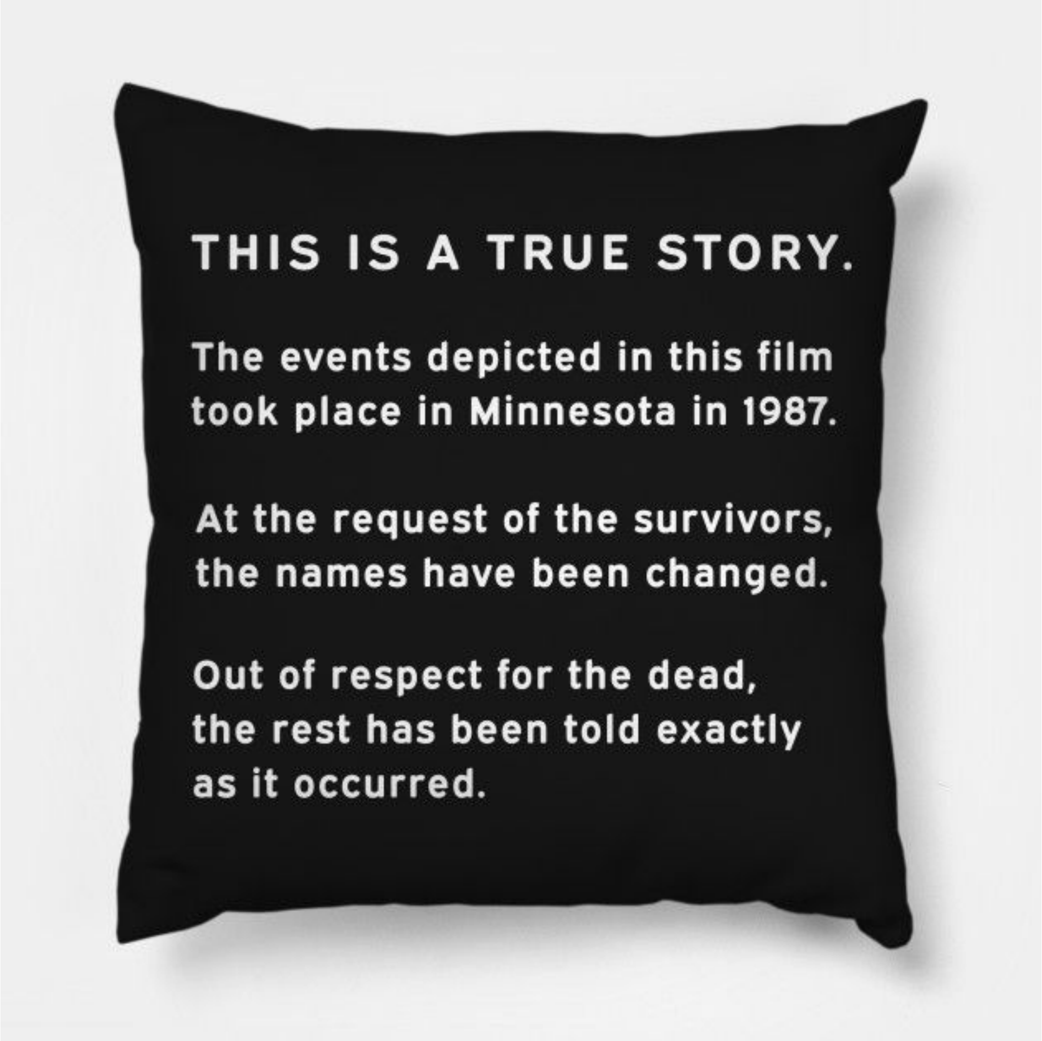
# Step 4: Seed → address(es)

BIP 32 - Hierarchical Deterministic Wallets



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

# ZenGo:
# The seed saviors



THIS IS A TRUE STORY.

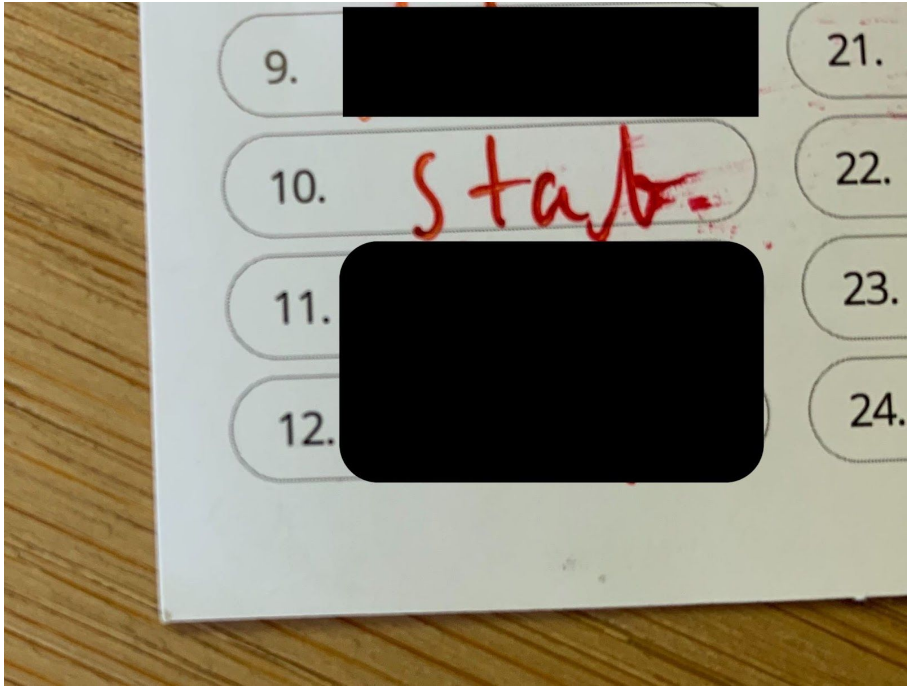
The events depicted in this film took place in Minnesota in 1987.

At the request of the survivors, the names have been changed.

Out of respect for the dead, the rest has been told exactly as it occurred.

# What was word #10?

- The story: https://zengo.com/the-wallet-seed-saviors/



9. [redacted]

10. stab

11. [redacted]

12.

21.

22.

23.

