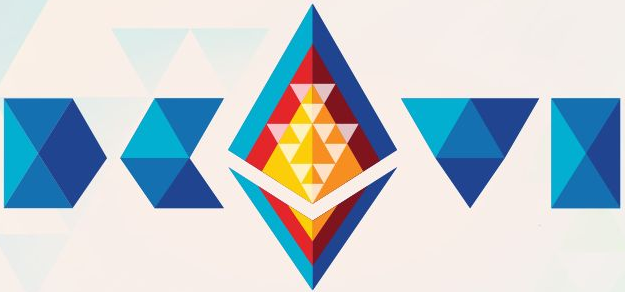
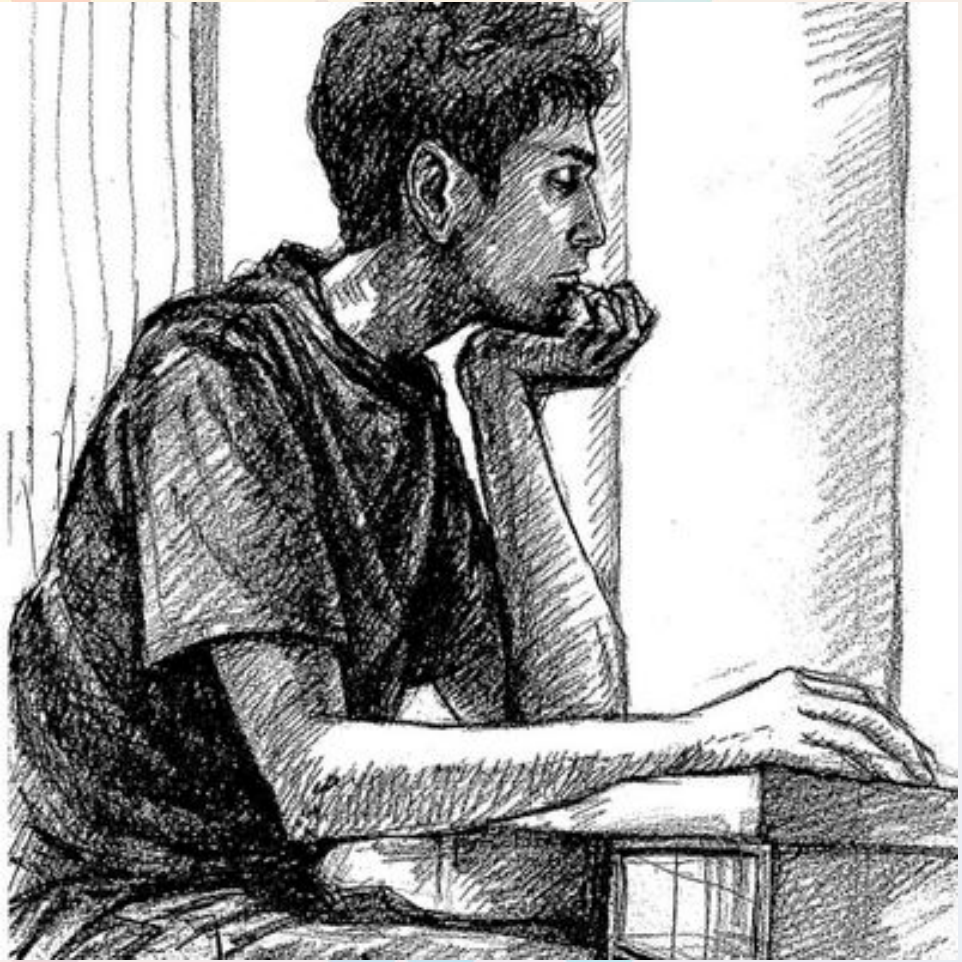


Building EVM transaction decoders

Opensource, extendable and modular

Lefteris Karapetsas

Creator of bugs @ rotki



Who am I?

- Worked in Ethereum since 2014
- C++ client, Solidity compiler
- Built the DAO, whitehat-hacked it and fought in the DAO wars and later the ETC cleanup.
- Worked on L2 payments channels with Raiden Network
- Founder of rotki

Section 1

Understanding user transactions

The problem we face

Getting transactions of a user

- No built-in way to get transactions of an address
- Need to utilize third party services
- Not decentralized

Once you have the transactions you don’t know what they mean

- Lack of a universal transaction decoders
- Need to use third party services
- Most services are centralized and mostly protocol specific

Value:	0.000000000015711408 Ether (< $0.000001)
Transaction Fee:	0.010729956096490184 Ether ($14.23)
Gas Price:	0.000000019810710191 Ether (19.810710191 Gwei)
Gas Limit & Usage by Txn:	617,207 | 541,624 (87.75%)
Gas Fees:	Base: 19.342201445 Gwei | Max: 19.810710191 Gwei | Max Priority: 19.810710191 Gwei
Burnt & Txn Savings Fees:	Burnt: 0.01047620051544668 Ether ($13.89) Txn Savings: 0 Ether ($0.00)
Other Attributes:	Txn Type: 2 (EIP-1559) Nonce: 11536 Position In Block: 279
Input Data:	0x0906f7028f30ba1f75ec27a014591789d117a1bb4703bff80000000000 00001783c255c185fbc7000100022f3f05af68a2ab029f319e265b8189b5 b5c2152cacc791b95b3da180010000000002dab907160036aad55e000002 ee802b312a957e00dd7bbc08eec8bb93d40e981d01000000000000023a68 362982f8720100018db1b906d47dfc1d84a87fc49bd0522e285b98b90100

View Input As

Understanding user transactions

Users don't understand what a transaction will do or has done easily.

- Big hex blob
- No metadata
- No human readable info

Current ways of gaining insight

The Ethereum community has some tools at the moment at its disposal to gain insight on user transaction

- Etherscan
- The Graph
- Centralized APIs
  - Covalent
  - Moralis
  - etc.

