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

This does not require an archive node.
The stateDiff command of parity tracing gives a per account difference of:

- Balance
- Code
- Nonce
- Storage
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
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.
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
A typical decoder

- 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

```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]):
        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 = chain_id_to_name.get(chain_id, f'Unknown Chain with id {chain_id}')
        amount = fromWei(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 = flat_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]
```
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: EventType
    event_subtype: EventSubType
    asset: Asset
    balance: Balance
    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
```
<table>
<thead>
<tr>
<th>TXHash</th>
<th>From</th>
<th>To</th>
<th>Gas Fee</th>
<th>Amount</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0xda601ce9aea84fadd1bd1e7ad12f7ce7a59a852b6a10ea177c7b09feb7aaaaf18</td>
<td><a href="mailto:left@is.eth">left@is.eth</a></td>
<td></td>
<td>&lt; 0.02 ETH</td>
<td>5.20 $</td>
<td>Burned &lt; 0.02 ETH in gas from <a href="mailto:left@is.eth">left@is.eth</a></td>
</tr>
<tr>
<td>0x9f7a2f2c3aa54abb4b108be492473dd93414bc14ce87adaf2b98063579df739d</td>
<td><a href="mailto:left@is.eth">left@is.eth</a></td>
<td></td>
<td>&lt; 2.64 BADGER</td>
<td>150.45 $</td>
<td>Receive &lt; 0.26 ETH from 1inchi-v2 swap</td>
</tr>
<tr>
<td>0x9f7a2f2c3aa54abb4b108be492473dd93414bc14ce87adaf2b98063579df739d</td>
<td><a href="mailto:left@is.eth">left@is.eth</a></td>
<td></td>
<td>&lt; 0.01 ETH</td>
<td>&gt; 0.77 $</td>
<td>Burned &lt; 0.01 ETH in gas from <a href="mailto:left@is.eth">left@is.eth</a></td>
</tr>
<tr>
<td>0x33ce3f08fbc4d7e894dfc89c96212e9492f5d560fe48f2d4a33fa33f560d</td>
<td><a href="mailto:left@is.eth">left@is.eth</a></td>
<td></td>
<td>APPROVAL BADGER</td>
<td></td>
<td>Approve 0x116e+59 BADGER of <a href="mailto:left@is.eth">left@is.eth</a> for spending by 0x111..F3ae</td>
</tr>
<tr>
<td>0x33ce3f08fbc4d7e894dfc89c96212e9492f5d560fe48f2d4a33fa33f560d</td>
<td><a href="mailto:left@is.eth">left@is.eth</a></td>
<td></td>
<td>&lt; 0.01 ETH</td>
<td>2.21 $</td>
<td>Burned &lt; 0.01 ETH in gas from <a href="mailto:left@is.eth">left@is.eth</a></td>
</tr>
<tr>
<td>0x33ce3f08fbc4d7e894dfc89c96212e9492f5d560fe48f2d4a33fa33f560d</td>
<td><a href="mailto:left@is.eth">left@is.eth</a></td>
<td></td>
<td>AIRDROP BADGER</td>
<td>193.65 $</td>
<td>Claim &lt; 42.86 BADGER from badger airdrop</td>
</tr>
</tbody>
</table>
rotki’s vision - abstraction layer
Opensource middleware that offers an abstraction layer for accounts, balances, PnL over multiple protocols and jurisdictions
Why do we need it?

Everyone is reinventing the wheel
- 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

Everyone is reinventing the wheel
Solution to the problem

**Problem:** Everyone reinventing the wheel

**Solution:** An open-source 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 to implement the module with guidance from a core team
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 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.
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 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 title 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 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.
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