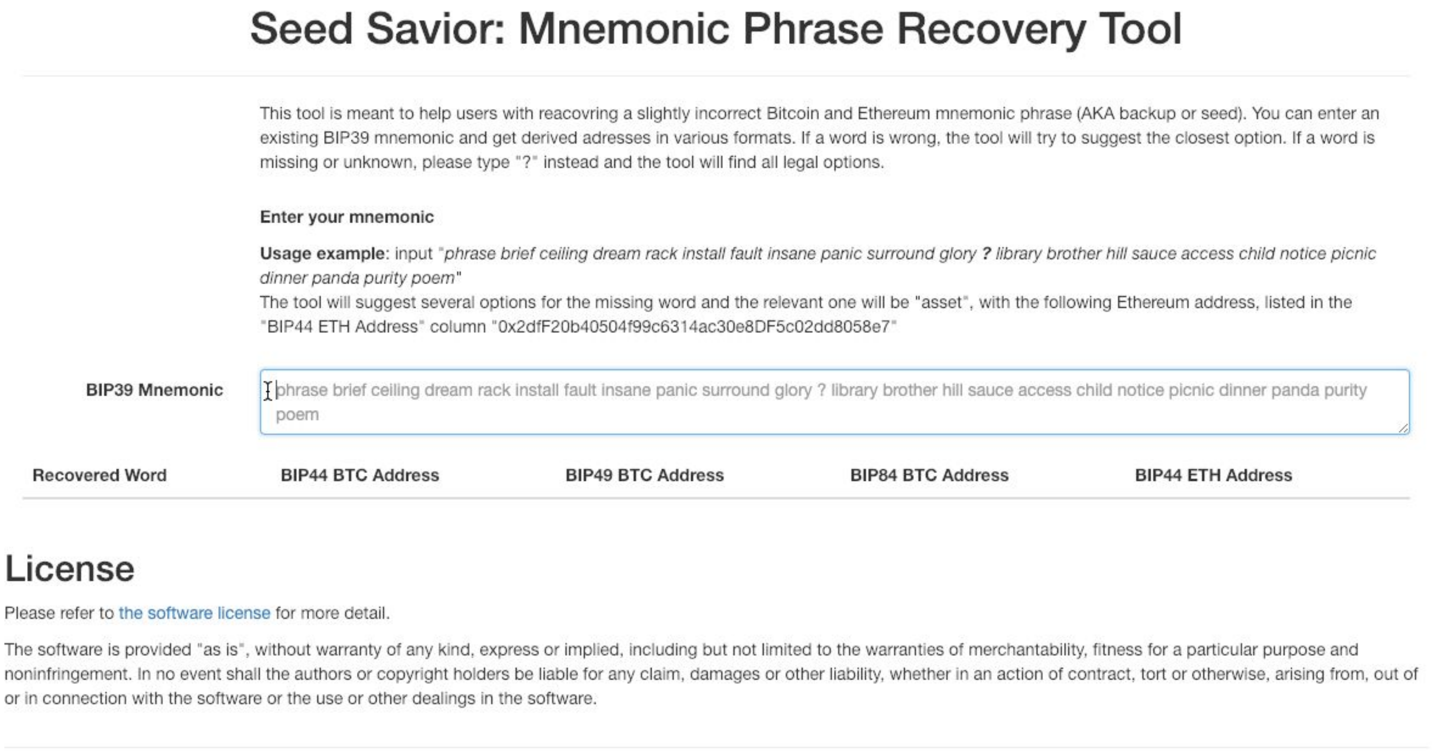
24.

# How hard is it to guess 1 word?

- Each word represents 11 bits
- 11 bits ➔ 2^11 = 2048
- 24 words ➔ 8 bits of checksum
  - So only 3 “free” bit ➔ 2^3 = 8 options
  - No need to derive an address for invalid seed phrases
- Very much doable in browser’s javascript
- [https://zengo-x.github.io/mnemonic-recovery/src/index.html](https://zengo-x.github.io/mnemonic-recovery/src/index.html)

ZenGo

# Demo



## Seed Savior: Mnemonic Phrase Recovery Tool

This tool is meant to help users with reacovring a slightly incorrect Bitcoin and Ethereum mnemonic phrase (AKA backup or seed). You can enter an existing BIP39 mnemonic and get derived adresses in various formats. If a word is wrong, the tool will try to suggest the closest option. If a word is missing or unknown, please type "?" instead and the tool will find all legal options.

**Enter your mnemonic**

**Usage example**: input "*phrase brief ceiling dream rack install fault insane panic surround glory* **?** *library brother hill sauce access child notice picnic dinner panda purity poem*"
The tool will suggest several options for the missing word and the relevant one will be "asset", with the following Ethereum address, listed in the "BIP44 ETH Address" column "0x2dfF20b40504f99c6314ac30e8DF5c02dd8058e7"

**BIP39 Mnemonic** phrase brief ceiling dream rack install fault insane panic surround glory ? library brother hill sauce access child notice picnic dinner panda purity poem

| Recovered Word | BIP44 BTC Address | BIP49 BTC Address | BIP84 BTC Address | BIP44 ETH Address |
|---|---|---|---|---|

## License

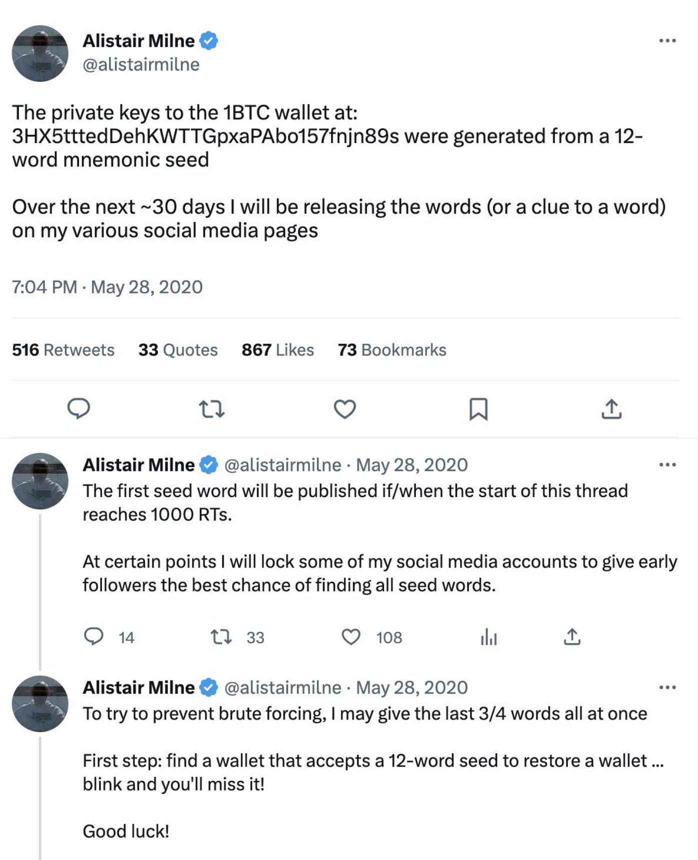
Please refer to the software license for more detail.

