**Compare Tracing API Outputs Across Clients**

**for**

[**ethereum-cdap**](https://github.com/ethereum-cdap)**/**[**cohort-one**](https://github.com/ethereum-cdap/cohort-one)

[**https://github.com/ethereum-cdap/cohort-one/issues/43**](https://github.com/ethereum-cdap/cohort-one/issues/43)

**Version draft.2**

**Prepared by**

**Jose Alfonso**

**Computer Engineer**

**DATE**

**November 2021**

**Table of Contents**

1. **Abstract**
2. **Motivation**
3. **Introduction**
   1. *Scope*
4. **Previous Work**
   1. *Work on Progress*
5. **JSON-RPC** 
   1. *API*
6. **Specification**
   1. *Scripts*
   2. *Data example*
   3. *Test Cases*
7. **Observations**
   1. **Comparison Table**
8. **Reading Material**
9. **Thanks!**
10. **Abstract**

A comparison table for Geth and Nethermind API traces is presented. There is a constant need for API standardization between clients. For this work, Blocks and Transactions APIs are compared and analyzed back and forward into the blockchain, observing data structure and value for returned objects.

1. **Motivation**

I was so lucky to find Ethereum - CDAP program !

Being a 48 year old Computer Engineer with more than 25 years graduated, but never worked in Computer Sciences (I was developed as an Oilfield and Oil Refinery expert), finding this program, has allowed me in a serious, professional, advanced and friendly environment to catch up with the latest, best and tip of the spear blockchain technology as Ethereum it is.

Getting involved in this project, ***Compare Tracing API Outputs Across Clients****,* understanding the need and the effort to create a standardized and consistent multiclient API method, allowed me to get familiar with the different APIs, to access and query Ethereum, its data structures and to present in a simple but precise table some Geth and Nethermind APIs compatibilities.

During this quest, I studied different EIPs, the Yellow Paper, Ethereum from inside and the most important, having met part of the team that makes all of this possible.

1. **Introduction**

With the inminent worldwide mass adoption of Blockchain thecnology and Cryptocurrencies, the fact that Ethereum is the biggest real-world day to day blockchain network, and the beginning of Consencus era happening in weeks from now, it is understandable that development and functionallity client's UI (APIs) evolved in some level of an unstandarized configuration.

The following work is a condensated approximation to show that cannonical states between all different clients is TRUE, and that it is just a matter, of a classic, standarization phase, normally required during the mega expansion states, that in our partiulary case, Ethereum is facing today.

* 1. **Scope:** This work is a response to Ethereum-cdap / cohort-one Issue #43. Compare Tracing API outputs across clients. @TimBeiko, @Piper

1. **Previous Work (not limited to:)**

* EIP 1474 Remote procedure call soecification. Paul Bochon, Erik Marks
* EIP 1898 Add 'blockHash' to JSON-RPC methods which accept a default block parameter. Charles Cooper
* EIP 234 Add 'blockHash' to JSON-RPC filter options. Micah Zoltu
* EIP-3155 EVM trace specification. Martin Holst Swende, Marius van der Wijden

1. **Work On progress (not limited to:)**

* Edge cases of Ethereum JSON-RPC endpoints 2.0. Tomasz Kajetan Stan'czak, Jared Doro
* Ethereum Trace Testing. Thomas Jay Rush

There are many other projects specifically dealing with this issue, but due to the generic reference that this work pretends to present the above mentioned should be more than enough reading material to catch up and understand the need for a final (if possible) EIP that once and for all standarizes this subject. Additional reading, if considered, would be listed at the end.

Note: MUST See: Nethermind, Tomasz Kajetan Stan'czak, Jared Doro. Work On Progress.

1. **JSON-RPC**

The Ethereum JSON-RPC is a collection of methods that all clients implement. This interface allows downstream tooling and infrastructure to treat different Ethereum clients as modules that can be swapped at will. (ethereum/execution-apis , Danny Ryan). From another perpesctive is a stateless, light-weight remote procedure call (RPC) protocol.

1. **API**

Application Programming Interface. In other words, an API is the messenger that delivers a request from point a to point b and then delivers a response from b to a.

Where the JSON-RPC APIs are for every Ethereum client a set of uniformed methods (data structures and the rules around their processing) that allow software applications, once connected to a node, interact with the Ethereum blockchain (reading blockchain data and/or sending transactions to the network).

1. **Specifications**
2. Scripts
3. Data Example
4. Test Cases

For the above data please consult: <https://github.com/JEAlfonsoP/evm-tracing-comparison>

1. **Observations (from mine limited position)**

1. **Comparison Table**

Clients: Geth, Nethermind

|  |  |  |
| --- | --- | --- |
| ***API*** | ***Geth*** | ***Nethermind*** |
| *getBlock: function()* | Supported | Not Supported |
| *getBlockByHash: function()*  *getBlockByNumber: function()* | Supported  Base Fee Per Gas, is first value returned | Supported  Includes in the output: Author (= miner)  Transaction Data (= input) |
| *getBlockNumber: function(callback)* | Supported | Supported |
| *getBlockTransactionCount: function()* | Supported | Same as getBlock. It has two queries one by Block Number and one by Block Hash |
| *getRawTransaction: function()* | Supported | Not Supported\* |
| *getTransaction: function()*  *getTransactionFromBlock: function()* | Supported | Supported. Includes in the output: Transaction Data (=input) |
| *getTransactionCount: function()* | Supported | Supported |
| *getTransactionReceipt: function()* | Supported | Not Supported |
| *debug.traceTransaction: function()* | Supported | Supported |
| debug.traceBlockByHash: function()  debug.traceBlockByNumber: function() | Supported | Supported |

*Clients generate in a different order the values returned for each query.*

1. **Reading Material:**

* Ethereum Yellow Paper
* EIP 1474, 1898, 234, 3155, 1559, 4345, and others
* <https://geth.ethereum.org>
* <https://docs.nethermind.io>
* <https://ethervm.io>
* <https://www.jsonrpc.org/specification>
* <https://github.com/DockBoss/Execution-API-Spec>

1. **Thanks!**

Tim Beiko, Piper, lightclient, holiman, ralexstokes, Pooja Ranjan, ligi