Slock.it - Programming Task
Ethereum Name Service Events Investigation

Submitted By: Chirag Mahaveer Parmar

Problem Statement

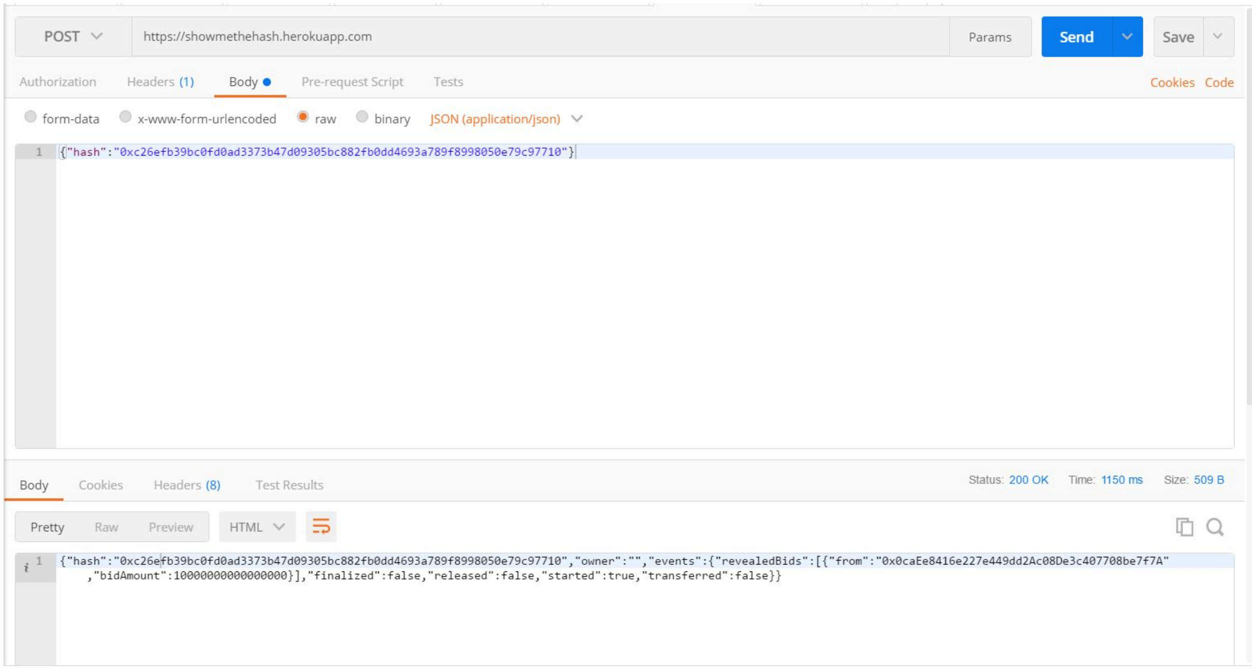
Write a test application to investigate or list or visualize the ENS contract events of the past (e.g. the last 1 or 2 days). focussing in particular on the bid revelation event: unsealBid (bytes32 _hash, uint256 _value, bytes32 _salt). Any programming language can be used to complete the given task, ideally, it would have some type of GUI (Windows, Linux, Android, iOS or web app), testing and documentation.

Approach

A little reconnaissance on the Ethereum Name Service reveals that the service is built on smart contracts. This points to the fact that various events of the ENS auctions are triggered via transactions made to the ENS-registrar and since transactions are public it becomes really easy to read the ENS contract events. Therefore, one could either subscribe to *newBlockHeaders* and parse the new blocks for ENS events or you could scrape through the blockchain for ENS events. Since the problem statement demands for events in the past, subscribing is not an option. Hence, an approach that scrapes the blockchain at regular time intervals made sense.

Alongwith scraping the blockchain, the program also stores the data as a *JSON* object locally to avoid loss of data upon program crash. So upon boot the application looks for a file named *data.json* and loads the previous data from it. If *data.json* does not exist then the program moves into the *first run sequence,* where it scrapes the blockchain from block *6060000* (three days prior to the day programming task was assigned) till the current block. After the *first run sequence* is completed the program is set to fire the scraping function, *processBlocks,* every minute.