The software is provided "as is", without warranty of any kind, express or implied, including but not limited to the warranties of merchantability, fitness for a particular purpose and noninfringement. In no event shall the authors or copyright holders be liable for any claim, damages or other liability, whether in an action of contract, tort or otherwise, arising from, out of or in connection with the software or the use or other dealings in the software.

# The BTC challenge

ZenGo

# BTC Challenge: How it started



**Alistair Milne**
@alistairmilne

The private keys to the 1BTC wallet at: 3HX5tttedDehKWTTGpxaPAbo157fnjn89s were generated from a 12-word mnemonic seed

Over the next ~30 days I will be releasing the words (or a clue to a word) on my various social media pages

7:04 PM · May 28, 2020

**516** Retweets **33** Quotes **867** Likes **73** Bookmarks

**Alistair Milne** @alistairmilne · May 28, 2020

The first seed word will be published if/when the start of this thread reaches 1000 RTs.

At certain points I will lock some of my social media accounts to give early followers the best chance of finding all seed words.

14 33 108

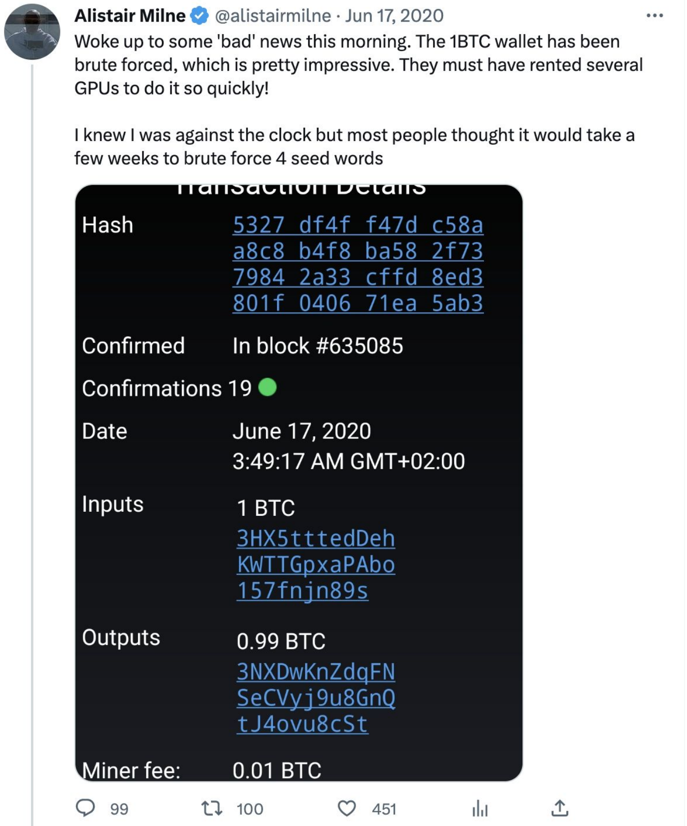
**Alistair Milne** @alistairmilne · May 28, 2020

To try to prevent brute forcing, I may give the last 3/4 words all at once

First step: find a wallet that accepts a 12-word seed to restore a wallet ... blink and you'll miss it!

Good luck!

# BTC Challenge: How it ended



**Alistair Milne** @alistairmilne · Jun 17, 2020

Woke up to some 'bad' news this morning. The 1BTC wallet has been brute forced, which is pretty impressive. They must have rented several GPUs to do it so quickly!

I knew I was against the clock but most people thought it would take a few weeks to brute force 4 seed words

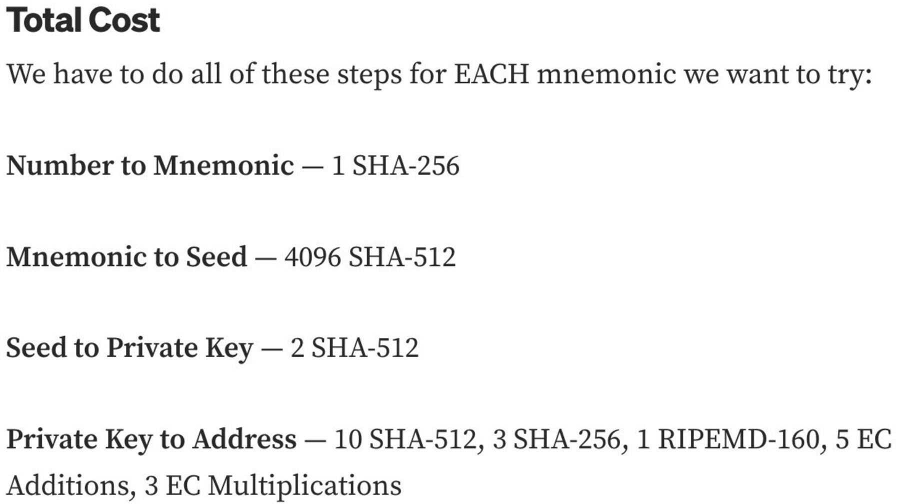
Transaction Details

| | |
|---|---|
| Hash | 5327 df4f f47d c58a<br>a8c8 b4f8 ba58 2f73<br>7984 2a33 cffd 8ed3<br>801f 0406 71ea 5ab3 |
| Confirmed | In block #635085 |
| Confirmations 19 ● | |
| Date | June 17, 2020<br>3:49:17 AM GMT+02:00 |
| Inputs | 1 BTC<br>3HX5tttedDeh<br>KWTTGpxaPAbo<br>157fnjn89s |
| Outputs | 0.99 BTC<br>3NXDwKnZdqFN<br>SeCVyj9u8GnQ<br>tJ4ovu8cSt |
| Miner fee: | 0.01 BTC |

ZenGo

# BTC Challenge: The middle part

- Guessing 4 words out of 12
- 2^44-4 = 1.1 Trillion



**Total Cost**

We have to do all of these steps for EACH mnemonic we want to try:

