CRYPTO-SPATIAL : AN OPEN STANDARDS SMART CONTRACTS LIBRARY FOR BUILDING GEOSPATIALLY ENABLED DECENTRALIZED APPLICATIONS ON THE ETHEREUM BLOCKCHAIN

BENAHMED DAHO Ali[1] *

[1] Geodetic Sciences and Topographic Works Engineer, Ain Temouchent, Algeria- bidandou@yahoo.fr

Commission VI, WG VI/6

KEY WORDS: Ethereum Blockchain, Decentralized Applications, Smart Contracts, Ethereum, IPFS, OrbitDB, Land Administration, OGC Open Standards.

ABSTRACT:

Blockchain is an emerging immature technology that disrupt many well established industries. In this contribution we present a smart contracts library, named Crypto-Spatial, written with Solidity for the Ethereum Blockchain and designed to serve as a framework for geospatial data decentrilized permanent storage and retrieviewal.Blockchain is an emerging immature technology that disrupt many well established industries. In this contribution we present a smart contracts library, named Crypto-Spatial, written with Solidity for the Ethereum Blockchain and designed to serve as a framework for geospatial data decentrilized permanent storage and retrieviewal.

1. INTRODUCTION

1.1 The Ethereum blockchain paradigm

Ethereum blockchain can be viewed as a transaction-based state machine: we begin with a genesis state and incrementally execute transactions to morph it into some final state. It is this final state which we accept as the canonical "version" of the world of Ethereum. The state can include such information as account balances, reputations, trust arrangements, data pertaining to information of the physical world; in short, anything that can currently be represented by a computer is admissible. Transactions thus represent a valid arc between two states; the 'valid' part is important. A valid state transition is one which comes about through a transaction. Formally:

$$\boldsymbol{\sigma}_{t+1} \equiv \Upsilon(\boldsymbol{\sigma}_t, T) \tag{1}$$

where Υ is the Ethereum state transition function. In Ethereum, Υ, together with $\boldsymbol{\sigma}$ are considerably more powerful than any existing comparable system; Υ allows components to carry out arbitrary computation, while $\boldsymbol{\sigma}$ allows components to store arbitrary state between transactions.

Transactions are collated into blocks; blocks are chained together using a cryptographic hash as a means of reference. Blocks function as a journal, recording a series of transactions together with the previous block and an identifier for the final state. They also punctuate the transaction series with incentives for nodes to *mine*. This incentivisation takes place as a state-transition function, adding value to a nominated account. Formally, we expand to:

$$\begin{aligned}
\boldsymbol{\sigma}_{t+1} &\equiv \Pi(\boldsymbol{\sigma}_t, B) && (2)\\
B &\equiv (..., (T_0, T_1, ...), ...) && (3)\\
\Pi(\boldsymbol{\sigma}, B) &\equiv \Omega(B, \Upsilon(\Upsilon(\boldsymbol{\sigma}, T_0), T_1)...) && (4)
\end{aligned}$$

Where Ω is the block-finalisation state transition function; B is this block, which includes a series of transactions amongst some other components; and Π is the block-level state-transition function.

This is the basis of the blockchain paradigm, a model that forms the backbone of not only Ethereum, but all decentralised consensus-based transaction systems to date.

In term of implementation, there are many choices of blockchains: over 200 Bitcoin variants, Ethereum and other permissioned blockchains. To meaningfully compare them, [Ji Wang and al.] identified four abstraction layers found in all of these systems. (1) The consensus layer contains protocols via which a block is considered appended to the blockchain. (2) The data layer contains the structure, content and operations on the blockchain data. (3) The execution layer includes details of the runtime environment support blockchain operations. Finally, (4) the application layer includes classes of blockchain applications.

The Crypto-Spatial framework, described in this contribution, is designed for the Ethereum Blockchain and propose a set of smart contracts for the execution layer and a cheap geometry storage solution on IPFS for the application layer.

1.2 Ethereum smart contracts security issues

In this section we systematize the security vulnerabilities of Ethereum smart contracts. We group the vulnerabilities in three classes, according to the level where they are introduced (Solidity, EVM bytecode, or blockchain). Further, we illustrate each vulnerability at the Solidity level through a small piece of code. All these vulnerabilities can be (actually, most of them have been) exploited to carry on attacks which e.g. steal money from contracts. Table 1 summarizes our taxonomy of vulnerabilities, with links to the attacks illustrated in Section 4.

*Corresponding author

1.3 IPFS and OrbitDB

To store cheaply Geometry data OrbitDB storage based on IPFS technology is used in combination with on-chain smart contracts.

IPFS is a distributed file system which synthesizes successful ideas from many peer-to-peer sytems, including DHTs, BitTorrent, Git, and SFS. The contribution of IPFS is simplifying, evolving, and connecting proven techniques into a single cohesive system, greater than the sum of its parts. IPFS presents a new platform for writing and deploying applications, and a new system for distributing and versioning large data. IPFS could even evolve the web itself. IPFS is peer-to-peer; no nodes are privileged. IPFS nodes store IPFS objects in local storage. Nodes connect to each other and transfer objects. These objects represent files and other data structures. [Benet, J. (2014). IPFS-content addressed, versioned, P2P file system. arXiv preprint arXiv:1407.3561.]

OrbitDB. It is a distributed, peer-to-peer database that is built on top of IPFS [60]. OrbitDB supports various kinds of databases including key-value and log databases. This makes OrbitDB an excellent choice for the decentralized prototype.

The bids are stored in GlobalDB as JSON objects for simple parsing to extract the data. The databases also have listeners implemented that triggers when the databases are replicating. Thereafter, the listeners trigger the user interface to update. This ensures that the users will always have the most recent bids available. In figure 3.7 the database architecture is shown. [OrbitDB. OrbitDB. url: https://github.com/orbitdb/orbit-db/ (visited on 2018-05-12).]

1.4 Blockchain business models

The maximum paper length is restricted to 8 pages. Invited papers can be increased to 12 pages.

1.5 Blockchain developer tools

The maximum paper length is restricted to 8 pages. Invited papers can be increased to 12 pages.

2. THE CRYPTO-SPATIAL FRAMEWORK

2.1 Title

The title should appear centered in bold capital letters, at the top of the first page of the paper with

FOAM Protocole White paper

OGC Open standards

Geodesic Discrete Global Grid Systems

Desgin class diagram

Implementation - Solidity

3. DECENTRALIZED LAND ADMINISTRATION

Type text single-spaced, **with** one blank line between paragraphs and following headings. Start paragraphs flush with left margin.

3.1 The workflow

Major headings are to be centered, in bold capitals without underlining, after two blank lines and followed by a one blank line.

ACKNOWLEDGEMENTS (OPTIONAL)

Acknowledgements of support for the project/paper/author are welcome.

APPENDIX (OPTIONAL)

Any additional supporting data may be appended, provided the paper does not exceed the limits given above.

Revised May 2019