Manual inspection of transactions made to the ENS-registrar would reveal that new transactions are executed for different events pointing to the same ENS hash. These transactions are used to create auctions, create new bids, reveal previous bids, finalize auctions etc. To know the exact purpose of a transaction the transaction input data must be decoded using the *contract ABI*. Since all events of the ENS are based on the unique hash of a ENS domain the data is categorised according to the ENS hash. Therefore, every time a new transaction is scraped and decoded, the program first checks if a record with the same ENS hash exists. If yes, then it appends information to that record otherwise it creates a new record.



https://showmethehash.herokuapp.com

JSON (application/json)

```
{"hash":"0xc26efb39bc0fd0ad3373b47d09305bc882fb0dd4693a789f8998050e79c97710"}
```

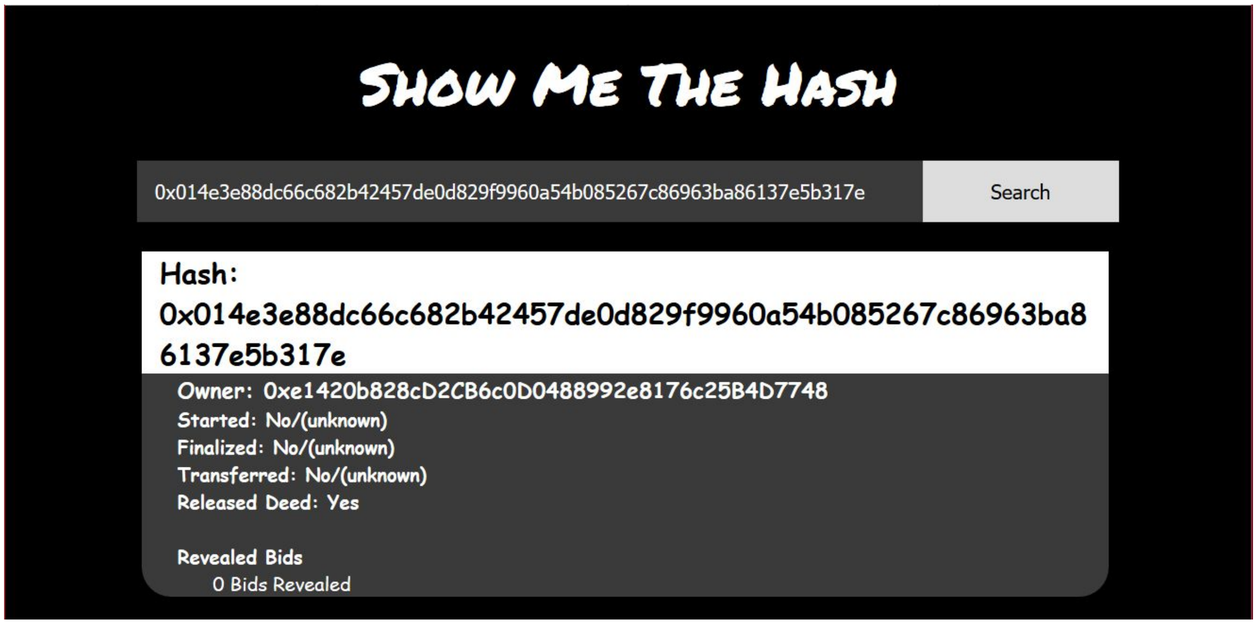
Status: 200 OK Time: 1150 ms Size: 509 B

```
{"hash":"0xc26efb39bc0fd0ad3373b47d09305bc882fb0dd4693a789f8998050e79c97710","owner":"","events":{"revealedBids":[{"from":"0x0caEe8416e227e449dd2Ac08De3c407708be7f7A"
    ,"bidAmount":10000000000000000}],"finalized":false,"released":false,"started":true,"transferred":false}}
```

POST request made to the Web App via Postman

The program also features a *expressjs* based web app that makes data handling easier. The web app is configured to accept *POST* request with a *JSON* body specifying the hash. Upon a *POST* request the program quickly searches through the entire *JSON* object for the hash specified and responds with another *JSON* object containing information about the ENS hash. The *expressjs* web app uses *body-parser* to read all incoming *POST* requests.

The front end of the application beautifies the data for the user. The front end features a search box that accepts the hash as input. A script on the front end then uses the entered hash to make a POST request to the back-end web app. It then dynamically creates HTML elements to present the data in a user-readable manner.