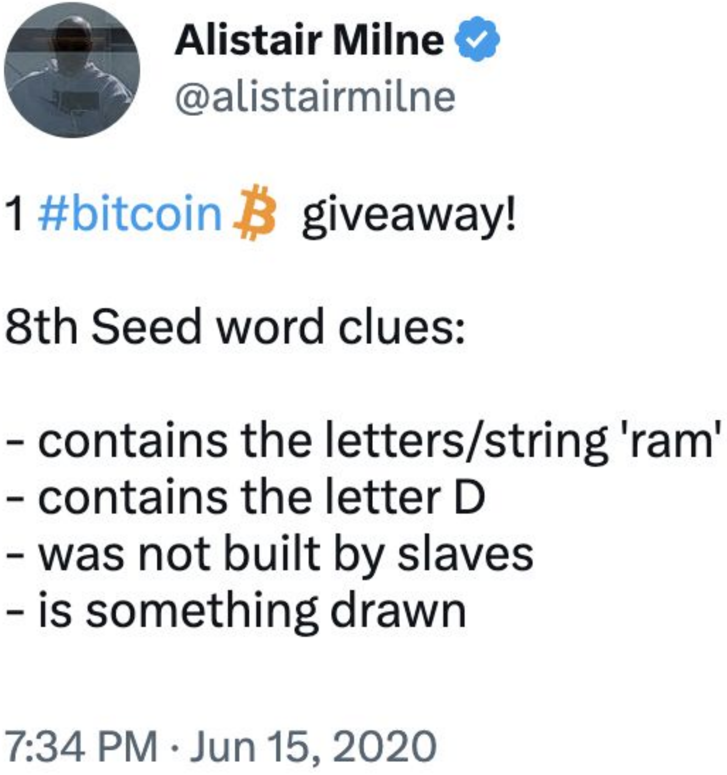
**Number to Mnemonic** — 1 SHA-256

**Mnemonic to Seed** — 4096 SHA-512

**Seed to Private Key** — 2 SHA-512

**Private Key to Address** — 10 SHA-512, 3 SHA-256, 1 RIPEMD-160, 5 EC Additions, 3 EC Multiplications

# BTC Challenge: #8 word released



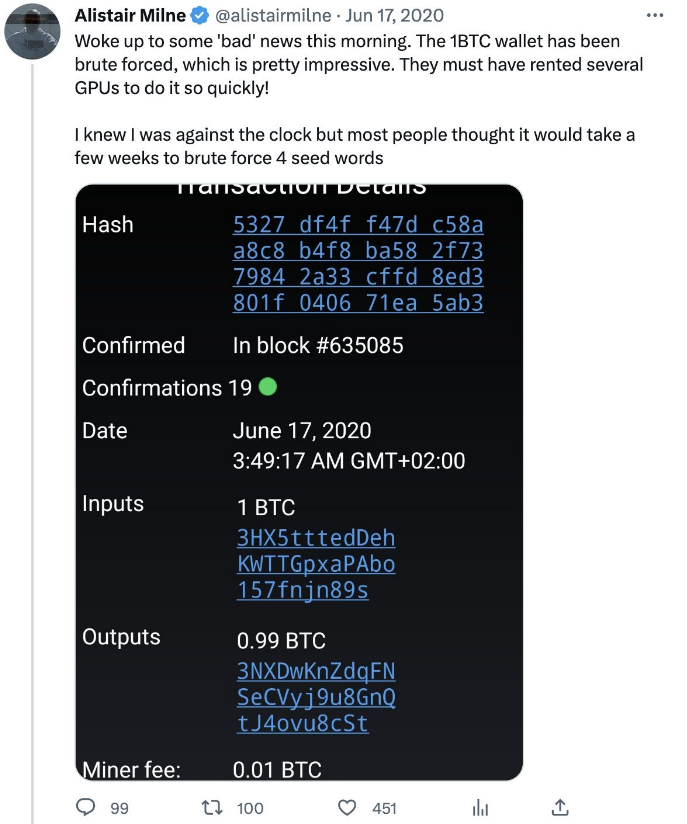
Alistair Milne
@alistairmilne

1 #bitcoin ₿ giveaway!

8th Seed word clues:

- contains the letters/string 'ram'
- contains the letter D
- was not built by slaves
- is something drawn

7:34 PM · Jun 15, 2020



Alistair Milne @alistairmilne · Jun 17, 2020

Woke up to some 'bad' news this morning. The 1BTC wallet has been brute forced, which is pretty impressive. They must have rented several GPUs to do it so quickly!

I knew I was against the clock but most people thought it would take a few weeks to brute force 4 seed words

Transaction Details

Hash 5327 df4f f47d c58a a8c8 b4f8 ba58 2f73 7984 2a33 cffd 8ed3 801f 0406 71ea 5ab3

Confirmed In block #635085

Confirmations 19 ●

Date June 17, 2020 3:49:17 AM GMT+02:00

Inputs 1 BTC 3HX5tttedDehKWTTGpxaPAbo157fnjn89s

Outputs 0.99 BTC 3NXDwKnZdqFNSeCVyj9u8GnQtJ4ovu8cSt

Miner fee: 0.01 BTC

# BTC Challenge: How much time would it take?

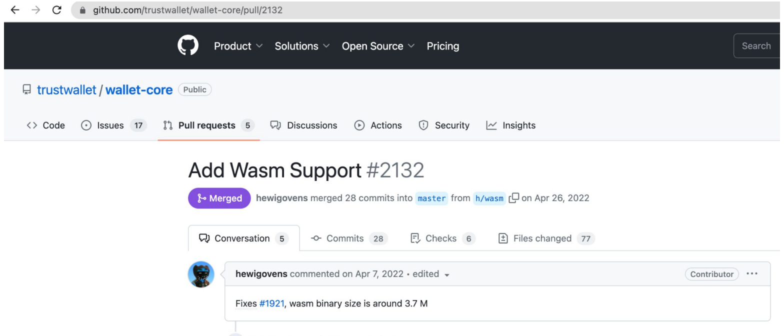
- On laptop: 25 years
- Beast machine (32 cores): 4 years
- GPUs: ~30 hours
- Read more here: https://medium.com/@johncantrell97/how-i-checked-over-1-trillion-mnemonics-in-30-hours-to-win-a-bitcoin-635fe051a752

# Trust wallet extension

ZenGo

# Trust Wallet Extension

- Trust wallet: mobile app, seed based (acquired by Binance)
- Core crypto implemented in C++
  - Open source [https://github.com/trustwallet/wallet-core](https://github.com/trustwallet/wallet-core)
- How would you build a wallet extension?
  - Compile to WASM
- What can go wrong?



