DAO Caster
By Jules Lai

Motivation

One address to post all DAO updates

My motivation for choosing this project (called DAO Caster) was to make it easy for any DAO to post updates to one smart contract location address. This smart contract would be located at the same address across different Ethereum compatible blockchains, thus making it easy to find.

Hence one address to listen to all DAO updates

The analogy in my mind is of an information faucet where infrastructure can be built around it, for example different types of DAO discovery services. One such service might only be interested in good cause projects so would listen to DAO update events from DAO Caster smart contract and only add those to their own site the ones that met their criteria.

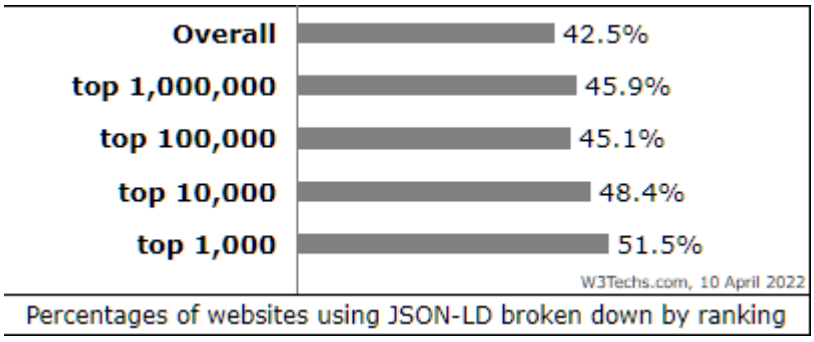
JSON-LD format to add context

The idea for the project started with ERC-4824 (a pull request, not yet merged to the Ethereum EIP master GitHub repo at the time of writing this). This outlined a way to utilise off-chain storage in JSON-LD format for the DAO data. This format is used heavily throughout e-commerce websites…

Usage of JSON-LD broken down by ranking

This diagram shows the percentages of websites using JSON-LD broken down by ranking. See technologies overview for explanations on the methodologies used in the surveys.

How to read the diagram:
JSON-LD is used by 42.5% of all the websites.
JSON-LD is used by 45.9% of all the websites that rank in the top 1,000,000.



Percentages of websites using JSON-LD broken down by ranking

JSON-LD is a powerful format yet human readable

JSON-LD is a powerful structured link data framework that provide semantic meaning via schemas to data., thus ambiguity of property names are removed. For example, a name property for a person, might be their first and last name or could be a login name etc. Its meaning can be found by looking up the schema reference where the property is described in text.

JSON-LD has an advantage that it is human-readable and stored off-line, can be very cost efficient. This format opens up all sorts of opportunities of interoperability between applications.

The Task

Security

I pointed out on the Ethereum Magicians forum, off-chain storage could be anywhere, for example on centralised storage. In other words, the data may be made up/altered by malicious actors.

Trust

JSON-LD was not built with blockchains in mind. In the traditional web2 world, trust is relied on a few actors, for example with big multinational companies like Google or Amazon. In web3 world, decentralisation works best in a system when its trustless; otherwise, there are too many actors to trust which in a lot of cases is not practical.

EIP-3722 Poster

User Asgeir on the Ethereum Magicians Forum suggested using EIP-3722 to post out the DAO updates. EIP-3722 Poster is designed for posts in text to be emitted as events from a singleton contract that can be deployed on any EVM compatible blockchain. It is designed for social media applications rather than DAOs and doesn't solve the problem of lack of security of off-line JSON-LD files for storing the DAO data.

My proposal

I proposed a combination of both poster and ERC-4824 to help alleviate security concerns around off-chain storage. With EIP-3722 every emitted post includes the msg.sender address that cannot be tampered as it's bulti into Solidity blockchain infrastructure. My solution is to break up the JSON-LD file into many smaller files. Each event emitted would then contain both msg.sender and the hash link to the data.

Verification

As all DAO updates emitted include the hash link to the data, it can be proved that whatever it is pointing to has not been tampered with by verifying the hash.

Timestamps

Yet more information can be garnered from events; what block number they were emitted from. Thus, every DAO post, a timestamp can be found for when the update was emitted from its block number. DAO data can be retrieved from specific block number ranges, for example the last 6000 blocks (approx. the last 24 hours from Ethereum mainnet)

Technical way to a fixed smart contract address

It is straightforward to fix an address. Indeed, the address can be precomputed even; calculated from the hash of the contract byte code and a few other parameters, including a salt. Then contract deployed via CREATE2.

The Code

I wrote three applications to demonstrate my project idea.