Swap 0.157413805631513259 Ether For 1.694411558396230599 SHAKE On Uniswap V2
Swap 1.694411558396230599 SHAKE For 13,479.45582321999929891 MILK2 On Uniswap V2
Swap 13,479.45582321999929891 MILK2 For 0.160555318559045746 Ether On Uniswap V2
Swap 0.147136821360800261 Ether For 6,643.477826522244028633 ANGLE On Sushiswap
Swap 6,643.477826522244028633 ANGLE For 205.156457391863548534 agEUR On Sushiswap
Swap 205.156457391863548534 agEUR For 0.150862573367402416 Ether On Uniswap V3
Swap 0.534634265123429893 Ether For 526.893572699 TONCOIN On Uniswap V3
Swap 526.893572699 TONCOIN For 5,381.073373840155623686 HOP On Uniswap V3
Swap 5,381.073373840155623686 HOP For 0.538943686975519 65 Ether On Uniswap V3

0xb28ebe0ed76e01825483a723d21a6ab29743cddc

Contract 0x30a123cbf79fdb6ac10556b20531545de0da652b

From 0x30a123cbf79f... To Uniswap V2: SH... For 0.157413805631513259 ($208.71) Wrapped Ethe... (WETH)
From Uniswap V2: SH... To Uniswap V2: SH... For 1.694411558396230599 ($207.69) SHAKE token ... (SHAKE)
From Uniswap V2: SH... To Uniswap V2: MIL... For 13,479.45582321999929891 ($207.34) MilkyWay Tok... (MILK2)
From Uniswap V2: MIL... To 0x30a123cbf79f... For 0.160555318559045746 ($212.87) Wrapped Ethe... (WETH)
From Uniswap V3: agE... To 0x30a123cbf79f... For 0.150862573367402416 ($200.02) Wrapped Ethe... (WETH)
From 0x30a123cbf79f... To SushiSwap: ANG... For 0.147136821360800261 ($195.08) Wrapped Ethe... (WETH)
From SushiSwap: ANG... To SushiSwap: agE... For 6,643.477826522244028633 ($207.62) ANGLE (ANGLE)
From SushiSwap: agE... To Uniswap V3: agE... For 205.156457391863548534 ($200.56) agEUR (agEUR)
From Uniswap V3: HOP To 0x30a123cbf79f... For 0.538943686975519 65 ($714.56) Wrapped Ethe... (WETH)
From Uniswap V3: TO... To Uniswap V3: HOP For 5,381.073373840155623686 ($698.38) Hop (HOP)
From Uniswap V3: TO... To Uniswap V3: TO... For 526.893572699 ($706.04) Wrapped TON ... (TONCOI...)
From 0x30a123cbf79f... To Uniswap V3: TO... For 0.534634265123429893 ($708.85) Wrapped Ethe... (WETH)

Etherscan

Pros:

- Ease of use
- Useful insight
- Free

Cons:

- Centralized
- Proprietary/Closed source
- Knows everything about you
- Does not decode everything

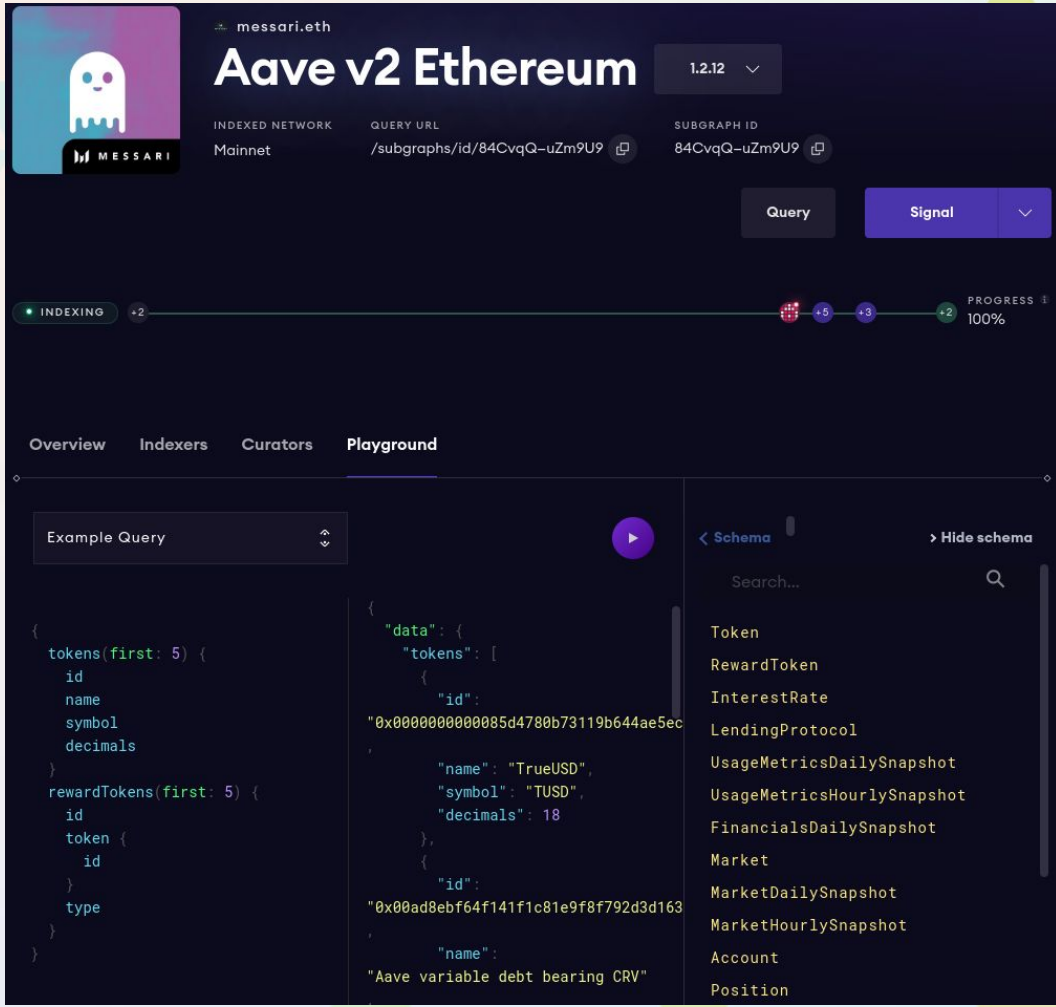
The Graph

Pros:

- Good for single protocol data

Cons:

- Needs payment per query
- Built for single protocol data (subgraphs)
- Does not scale. Needs separate subgraph per protocol supported.
- Does not work with local apps



Centralized APIs

Centralized APIs like Covalent, Moralis, Alchemy etc.