Add Wasm Support #2132

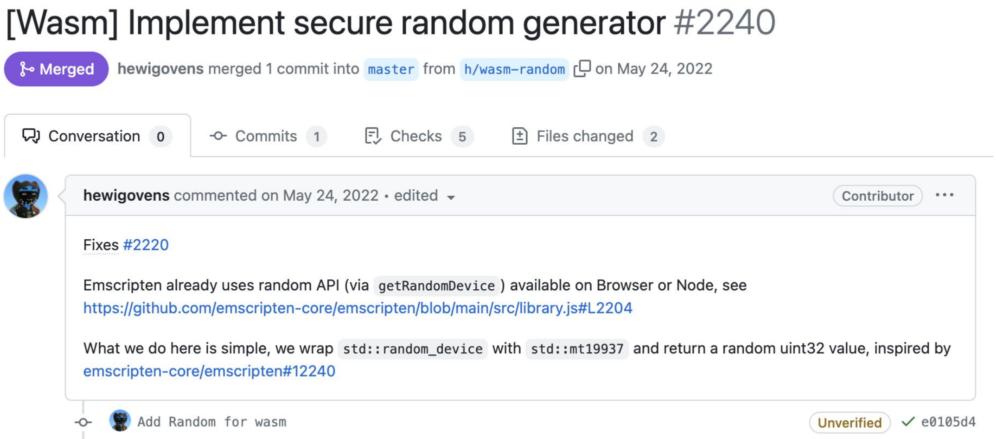
hewigovens merged 28 commits into master from h/wasm on Apr 26, 2022

hewigovens commented on Apr 7, 2022 • edited

Fixes #1921, wasm binary size is around 3.7 M

# Randomness

- In mobile environment, Trust wallet uses the OS random API
- No (immediate) access from browser to OS random API
- Trust solution: Use some C++ API
- Can you spot the issue?



[Wasm] Implement secure random generator #2240

hewigovens merged 1 commit into `master` from `h/wasm-random` on May 24, 2022

hewigovens commented on May 24, 2022 • edited

Fixes #2220

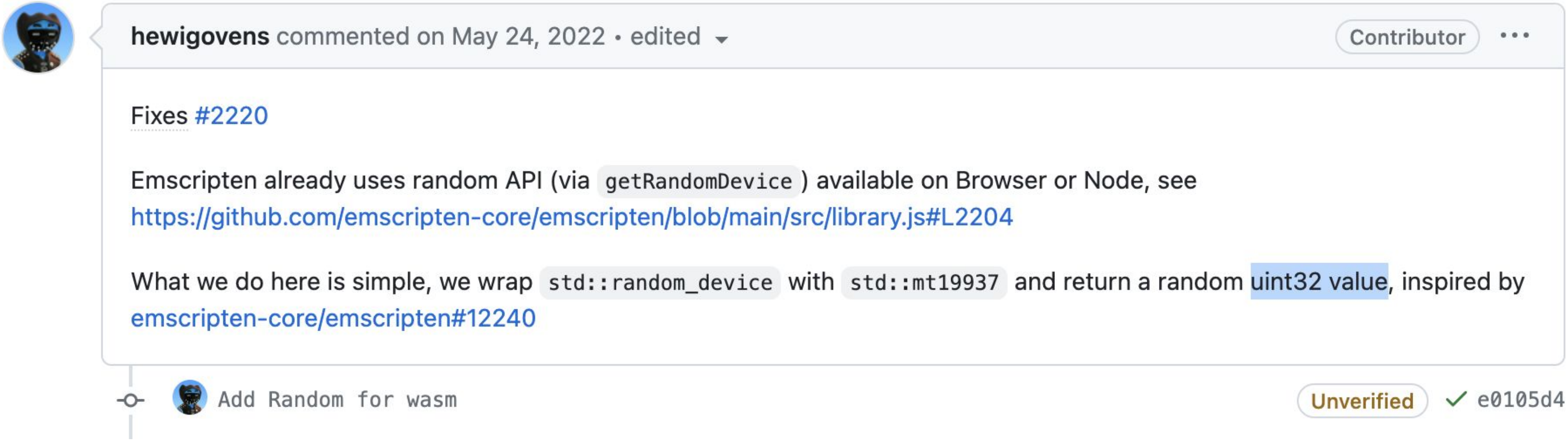
Emscripten already uses random API (via `getRandomDevice`) available on Browser or Node, see https://github.com/emscripten-core/emscripten/blob/main/src/library.js#L2204

What we do here is simple, we wrap `std::random_device` with `std::mt19937` and return a random uint32 value, inspired by emscripten-core/emscripten#12240

```
Add Random for wasm
```

ZenGo

# The issue



**hewigovens** commented on May 24, 2022 • edited ▾ Contributor •••

Fixes #2220

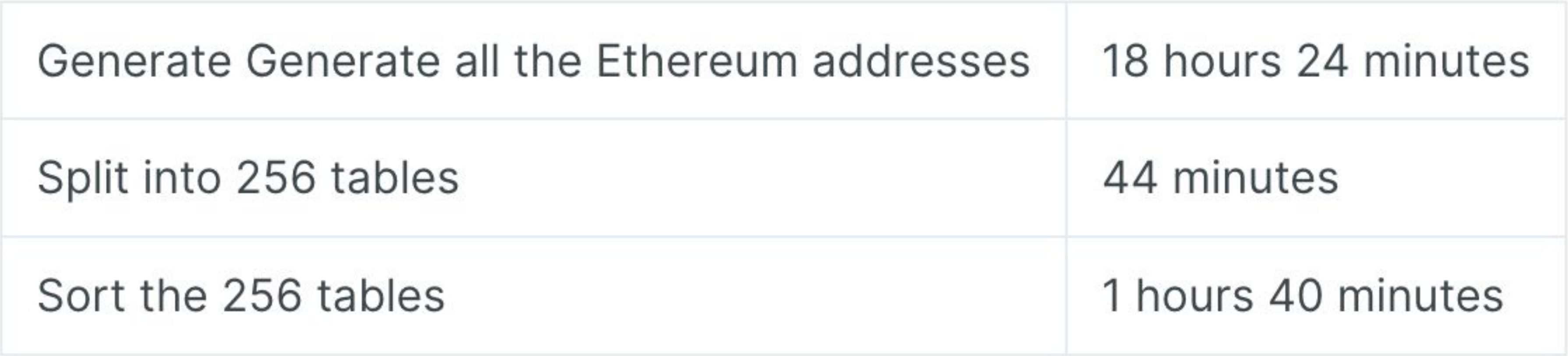
Emscripten already uses random API (via `getRandomDevice`) available on Browser or Node, see https://github.com/emscripten-core/emscripten/blob/main/src/library.js#L2204

What we do here is simple, we wrap `std::random_device` with `std::mt19937` and return a random uint32 value, inspired by emscripten-core/emscripten#12240

`Add Random for wasm` Unverified ✓ `e0105d4`

# Brute-forcing 32 bits

- 2 GPUs
- < 24 hours



| | |
|---|---|
| Generate Generate all the Ethereum addresses | 18 hours 24 minutes |
| Split into 256 tables | 44 minutes |
| Sort the 256 tables | 1 hours 40 minutes |