DAO Caster

This is the Solidity dapp, a single smart contract that posts out events of DAO updates when called.

```
event DAOUpdate(
    address indexed sender,
    address indexed dao,
    string name,
    string description,
    string governanceURI
);

event MemberUpdate(
    address indexed sender,
    address indexed dao,
    address indexed member,
    string memberURI
);

event ProposalUpdate(
    address indexed sender,
    address indexed dao,
    string indexed proposalId,
    string proposalURI
);

event ActivityLogUpdate(
    address indexed sender,
    address indexed dao,
    string indexed activityId,
    string activityLogURI
);
```

All events emit the msg.sender (as sender) and the dao address. The uri strings are assumed to be based off the hash of the content it points to. DAO discovery services should have a white list of acceptable link formats so that content can be verified that it hasn't been tampered by malicious actor.

DAO Charity

A DAO dapp that is based from OpenZeppelin's Governance smart contract. This contract works out of the box with OpenZeppelin's ERC20 Vote contract.

With DAO Charity, I am able to add new proposals; including the necessary data fields for smart contract execution of a successful proposal. The UI allows individual smart contract function calls and their order to be rearranged before commit. DAO Charity then calls the DAO Caster smart contract to emit the DAO events for the new proposal (both proposalUpdate and ActivityLogUpdate).

Events also emitted include memberUpdate, when a new member joins the DAO by buying the governance token.

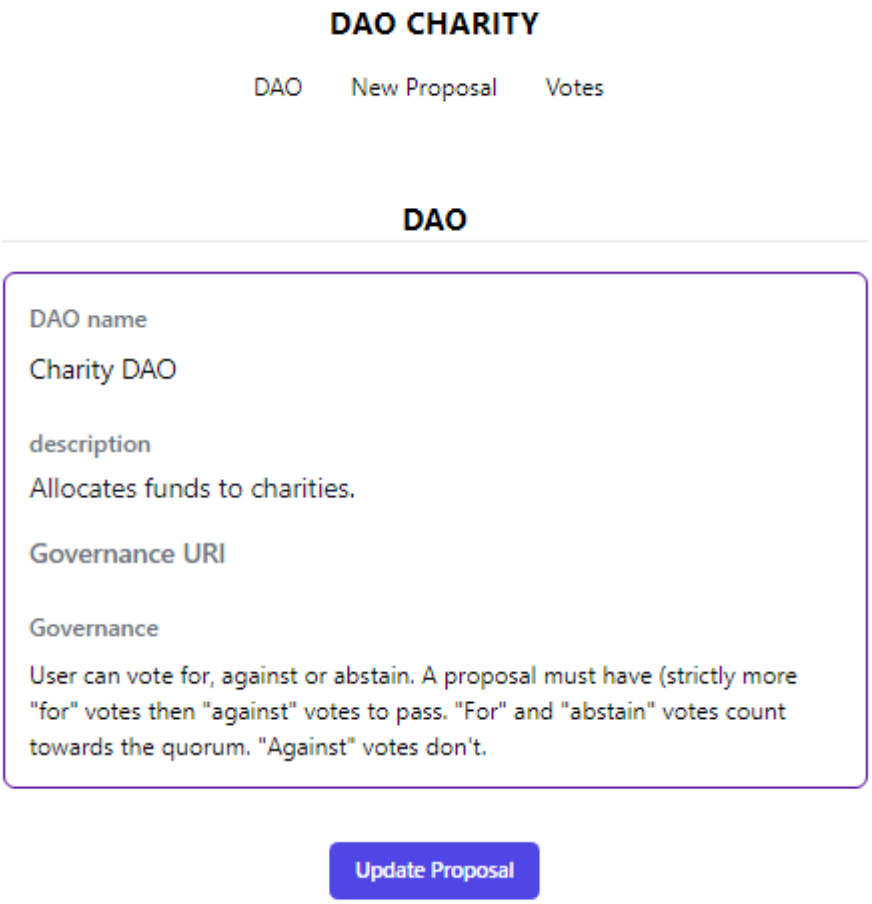
DAO Discovery

This application simulates a DAO discovery service that uses the events emitted by DAO Caster for, for example curation purposes. DAOs can be searched for as well as their proposals, member list and activity logs. Events from DAO Caster can be restricted between block number ranges. A subset of events from DAOs can be retrieved for a list of blockchain addresses, as well as other search criteria such as proposal ids, activity ids, member account addresses.