Pros:

- Ease of use

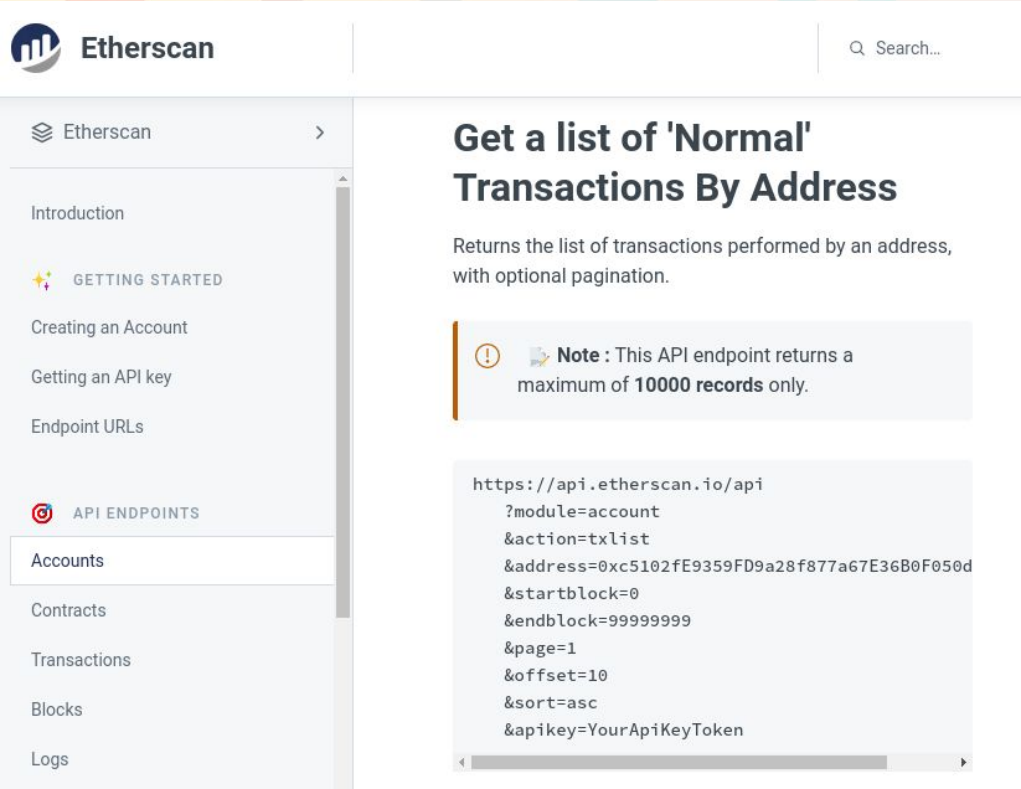
Cons:

- Centralized
- Knows everything about you
- Needs to support each protocol you need

Section 2

Getting accurate historical data

The **original sin** of Ethereum.
No built-in way to get
transactions for an address.



Etherscan APIs

Easiest way to get transactions for an address.

Drawbacks:

- Does not detect all address appearances
- Rate limited (can pay for bigger limits)
- Centralized
  - Can go down
  - Access to API can be cut
  - Can monitor you and map ip to address and data

Trueblocks

The best and most complete way of getting transaction data.

Pros:

- Detects all appearances of an address
- Is decentralized, gets all data from local nodes
- Is super fast
- Shares the created index with others

Drawbacks:

- Hard to setup
- Requires a local node
- Requires trueblocks to create the index

TrueBlocks Docs Data Model Blog Papers

chifra export

The `chifra export` tools provides a major part of the functionality of the TrueBlocks system. Using the index of appearances created with `chifra scrape` and the list of transaction identifiers created with `chifra list`, `chifra export` completes the actual extraction of an address's transactional history from the node.

You may use `topics`, `fourbyte` values at the start of a transaction's input data, and/or a log's `source address` or `emitter` to filter the results.

You may also choose which portions of the Ethereum data structures (`--transactions`, `--logs`, `--traces`, etc.) as you wish.

By default, the results of the extraction are delivered to your console, however, you may export the results to any database (with a little bit of work). The format of the data, its content and its destination are up to you.

```
Purpose:
  Export full detail of transactions for one or more addresses.

Usage:
  chifra export [flags] <address> [address...] [topics...] [fourbytes...]

Arguments:
  addrs - one or more addresses (0x...) to export (required)
  topics - filter by one or more log topics (only for --logs option)
  fourbytes - filter by one or more fourbytes (only for transactions and trac
```

The stack of true decentralization

Working in crypto should be striving for decentralization!

- Hardware: Self-hosted hardware like dappnode that is under your control
- Client: A node for each chain you need fast data for. Mainet, gnosis chain etc.
- Indexer: A trueblocks indexer for each client so you can answer the question of how to go from an address to a list of transaction hashes
- Aggregating & decoding: rotki platform as a way to aggregate all data and go from transactions to a common data format consumable by a human

Section 3

Decoder input

Decoder input

Once data is retrieved it needs to be processed to extract needed information. This is done through two methods

- Transaction traces
  - Geth style traces
  - Parity style traces

- Transaction receipts

Client	Method invocation
Go	`debug.TraceTransaction(txHash common.Hash, logger *vm.LogConfig) (*ExecutionResult, error)`
Console	`debug.traceTransaction(txHash, [options])`
RPC	`{"method": "debug_traceTransaction", "params": [txHash, {}]}`

Example

```
> debug.traceTransaction("0x2059dd53ecac9827faad14d364f9e04b1d5fe5b506e3acc886eff7a6f88a696a")
{
  gas: 85301,
  returnValue: "",
  structLogs: [{
      depth: 1,
      error: "",
      gas: 162106,
      gasCost: 3,
      memory: null,
      op: "PUSH1",
      pc: 0,
      stack: [],
      storage: {}
  },
    /* snip */
  {
      depth: 1,
      error: "",
      gas: 100000,
      gasCost: 0,
      memory: ["0000000000000000000000000000000000000000000000000000000000000006", "000000000000000
      op: "STOP",
      pc: 120,
      stack: ["000000000000000000000000000000000000000000000000000000000d67cbec9"],
      storage: {
        0000000000000000000000000000000000000000000000000000000000000004: "8241fa522772837f0d05511f
        0000000000000000000000000000000000000000000000000000000000000006: "000000000000000000000000
        f652222313e28459528d920b65115c16c04f3efc82aaedc97be59f3f377c0d3f: "000000000000000000000000
      }
  }]
```

Geth style trace

- State of virtual machine at each execution step including all details (Opcode, PC, storage diff etc.)
- Very detailed but hard to read/use
- Can grow extremely large for complex transactions

Parity style - trace

The `trace` command gives you a call tree of the transaction showing you the call stack generated.