# Profanity vanity Addresses

ZenGo

# Vanity addresses: Motivation

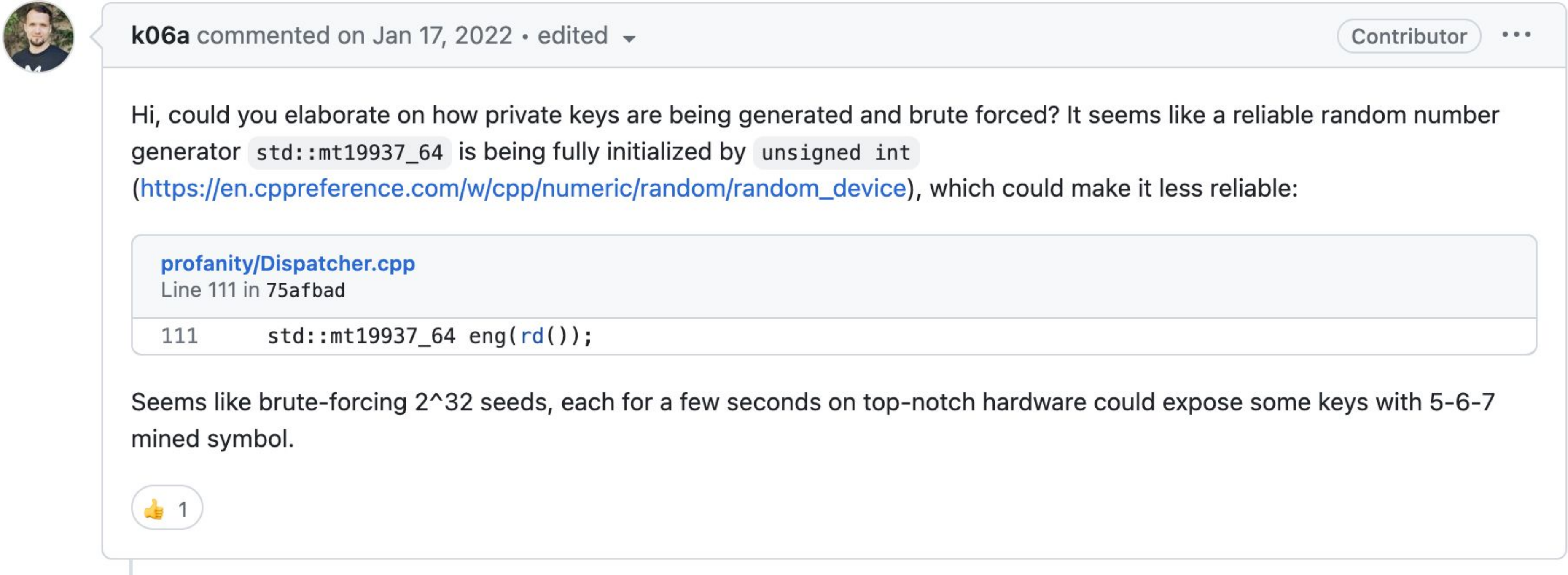
- Vanity
- People like personalization
- Can be thought as a security measure against phishing
- Some gas savings if contract address starts with enough leading 0s

# Vanity addresses: How to

- Bruteforce
- Each hex character represents 4 bits
- To create 0babe3333bb:
  - https://etherscan.io/address/0x0babe3333bb2904dc3cdc16b80b64dc3ec5ac4d3
  - 11 (9? 10?) vanity hex digits = 44 bit = 2^44
- Profanity does that for you
  - https://github.com/johguse/profanity
  - Optimized for GPUs

ZenGo

# Vanity addresses: The issue



**k06a** commented on Jan 17, 2022 • edited

Hi, could you elaborate on how private keys are being generated and brute forced? It seems like a reliable random number generator `std::mt19937_64` is being fully initialized by `unsigned int` (https://en.cppreference.com/w/cpp/numeric/random/random_device), which could make it less reliable:

profanity/Dispatcher.cpp
Line 111 in 75afbad

```
111        std::mt19937_64 eng(rd());
```

Seems like brute-forcing 2^32 seeds, each for a few seconds on top-notch hardware could expose some keys with 5-6-7 mined symbol.

ZenGo

# Abusing profanity: Naive method

- Create all 32 bit options of addresses (like with Trust Wallet)
- Then try to bruteforce by incrementing each private key
- So for 10 hex chars vanity:
    - 2^(32 + 40) = too much

> *At first sight, it appeared that 8+ character vanity addresses were quite safe (please, read through the end of this post)*
>
> - **1 inch blog**

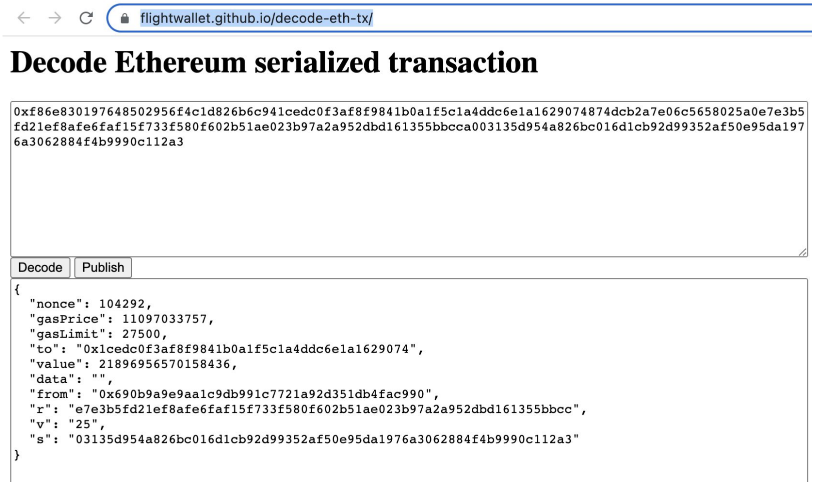
ZenGo

# Abusing profanity: Not naive method

- Create all 32 bit options of **public keys** (like with Trust Wallet)
  - Less expensive than addresses
  - Doable in < 24 hours
- **Start from a public key of a vanity address**
  - They are highly self evident
  - Bruteforce “backwards”:
    - decrement public key until you reach known public key
    - $P = S * G$ → $P' = (S - 1) * G = S*G - G = P - G$
- Instead of 2^(32+40) , 2^32+2^40 ~= 2^40 → bruteforce-able