Show Me The Hash

0x014e3e88dc66c682b42457de0d829f9960a54b085267c86963ba86137e5b317e

Hash:
0x014e3e88dc66c682b42457de0d829f9960a54b085267c86963ba86137e5b317e

Owner: 0xe1420b828cD2CB6c0D0488992e8176c25B4D7748
Started: No/(unknown)
Finalized: No/(unknown)
Transferred: No/(unknown)
Released Deed: Yes

Revealed Bids
0 Bids Revealed

Beautified Data a.k.a Front-End

The back-end program is designed using node.js and is currently deployed on heroku with the endpoint https://showmethehash.herokuapp.com/ . The front-end is designed using basic HTML and CSS with little bit of Javascript in it. The front-end is currently deployed on my personal website, https://www.chiragparmar.me/shoemethehash.html

NOTE: an image showing the heroku deployment is attached at the end of the report

Testing

A video demonstrating the working of the web app is published on youtube. The link is attached here. The only contract event from which information cannot be extracted is the newBid event. This is because the Bid is sealed conveying no information about the amount or the auction the bid is placed on. Apart from that the only downside of the current deployment is that the free tier of heroku idles after 30 min of inactivity and limited uptime per month.

Appendix

Libraries

1. ***fs*** - used for file operations, mainly for storing and loading the JSON object
2. ***abi-decoder*** - used to decode the input data of the transactions
3. ***express*** - used for integrating REST API's with the web app
4. ***body-parser*** - used for parsing the incoming json POST request body.
5. ***web3*** - used to interact with the blockchain and fetch blocks. It used the infura mainnet provider (I had to use infura since I had limited disk space on my PC)

Functions

processBlocks(myaccount, startBlockNumber, endBlockNumber, callback)

The *processBlocks* function is the scraping function of the program. It accepts 3 parameters and a callback function.

Parameters:

1. ***myaccount***: specifies which account to look for. In our case it is the contract account of ENS-registrar
2. ***startBlockNumber***: specifies the starting-point of blockchain scraping. For the *first run sequence* it is 6060000.
3. ***endBlockNumber***: specifies the end-point of blockchain scraping. It is the most recent block of the blockchain.
4. ***callback***: specifies the callback function to be triggered upon completion of scraping.

The processBlocks function runs for loop iterating from the startBlockNumber to the *endBlockNumber*. The appropriate block is fetched and parsed for transactions pointing to the ENS-registrar. Input data of these transaction is then decoded using the *abi-decoder* and categorised according to the ENS hash.

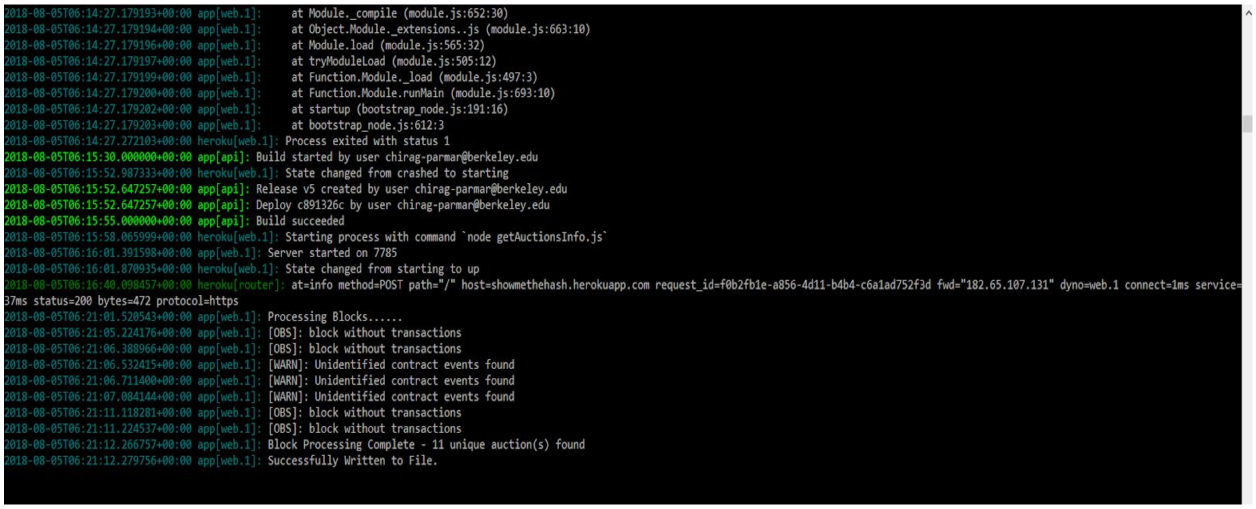
checkUniqueness(hash)

The *checkUniqueness* function accepts the ***ENS hash*** as a parameter and checks if a record of the hash already exists. If yes, it returns the index of the existing record otherwise it returns the index where the new record should be created

highestBidder(array)

The *highestBidder* function accepts an ***array*** as parameter. This array contains JSON objects containing information regarding the address of bidder and the bid amount. The function iterates through the array elements to find out the highest bidder. Returns the address of the highest bidder.