This does not require an archive node.

```
In [10]: chain.provider.get_call_tree(tx)
Out[10]:
CALL: 0x2e59A20f205bB85a89C53f1936454680651E618e.<0xf98a4eca> [159140 gas]
├─ CALL: 0xb8FFC3Cd6e7Cf5a098A1c92F48009765B24088Dc.<0xbe00bbd8> [8263 gas]
│   └─ DELEGATECALL: 0x2b33CF282f867A7FF693A66e11B0FcC5552e4425.<0xbe00bbd8> [2820 gas]
└─ DELEGATECALL: 0x72fb5253AD16307B9E773d2A78CaC58E309d5Ba4.<0xf98a4eca> [140266 gas]
    ├─ CALL: 0xb8FFC3Cd6e7Cf5a098A1c92F48009765B24088Dc.<0xbe00bbd8> [3763 gas]
    │   └─ DELEGATECALL: 0x2b33CF282f867A7FF693A66e11B0FcC5552e4425.<0xbe00bbd8> [2820 gas]
    ├─ CALL: 0x853cc0D5917f49B57B8e9F89e491F5E18919093A.<0x04bf2a7f> [8519 gas]
    │   └─ DELEGATECALL: 0xBF1Ce0Bc4EdaAD8e576b3b55e19c4C15Cf6999eb.<0x04bf2a7f> [3006 gas]
    └─ DELEGATECALL: 0x5cEb19e1890f677c3676d5ecDF7c501eBA01A054.<0x279cea35> [78866 gas]
        └─ CALL: 0xae7ab96520DE3A18E5e111B5EaAb095312D7fE84.<0x8cef3612> [72295 gas]
            ├─ CALL: 0xb8FFC3Cd6e7Cf5a098A1c92F48009765B24088Dc.<0xbe00bbd8> [3763 gas]
            │   └─ DELEGATECALL: 0x2b33CF282f867A7FF693A66e11B0FcC5552e4425.<0xbe00bbd8> [2820 gas]
            └─ DELEGATECALL: 0x47EbaB13B806773ec2A2d16873e2dF770D130b50.<0x8cef3612> [60409 gas]
                └─ CALL: 0xb8FFC3Cd6e7Cf5a098A1c92F48009765B24088Dc.<0xfdef9106> [23636 gas]
                    └─ DELEGATECALL: 0x2b33CF282f867A7FF693A66e11B0FcC5552e4425.<0xfdef9106> [22675 gas]
                        └─ CALL: 0x9895F0F17cc1d1891b6f18ee0b483B6f221b37Bb.<0xfdef9106> [16172 gas]
                            ├─ CALL: 0xb8FFC3Cd6e7Cf5a098A1c92F48009765B24088Dc.<0xbe00bbd8> [3763 gas]
                            │   └─ DELEGATECALL: 0x2b33CF282f867A7FF693A66e11B0FcC5552e4425.<0xbe00bbd8> [2820 gas]
                            └─ DELEGATECALL: 0x9f3b9198911054B122fDb865f8A5Ac516201c339.<0xfdef9106> [4268 gas]

In [11]: chain.provider.get_transaction(tx).show_trace()
Call trace for '0xe61167aa87b2a7aa9bd68834bf703877d22315d6d765345ebf0135eb8c33c406'
txn.origin=0xDbBC6A93ae517D3ea568C04219cbBBd025f01CB6
Voting.executeVote(_voteId=134) [159140 gas]
├─ Kernel.getApp(_namespace=0xf1..ac0f, _appId=0x0a..0e1e) -> Voting [8263 gas]
│   └─ (delegate) Kernel.getApp(_namespace=0xf1..ac0f, _appId=0x0a..0e1e) -> Voting [2820 gas]
└─ (delegate) Voting.executeVote(_voteId=134) [140266 gas]
    ├─ Kernel.getApp(_namespace=0xd6..02fb, _appId=0xdd..bd61) -> EVMScriptRegistry [3763 gas]
    │   └─ (delegate) Kernel.getApp(_namespace=0xd6..02fb, _appId=0xdd..bd61) -> EVMScriptRegistry [2820 gas]
    ├─ EVMScriptRegistry.getScriptExecutor(_script=0x00..1388) -> CallsScript [8519 gas]
    │   └─ (delegate) EVMScriptRegistry.getScriptExecutor(_script=0x00..1388) -> CallsScript [3006 gas]
    └─ (delegate) CallsScript.execScript(_script=0x00..1388, '', _blacklist=[]) -> '' [78866 gas]
        └─ stETH.setFeeDistribution(
                _treasuryFeeBasisPoints=5000,
                _insuranceFeeBasisPoints=0,
                _operatorsFeeBasisPoints=5000
            ) [72295 gas]
            ├─ Kernel.getApp(_namespace=0xf1..ac0f, _appId=0x3c..f320) -> Lido [3763 gas]
            │   └─ (delegate) Kernel.getApp(_namespace=0xf1..ac0f, _appId=0x3c..f320) -> Lido [2820 gas]
            └─ (delegate) stETH.setFeeDistribution(
                    _treasuryFeeBasisPoints=5000,
                    _insuranceFeeBasisPoints=0,
                    _operatorsFeeBasisPoints=5000
                ) [60409 gas]
                └─ Kernel.hasPermission(_who=Voting, _where=Lido, _what=0x46..3bd8, _how='') -> True [23636 gas]
                    └─ (delegate) Kernel.hasPermission(_who=Voting, _where=Lido, _what=0x46..3bd8, _how='') -> True [22675 gas]
                        └─ ACL.hasPermission(_who=Voting, _where=Lido, _what=0x46..3bd8, _how='') -> True [16172 gas]
                            ├─ Kernel.getApp(_namespace=0xf1..ac0f, _appId=0xe3..ad6a) -> ACL [3763 gas]
                            │   └─ (delegate) Kernel.getApp(_namespace=0xf1..ac0f, _appId=0xe3..ad6a) -> ACL [2820 gas]
                            └─ (delegate) ACL.hasPermission(_who=Voting, _where=Lido, _what=0x46..3bd8, _how='') -> True [4268 gas]
```