Results

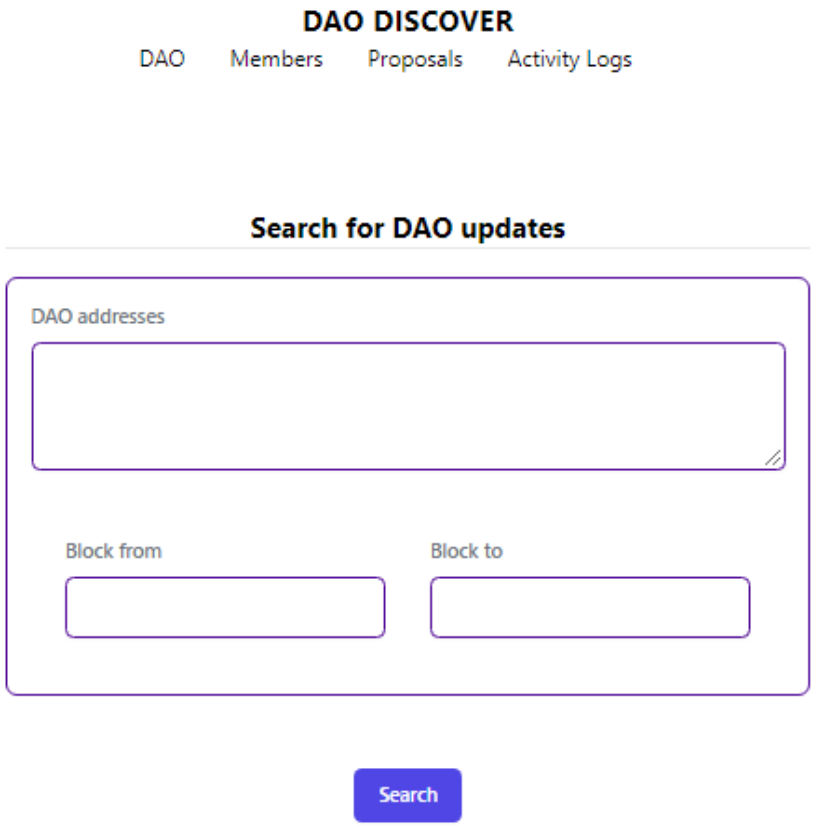
I set up some dummy DAO details called DAO Charity.

I then called the DAO Caster to emit the DAO details as an update.



Then I set up a dummy DAO discovery service that was able to retrieve the DAOUpdate event and its JSON-LD file. Noticed how the sender account address (msg.sender) is retrieved as it was sent out by the DAOUpdate. Additional information can be obtained from the event, such as the block number the event was emitted.

The discover service can search by DAO addresses and restricted to between block numbers. Keeping it blank means to retrieve everything without filters. In this text case, just one event was emitted so far.



Next, I set up a new proposal.

DAO CHARITY

DAO New Proposal Votes

New Proposal

Title

My Propsal1

Proposal ID

48ea7262-f1b4-48b0-8d58-490b9945dee8

Content URI

https://ipfs.io/ipfs/QmNcKMfeJx6kd2cg4ZPfzjuQNdTLztPBGfSaAA9fyi...

12/04/2022 14/04/2022

Calls

Call ↑ ↓ Delete

Operation

delegate call

from

0x3044B131BC986404b6a5796...

to

0x54940cDa2cDDF91d5F6B120...

value

500

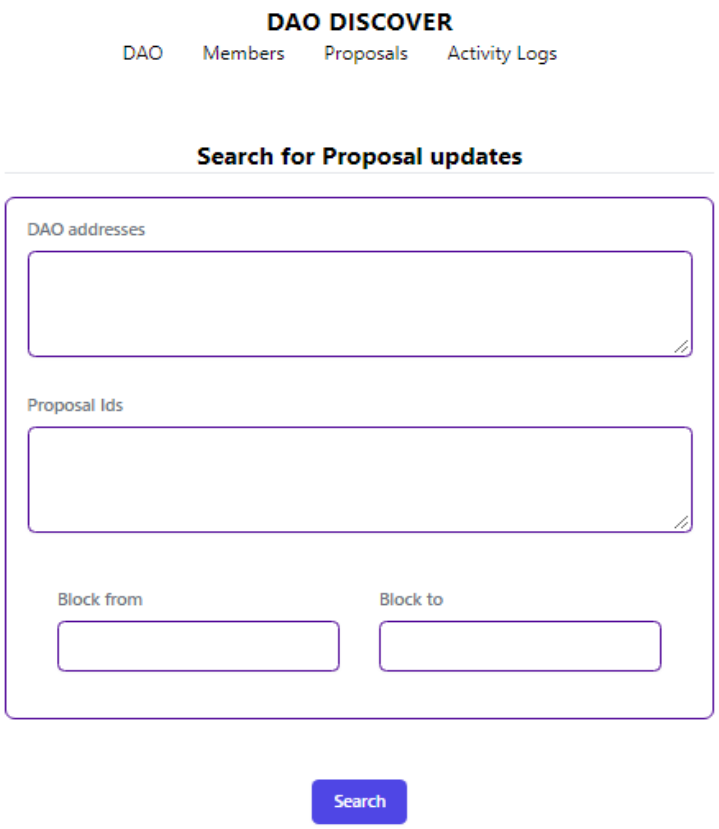
call data

3fa18c8710246e0ef19973427ad91350ee01f9a03e51ec436bf4b2bdf8019...