# The public key of an EOA Ethereum address

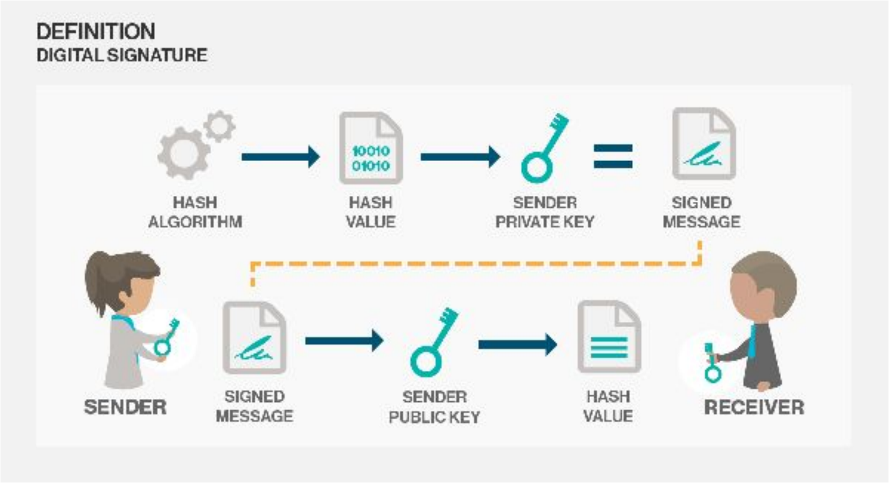
- EOA address: The last 20 bytes of the hash of the public key
  - Does not reveal its public key
- However, it can be extracted from the Tx signature (v,r,s) on chain



## Decode Ethereum serialized transaction

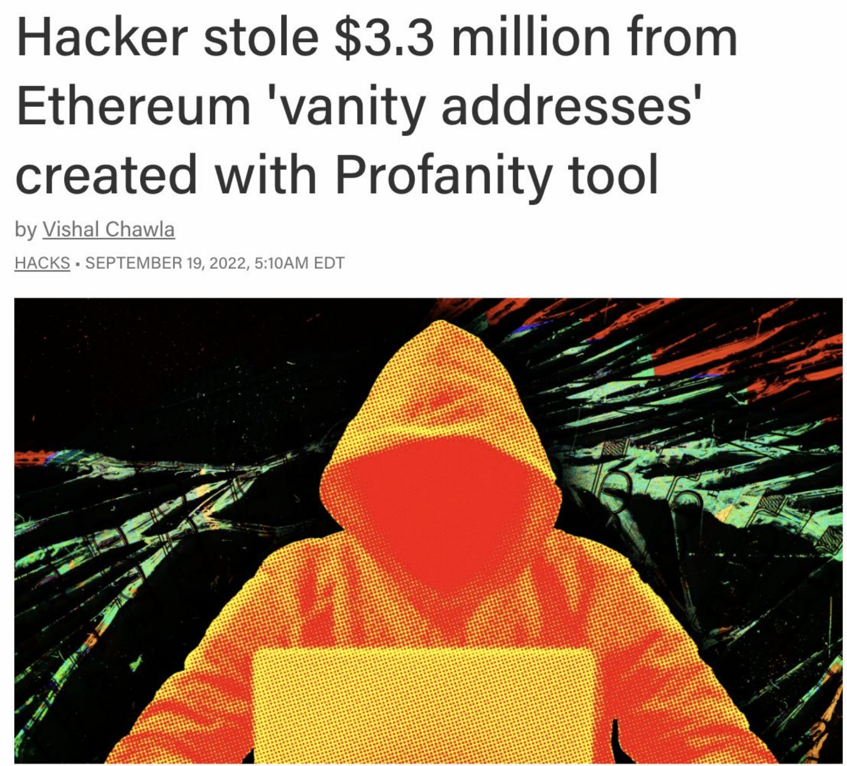
```
0xf86e830197648502956f4c1d826b6c941cedc0f3af8f9841b0a1f5c1a4ddc6e1a1629074874dcb2a7e06c5658025a0e7e3b5
fd21ef8afe6faf15f733f580f602b51ae023b97a2a952dbd161355bbcca003135d954a826bc016d1cb92d99352af50e95da197
6a3062884f4b9990c112a3
```

```
{
  "nonce": 104292,
  "gasPrice": 11097033757,
  "gasLimit": 27500,
  "to": "0x1cedc0f3af8f9841b0a1f5c1a4ddc6e1a1629074",
  "value": 21896956570158436,
  "data": "",
  "from": "0x690b9a9e9aa1c9db991c7721a92d351db4fac990",
  "r": "e7e3b5fd21ef8afe6faf15f733f580f602b51ae023b97a2a952dbd161355bbcc",
  "v": "25",
  "s": "03135d954a826bc016d1cb92d99352af50e95da1976a3062884f4b9990c112a3"
}
```



# Vanity addresses: The losses (phase #1)

○



Hacker stole $3.3 million from Ethereum 'vanity addresses' created with Profanity tool