Parity style - state difff

```
In [20]: chain.provider._make_request("trace_replayTransaction", [tx, ["stateDiff"]])['result']['stateDiff']
Out[20]:
{'0x1a9c8182c09f50c8318d769245bea52c32be35bc': {'balance': '=',
  'code': '=',
  'nonce': '=',
  'storage': {'0xf37250c9cd5cb3956215851fcc64976a7eb364cef6fd3947291f1a486b18d7e6': {'*': {'from': '0x0000000000000000000000000000000000000000000000000000000000000001',
     'to': '0x0000000000000000000000000000000000000000000000000000000000000000'}}}},
 '0x1f9840a85d5af5bf1d1762f925bdaddc4201f984': {'balance': '=',
  'code': '=',
  'nonce': '=',
  'storage': {'0xbadf4d9ae6d3bd2d2810ec136f3f25ad1b0a470538d81bb7903ed8a7ac897a1f': {'*': {'from': '0x000000000000000000000000000000000000000000007008d6db998545958fa2f7',
     'to': '0x0000000000000000000000000000000000000006f3514bfcab857f48fa2f7'}},
   '0xf6b9ad2a69df067cdbca51d731c9eaf879ad6c6ee94c6e31470708a49a40b068': {'*': {'from': '0x0000000000000000000000000000000000000000000000000000000000000000',
     'to': '0x0000000000000000000000000000000000000000000d3c21bcecceda1000000'}}}},
 '0x2b3842124edc04ae8bb41739d06ba20e33277bf33': {'balance': {'*': {'from': '0x61ed6f2bd5da8e',
    'to': '0x5d80bd8f2f868e'}},
  'code': '=',
  'nonce': {'*': {'from': '0x73', 'to': '0x74'}},
  'storage': {}},
 '0xc4e172459f1e7939d522503b81afaac1014ce6f6': {'balance': '=',
  'code': '=',
  'nonce': '=',
  'storage': {'0xcc69885fda6bcc1a4ace058b4a62bf5e179ea78fd58a1ccd71c22cc9b688793a': {'*': {'from': '0x0000000000000000000000000000000000000000000000000000000000000000',
     'to': '0x0000000000000000000000000000000000000000000000000000000000000100'}}}},
 '0xea674fdde714fd979de3edf0f56aa9716b898ec8': {'balance': {'*': {'from': '0x740c6ac570956c3a1d',
    'to': '0x740c6f322232128e1d'}},
  'code': '=',
  'nonce': '=',
  'storage': {}}}
```

The stateDiff command of parity tracing gives a per acount difference of:

- Balance
- Code
- Nonce
- Storage

Transaction Receipts

Contracts generate log events. These are contained in the receipt of a transaction. A log event is:

- Generated by almost anything. Token transfer, NFT mint, vault creation etc.
- Contains indexed data in the topics
- Contains also arbitrary data

```
receipt = EthereumTxReceipt(
    tx_hash=evmhash,
    contract_address=None,
    status=True,
    type=0,
    logs=[
        EthereumTxReceiptLog(
            log_index=342,
            data='0x0000000000000000000000000000000000000000000000000000000de0b6b3a7640000',
            address=A_CVX.resolve_to_evm_token().evm_address,
            removed=False,
            topics=[
                '0xddf252ad1be2c89b69c2b068fc378daa952ba7f163c4a11628f55a4df523b3ef',
                '0x000000000000000000000000000000005b186c93a50d3cb435fe2933427d36e6dc688e4b',
                '0x000000000000000000000000000000cf50b810e57ac33b91dcf525c6ddd9881b139332',
            ],
        ), EthereumTxReceiptLog(
            log_index=343,
            data='0xffffffffffffffffffffffffffffffffffffffffffffffffffffffff21f494c589bffff',
            address=A_CVX.resolve_to_evm_token().evm_address,
            removed=False,
            topics=[
                '0x8c5be1e5ebec7d5bd14f71427d1e84f3dd0314c0f7b2291e5b200ac8c7c3b925',
                '0x000000000000000000000000000000005b186c93a50d3cb435fe2933427d36e6dc688e4b',
                '0x000000000000000000000000000000cf50b810e57ac33b91dcf525c6ddd9881b139332',
            ],
        )
    ],
)
```

Querying is expensive.
Persistence is key.

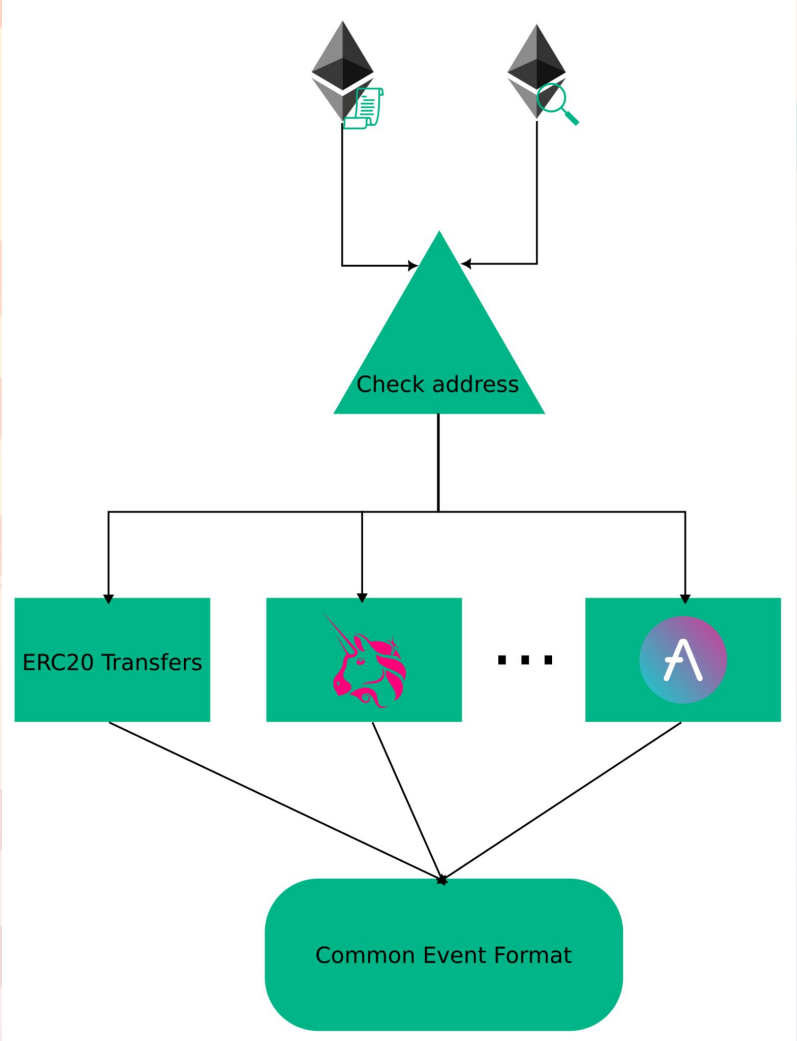
Data persistence

- Performing traces takes time
- Querying for transactions takes time
- Querying receipts takes time
- Getting logs take time