New Call

Cancel Proposal Add Proposal

With DAO discovery service, able to retrieve the proposal details.



Proposal updates found

sender: 0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC block: 5


```
{
  "@context": "http://www.daostar.org/schemas",
  "dao": "0x778062419865C905988a5fC8836f7B893c30e52E",
  "@id": "48ea7262-f1b4-48b0-8d58-490b9945dee8",
  "@type": "proposal",
  "name": "My Propsal1",
  "contentURI": "https://ipfs.io/ipfs/QmNcKMfeJx6kd2cg4ZPfzjuQNdTLztPBGfSaAA9fyideW5",
  "status": "Pending",
  "startTimestamp": 1649635200,
  "endTimestamp": 1649635200,
  "calls": [
    {
      "operation": "delegate call",
      "from": "0x3044B131BC986404b6a579666291BC6D618F8b38",
      "to": "0x54940cDa2cDDF91d5F6B120548163B4a1b0d4e6a",
      "value": "500",
      "data": "3fa18c8710246e0ef19973427ad91350ee01f9a03e51ec436bf4b2bdf801958a"
    }
  ],
  "isMember": false
}
```


Warning: Sender was not a member!

Noticed how the JSON-LD file got flagged with "Warning Sender was not a member". This was achieved by using the sender account address sent out by the DAO Caster ProposalUpdate event to look for MemberUpdate events that had that sender account address. None were found for the block number the proposal was submitted in.

I also added extra details, as seen in the ActivityLog JSON-LD file:

There are the DAO address which was required for the security checks. Now that the JSON-LD files have sender details and the URI hashed links sent from the emitted events, the JSON-LD files can be checked and verified. I believe this makes the JSON-LD files in this web3 world much more secure and practical to use. Thus increases the security of JSON-LD files that link to them.

DAO DISCOVER

DAO Members Proposals Activity Logs

Search for Activity Log updates

DAO addresses

Activity Ids

Block from

Block to

Search

Activity Log updates found

sender: 0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC block: 6


```
{
  "@context": "http://www.daostar.org/schemas",
  "dao": "0x778062419865C905988a5fC8836f7B893c30e52E",
  "@id": "48ea7262-f1b4-48b0-8d58-490b9945dee8",
  "@type": "activity",
  "proposal": {
    "@id": "48ea7262-f1b4-48b0-8d58-490b9945dee8",
    "@type": "proposal",
    "activity": "submission"
  },
  "member": {
    "@type": "EthereumAddress",
    "address": "0x3C44CdDdB6a900fa2b585dd299e03d12FA4293BC"
  }
}
```


Warning: Sender was not a member!

Conclusion

The DAO Charity and Discovery applications show that it is easy to build infrastructure and tools around DAO Caster. This is really scratching the surface. This project shows that the advantages of blockchain can be married with the power of semantics of data offered by JSON-LD.

Blockchain is used to offer tight cryptographic security to off-line structured JSON-LD files. Each of these files can be linked together (via for example their id fields), with the knowledge that the underlying data is secure, thus making the whole secure.

JSON-LD schemas can offer the way for decentralised applications to interop with other decentralised and/or centralised applications and websites in a very gas efficient manner as the data is stored off-chain yet grounded with the security from the blockchain.

For DAOs where a lot of operations are performed off-chain, this can now be performed on-chain with the storage of data to secured JSON-LD files off-chain, whilst saving on the gas fees.

References:

ERC-4824 Decentralized Autonomous Organizations
AuthorsJoshua Tan (@thelastjosh), Isaac Patka (@ipatka), Ido Gershtein (ido@daostack.io), Eyal Eithcowich (eyal@deepdao.io), Michael Zargham (@mzargham), Sam Furter (@nivida)

EIP-3722: Poster (A ridiculously simple general purpose social media smart contract.)
Author Auryn Macmillan

Usage of JSON-LD broken down by ranking
https://w3techs.com/technologies/breakdown/da-jsonld/ranking