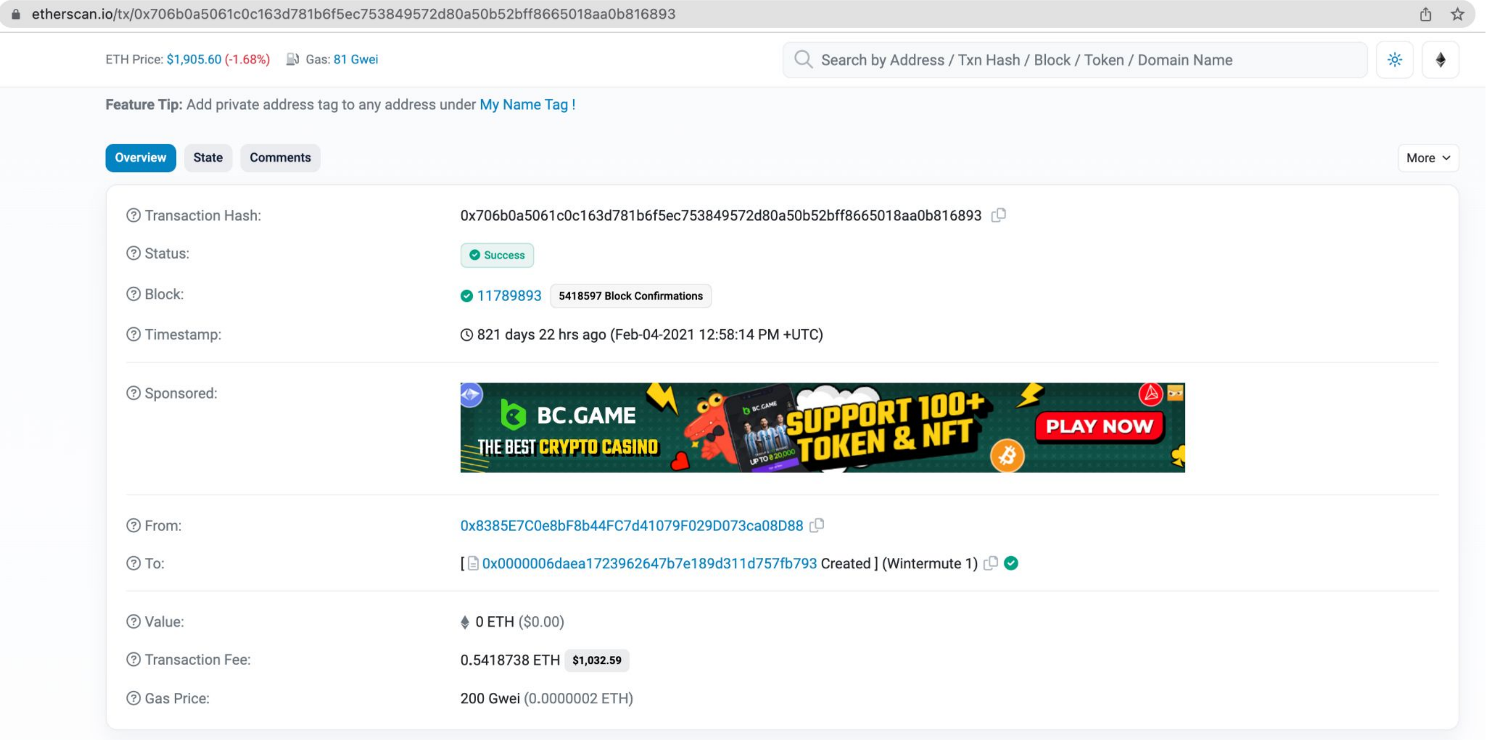
by Vishal Chawla

HACKS • SEPTEMBER 19, 2022, 5:10AM EDT

# Contracts address

- We only talked about EOA addresses and "forgot" about contract addresses
- Contract address: The last 20 bytes of the hash of the ~~public key~~
  - Deployer address
  - Deployer nonce
  - `sha3(rlp.encode([normalize_address(sender), nonce]))[12:]`
- Profanity does that calculation for the its users to create such deployer key that would yield the right contract address
- Apparently, others forgot about it as well...

# Contracts address: Wintermute



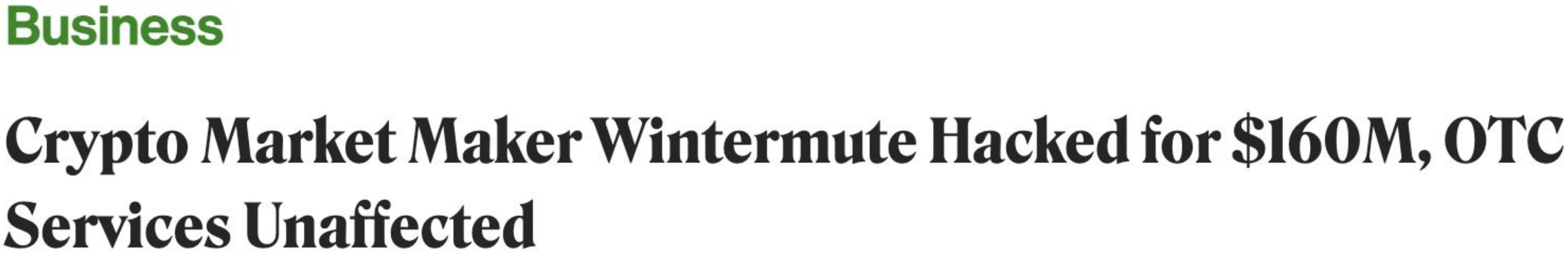
etherscan.io/tx/0x706b0a5061c0c163d781b6f5ec753849572d80a50b52bff8665018aa0b816893

ETH Price: $1,905.60 (-1.68%) Gas: 81 Gwei

**Feature Tip:** Add private address tag to any address under My Name Tag !

| | |
|---|---|
| Transaction Hash: | 0x706b0a5061c0c163d781b6f5ec753849572d80a50b52bff8665018aa0b816893 |
| Status: | Success |
| Block: | 11789893 5418597 Block Confirmations |
| Timestamp: | 821 days 22 hrs ago (Feb-04-2021 12:58:14 PM +UTC) |
| Sponsored: | |
| From: | 0x8385E7C0e8bF8b44FC7d41079F029D073ca08D88 |
| To: | [ 0x0000006daea1723962647b7e189d311d757fb793 Created ] (Wintermute 1) ✓ |
| Value: | 0 ETH ($0.00) |
| Transaction Fee: | 0.5418738 ETH $1,032.59 |
| Gas Price: | 200 Gwei (0.0000002 ETH) |

# Vanity addresses: The losses (phase #2)

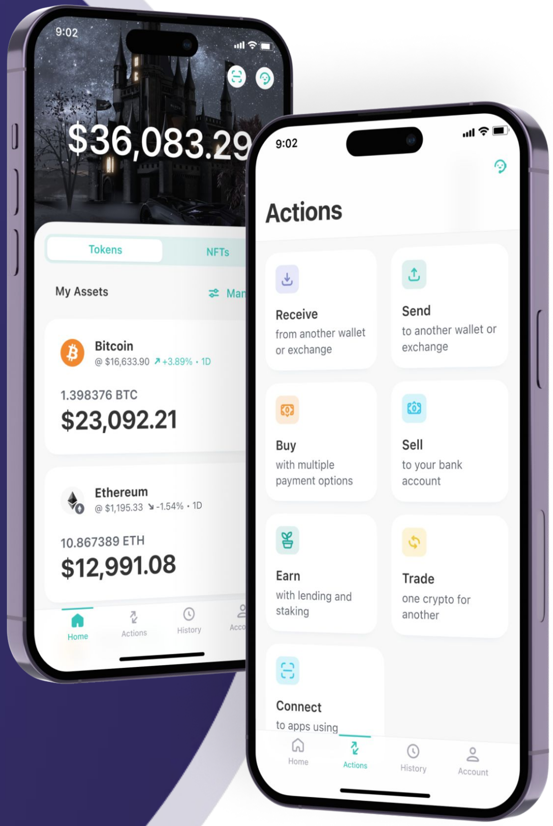


Business

Crypto Market Maker Wintermute Hacked for $160M, OTC Services Unaffected twitter.com/zengo

github.com/ZenGo-X

zengo.com/contact



ZenGo

Scan QR code to Download the App

www.zengo.com