The solution is to have a persistence layer, say a local sqlite DB so that data can be retrieved easily

Section 4

Decoders



Decoders overview

- Data comes in from traces and receipts
- Depending on the contract address a different decoder is chosen
- Each decoder processes given input and translates to a common event format

Modularity

- Decoders are modular
- One decoder per protocol
- Easy to write, easy to use
- Drag and drop

develop | rotki / rotkehlchen / chain / ethereum / **modules** / | Go to file | Add file | ···

yabirgb and **LefterisJP** Catch price errors during liquity trove deserialization 11d04f8 4 days ago **History**

..		
aave	Catch wrong asset type resolution	4 days ago
adex	Catch wrong asset type resolution	4 days ago
airdrops	Address refactor of assets improvements	13 days ago
balancer	Catch wrong asset type resolution	4 days ago
compound	Catch wrong asset type resolution	4 days ago
convex	Remove unused code	8 days ago
curve	Major changes are done. Now let's make it work.	13 days ago
dxdaomesa	Rename EthereumTransaction to EvmTransaction	25 days ago
ens	Rename EthereumTransaction to EvmTransaction	25 days ago
eth2	Remove return from on_account_addition callback	2 months ago
gitcoin	Major changes are done. Now let's make it work.	13 days ago
hop	Rename EthereumTransaction to EvmTransaction	25 days ago
kyber	Major changes are done. Now let's make it work.	13 days ago
l2	Major changes are done. Now let's make it work.	13 days ago
liquity	Catch price errors during liquity trove deserialization	4 days ago
makerdao	Address refactor of assets improvements	13 days ago
oneinch	Major changes are done. Now let's make it work.	13 days ago
pickle_finance	Address refactor of assets improvements	13 days ago
sushiswap	Adjust variables names	10 days ago
uniswap	Catch wrong asset type resolution	4 days ago
votium	Major changes are done. Now let's make it work.	13 days ago
yearn	Use the same field for asset identifier in UnknownAsset and Unsupport...	12 days ago

A typical decoder

```python
class HopDecoder(DecoderInterface):
    def _decode_send_eth(
            self,
            tx_log: EthereumTxReceiptLog,
            transaction: EvmTransaction,
            decoded_events: List[HistoryBaseEntry],
            all_logs: List[EthereumTxReceiptLog],
            action_items: List[ActionItem],
    ) -> Tuple[Optional[HistoryBaseEntry], Optional[ActionItem]]:
        if tx_log.topics[0] != TRANSFER_TO_L2:
            return None, None

        chain_id = hex_or_bytes_to_int(tx_log.topics[1])
        recipient = hex_or_bytes_to_address(tx_log.topics[2])
        amount_raw = hex_or_bytes_to_int(tx_log.data[:32])

        name = chainid_to_name.get(chain_id, f'Unknown Chain with id {chain_id}')
        amount = from_wei(FVal(amount_raw))

        for event in decoded_events:
            if event.event_type == HistoryEventType.SPEND and event.counterparty == ETH_BRIDGE and event.asset ==
A_ETH and event.balance.amount == amount:
                if recipient == event.location_label:
                    target_str = 'at the same address'
                else:
                    target_str = f'at address {recipient}'
                event.event_type = HistoryEventType.TRANSFER
                event.event_subtype = HistoryEventSubType.BRIDGE
                event.counterparty = CPT_HOP
                event.notes = f'Bridge {amount} ETH to {name} {target_str} via Hop protocol'
                break

        return None, None

    # -- DecoderInterface methods

    def addresses_to_decoders(self) -> Dict[ChecksumEvmAddress, Tuple[Any, ...]]:
        return {
            ETH_BRIDGE: (self._decode_send_eth,),
        }

    def counterparties(self) -> List[str]:
        return [CPT_HOP]
```

- Part of HOP protocol decoder
- Takes in already decoded ERC20 transfers to the contract
- Matches them to the bridging deposit
- Creates the bridging deposit in the common event format

Common event format

- Current implementation is PoC
- Aim to represent least common denominator
- Every single action can be broken into this format

```python
@dataclass(init=True, repr=True, eq=True, order=False, unsafe_hash=False, frozen=False)
class HistoryBaseEntry(AccountingEventMixin):
    """
    Intended to be the base unit of any type of accounting. All trades, deposits,
    swaps etc. are going to be made up of multiple HistoryBaseEntry
    """
    event_identifier: bytes  # identifier shared between related events
    sequence_index: int  # When this transaction was executed relative to other related events
    timestamp: TimestampMS
    location: Location
    event_type: HistoryEventType
    event_subtype: HistoryEventSubType
    asset: Asset
    balance: Balance
    # location_label is a string field that allows to provide more information about the location.
    # When we use this structure in blockchains can be used to specify addresses for example.
    # currently we use to identify the exchange name assigned by the user.
    location_label: Optional[str] = None
    notes: Optional[str] = None
    # identifier for counterparty.
    # For a send it's the target
    # For a receive it's the sender
    # For bridged transfer it's the bridge's network identifier
    # For a protocol interaction it's the protocol.
    counterparty: Optional[str] = None
    identifier: Optional[int] = None
    # contains event specific extra data. Optional, only for events that need to keep
    # extra information such as the CDP ID of a makerdao vault etc.
    extra_data: Optional[Dict[str, Any]] = None
```

0xda601ce9aea84fadd1bd1e7ad12fb7ce7a59a852b6a10ea77c7b09feb7aaaf18 | 05/12/2020 09:40:43 GMT+1 | DETAILS >

GAS FEE lefteris.eth	< 0.02 ETH 6.20 $	Burned < 0.02 ETH in gas from lefteris.eth	
SWAP lefteris.eth	< 42.54 BADGER 192.20 $	Swap < 42.54 BADGER in 1inch-v2	
RECEIVE lefteris.eth	< 0.26 ETH 150.45 $	Receive < 0.26 ETH from 1inch-v2 swap	

0x9ff7a2f2c3aa54abb4b108be492473dd93414bc14c87adaf2b98063579df739d | 05/12/2020 09:33:40 GMT+1 | DETAILS >

GAS FEE lefteris.eth	< 0.01 ETH > 0.77 $	Burned < 0.01 ETH in gas from lefteris.eth	
APPROVAL lefteris.eth	BADGER	Approve 1.16e+59 BADGER of lefteris.eth for spending by 0x1111...F3ae	

0x33ce3f08fbcd477e85edfc894c966212e94930f5d560fe7fe48f2d4a33fd568e | 05/12/2020 09:30:55 GMT+1 | DETAILS >

GAS FEE lefteris.eth	< 0.01 ETH 2.21 $	Burned < 0.01 ETH in gas from lefteris.eth	
AIRDROP lefteris.eth	< 42.86 BADGER 193.65 $	Claim < 42.86 BADGER from badger airdrop	

Section 5

rotki's vision - abstraction layer

Opensource middleware that offers an abstraction layer for accounts, balances, PnL over multiple protocols and jurisdictions

Everyone is reinventing the wheel

Why do we need it?