Heroku Deployment



```
2018-08-05T06:14:27.179193+00:00 app[web.1]:     at Module._compile (module.js:652:30)
2018-08-05T06:14:27.179194+00:00 app[web.1]:     at Object.Module._extensions..js (module.js:663:10)
2018-08-05T06:14:27.179196+00:00 app[web.1]:     at Module.load (module.js:565:32)
2018-08-05T06:14:27.179197+00:00 app[web.1]:     at tryModuleLoad (module.js:505:12)
2018-08-05T06:14:27.179199+00:00 app[web.1]:     at Function.Module._load (module.js:497:3)
2018-08-05T06:14:27.179200+00:00 app[web.1]:     at Function.Module.runMain (module.js:693:10)
2018-08-05T06:14:27.179202+00:00 app[web.1]:     at startup (bootstrap_node.js:191:16)
2018-08-05T06:14:27.179203+00:00 app[web.1]:     at bootstrap_node.js:612:3
2018-08-05T06:14:27.272103+00:00 heroku[web.1]: Process exited with status 1
2018-08-05T06:15:30.000000+00:00 app[api]: Build started by user chirag-parmar@berkeley.edu
2018-08-05T06:15:52.987333+00:00 heroku[web.1]: State changed from crashed to starting
2018-08-05T06:15:52.647257+00:00 app[api]: Release v5 created by user chirag-parmar@berkeley.edu
2018-08-05T06:15:52.647257+00:00 app[api]: Deploy c891326c by user chirag-parmar@berkeley.edu
2018-08-05T06:15:55.000000+00:00 app[api]: Build succeeded
2018-08-05T06:15:58.065999+00:00 heroku[web.1]: Starting process with command `node getAuctionsInfo.js`
2018-08-05T06:16:01.391598+00:00 app[web.1]: Server started on 7785
2018-08-05T06:16:01.870935+00:00 heroku[web.1]: State changed from starting to up
2018-08-05T06:16:40.098457+00:00 heroku[router]: at=info method=POST path="/" host=showmethehash.herokuapp.com request_id=f0b2fb1e-a856-4d11-b4b4-c6a1ad752f3d fwd="182.65.107.131" dyno=web.1 connect=1ms service=
37ms status=200 bytes=472 protocol=https
2018-08-05T06:21:01.520543+00:00 app[web.1]: Processing Blocks......
2018-08-05T06:21:05.224176+00:00 app[web.1]: [OBS]: block without transactions
2018-08-05T06:21:06.388966+00:00 app[web.1]: [OBS]: block without transactions
2018-08-05T06:21:06.532415+00:00 app[web.1]: [WARN]: Unidentified contract events found
2018-08-05T06:21:06.711400+00:00 app[web.1]: [WARN]: Unidentified contract events found
2018-08-05T06:21:07.084144+00:00 app[web.1]: [WARN]: Unidentified contract events found
2018-08-05T06:21:11.118281+00:00 app[web.1]: [OBS]: block without transactions
2018-08-05T06:21:11.224537+00:00 app[web.1]: [OBS]: block without transactions
2018-08-05T06:21:12.266757+00:00 app[web.1]: Block Processing Complete - 11 unique auction(s) found
2018-08-05T06:21:12.279756+00:00 app[web.1]: Successfully Written to File.
```

POST request body:

{“hash”:”<HASH TO BE INVESTIGATED>”}

Ex.

{"hash":"0xc26efb39bc0fd0ad3373b47d09305bc882fb0dd4693a789f8998050e79c97710"}

POST Request headers:

{‘Content-Type’:’application/json’}

Data Structures

ensData and ensDataArray

ensData is the JSON object that stores the information corresponding to a ENS hash. *ensDataArray* is an array of unique *ensData* JSON objects. *ensData* has structure like this,

```
var ensData = {
        "hash": '',
        "owner": '',
        "events": {
                "revealedBids": [],
                "finalized": false,
                "released": false,
                "started": false,
                "transferred": false,
        }
  }
```