- Everyone is reinventing the wheel
- Different protocols, different CEXes, different chains, different jurisdictions
- Impossible to keep up with everything as a single organization
- Maintain each single solution is a full time job

Solution to the problem

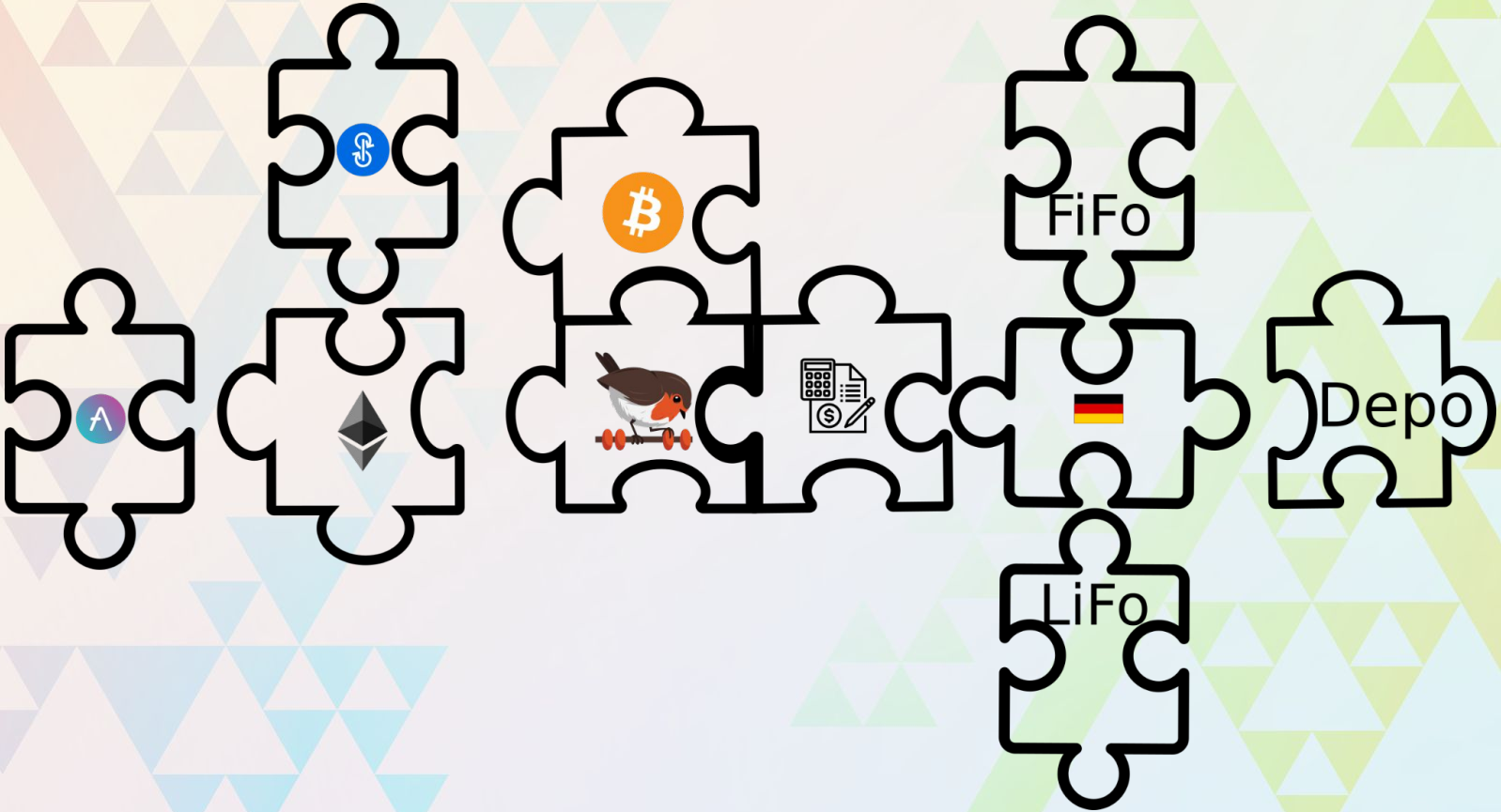
Problem: Everyone reinventing the wheel

Solution: An opensource platform/middleware maintained by a core team but with contributions by the entire industry and used by multiple projects

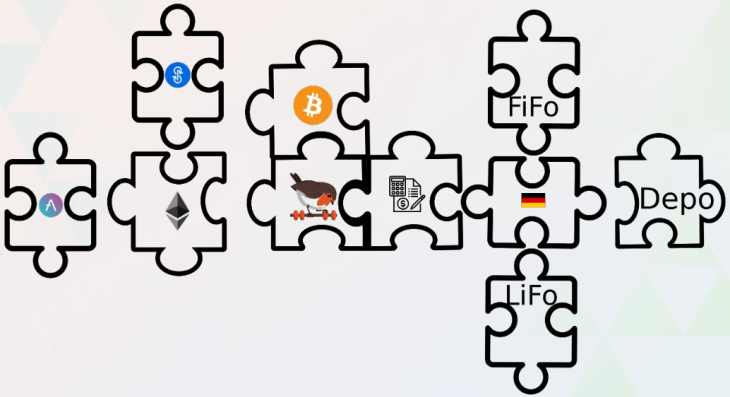
Problem: Different protocols, jurisdictions etc and impossible to keep up for a single organization

Solution: People incentivized from each chain/protocol/jurisdiction with the appropriate know/how implement the module with guidance from a core team

rotki middleware

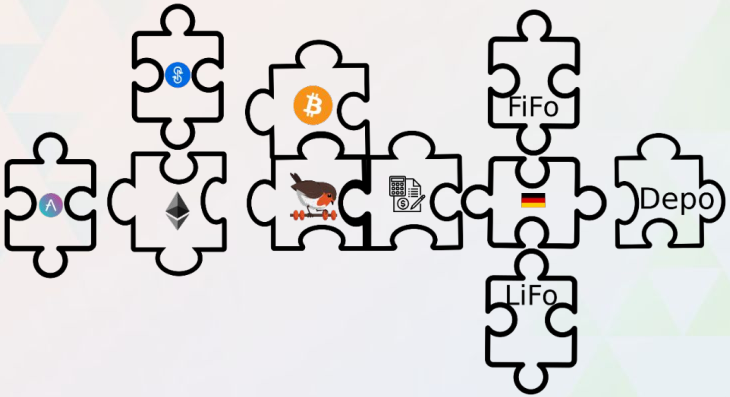


rotki middleware



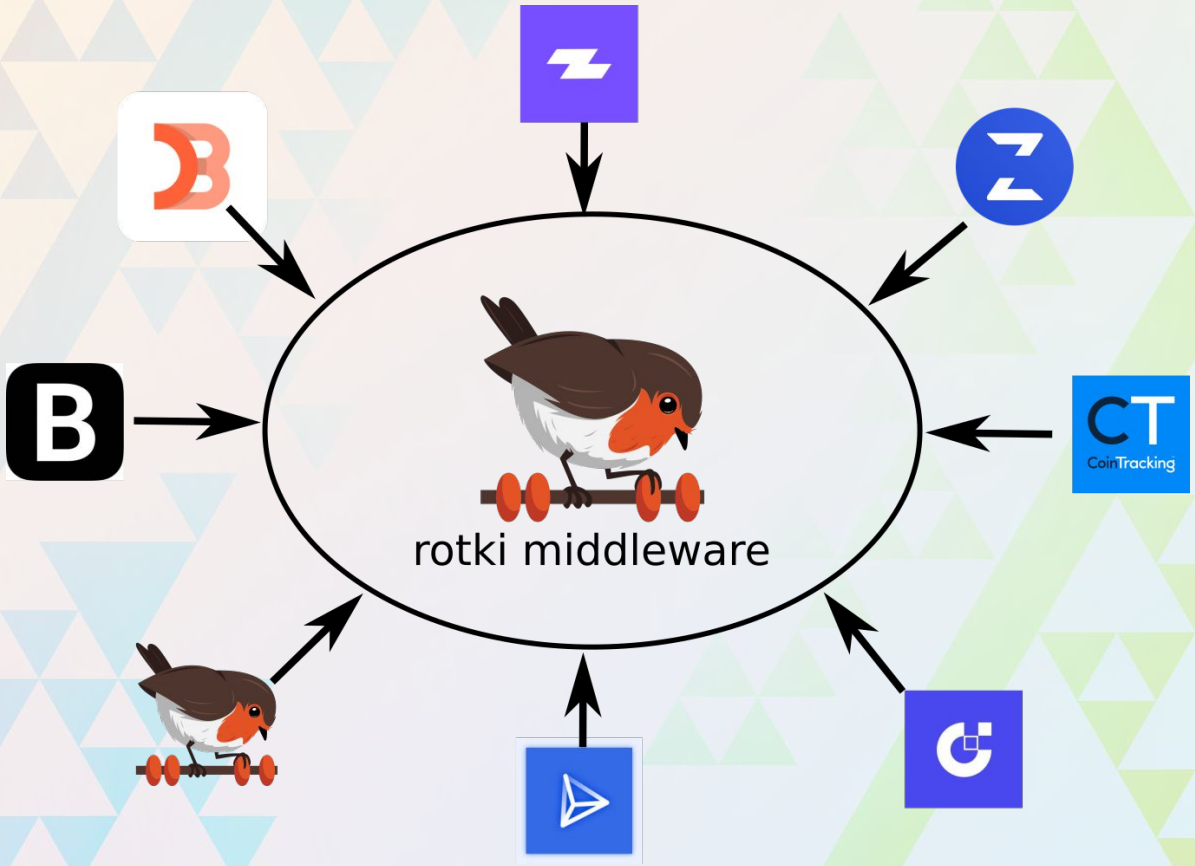
rotki middleware

rotki middleware



rotki middleware

rotki middleware



Requirement of such a platform

Needs to be

- Opensource
- Modular architecture
- Multilingual bindings

Incentivization

- Incentivize creator and maintainers of modules
- Incentivize the core team that builds and maintains it

How we got here

2017

I need to do my taxes.

Created some CLI scripts.

Later built a UI around them.

2020

rotki is founded as a German company.

Team of 2.

Maybe 200 users

2021

App grows. Many features.

Team of 3

2,000 users, 200 paying users.

2022

App matures more. Many integrations and features.

Team of 7

6,000 users, 550 paying users.

rotki middleware

Rotki's vision

Needs growth, time and funding.

Closing notes

- We are hiring: https://rotki.com/jobs
- Support us
  - Donate: https://gitcoin.co/grants/149/rotki
  - Buy premium: https://rotki.com/products
- Join our community:
  - Twitter: https://twitter.com/rotkiapp
  - Discord: https://discord.gg/aGCxHG7
- Interested in helping us grow? Talk to me: lefteris@rotki.com

Thank you!

Lefteris Karapetsas

Creator of bugs at rotki

lefteris@rotki.com

@lefterisjp

Here’s the timeline.

Event 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

Event 2

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

Event 3

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

Section 2 details with an image. Enter title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Section 1

Section 1 title here.

Section 1 title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

- Sollicitudin
- Consectetur
  - Condimentum
    - **Magna**
    - **Ligula**

Section 1 details with an image. Enter title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Enter your main point / statement here.

Section 1 details with a main point. Enter title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Section 2

Section 2 title here.

Section 2 title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

- Sollicitudin
- Consectetur
  - Condimentum
    - **Magna**
    - **Ligula**

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua.

- Sollicitudin
- Consectetur
  - Condimentum
    - **Magna**
    - **Ligula**

Section 2 details with an image. Enter title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Section 2 details with a main point. Enter title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Enter your main point / statement here.

Section 3

Section 3 title here.

Enter your main point / statement here.

Section 3 details with a main point. Enter title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Section 4

Section 4 title here.

Section 4 details with a main point. Enter title here.

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

Enter your main point / statement here.

Enter your main point / statement here.

Here’s the timeline.

Event 1

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

Event 2

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

Event 3

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam.

Thank you!

Your Name

Your title, your organization

email@emailaddress.com

@twitterhandle