A BLOCKCHAIN-BASED SUPPORT TO SAFEGUARDING THE CULTURAL HERITAGE

Mariano Basile, Gianluca Dini
Dept.of Ingegneria dell'Informazione
University of Pisa
Pisa, Italy
[name].[surname]@unipi.it

Andrea Marchetti, Clara Bacciu and Angelica Lo Duca
Institute of Informatics and Telematics
National Research Council
Pisa, Italy
[name].[surname]@iit.cnr.it

Abstract – The recordkeeping process in an accounting system usually produces digital archives. In case of a collection of structured data, a digital archive is typically maintained as a *centralized* database system. This solution constitutes a security and trust single-point-of-failure. We have faced this problem in the case of digitalization of "*Catalogo Generale dei Beni Culturali*" which responds to protection and promotion purposes of the Italian national heritage. In this work we propose a *blockchain-based architecture* for a digital archive for the *safeguarding of the cultural heritage*. The proposed solution is decentralised and efficient, both from an economic and performance viewpoint.

I. INTRODUCTION

Information is one the most valuable assets of every organization today. Collections of digital information are usually referred to as *digital archives*. In case of a collection of structured, interrelated data, a digital archive is basically maintained as a *database* and turns out to be usually under the control of a single entity.

However, this approach raises several security issues. First of all, it implies that customers trust organizations not to maliciously exploit the data and to effectively keep them secure. Furthermore, having data operated by private authorities makes them prone to security attacks and to become a single-point-of-failure. This was particularly common in Web 1.0.

Things did not get much better with Web 2.0. Cloud providers offer cheap, sometimes even free, scalable services, but scalability, and fault-tolerance, come at the cost of providers' owning the content, profiling users and collecting huge amount of data about user activity, i.e., keeping record of which content gets accessed, by whom and when, so to turn it into a profit. This has made surveillance and censorship easier.

We faced with the above issues when we considered the "*Catalogo Generale dei Beni Culturali*" (General Catalogue for Cultural Heritage), hereafter the "catalogue", under the control of the *Ministry of Cultural Heritage and Activities and Tourism*. This database plays a significant role in the spreading of knowledge and in the promotion of the Italian national heritage. At the same time, it helps to manage crimes and natural disasters that may affect the national heritage.

In this work we propose a blockchain-based architecture for the catalogue. The proposed solution is *decentralized*, so overcoming the security and trust issues, and efficient, both from an economic and performance viewpoint. The proposed solution allows only authorized principals to record items at risk of theft, forgery, or natural disaster. At the same time, it ensures authenticity, privacy, integrity and long-term data availability.

We have chosen a *blockchain*-based architecture in order to guarantee three out of four of the above properties, namely, authenticity, integrity and availability. The privacy requirement has

been achieved at application level by storing data in its encrypted form.
For performance and cost reasons, we also adopted an *off-chain* data storage strategy. Actually, the straightforward on-chain approach is neither practical nor economical in case of storage of large amount of data.
Finally, it is worth highlighting that the solution still requires a database since we are dealing with structured data on which searches still have to be performed. From this standpoint, it might seem that we are back where we started. Actually it is not the case. Indeed, the solution has been designed to be used by one entity and one entity only, that is the *State Police*.
The remainder of the paper is organized as follows: in Section II we discuss related work. In Section III we explain the way we created a (reduced-size) clone of the "Catalogo Generale dei Beni Culturali". In Section IV we present the reasons why using a blockchain-based approach for digital archives. In Section V we first introduce a straightforward solution (and its limitations) and then we describe the specific solution we propose and the assumption we made. In Section VI we present an early prototype. In Section VII we evaluate the cost of our scheme. Finally, Section VIII concludes the paper.

II. RELATED WORK

The problem of managing records through a blockchain has been largely investigated during the last few years. In her paper, Lemieux proposes a classification of blockchain applications (Lemieux 2017), based on which information is stored in the blockchain: a) *mirror type*, b) *digital record type*, c) *tokenized type*.

In the mirror type, the blockchain serves as a mirror, which stores only records fingerprints. The complete information of a record is stored into an external repository and the blockchain is used only to verify records integrity. In (Garcia 2017) the authors describe a first implementation of a decentralized metadata database, based on the combination of the blockchain and IPFS technologies. In their paper Liang et. al. describe ProvChain (Liang 2017), a system which guarantees data provenance in cloud environments. Vishwa et. al. (Vishwa 2018) illustrate a blockchain-based framework, which guarantees copyright compliance of multimedia objects by means of smart contracts.

In the digital record type, the blockchain is used to store all the records in the form of smart contracts. In (Bhowmik 2017) the authors illustrate a distributed and tamper-proof framework for media. Each media is represented by a watermark, which is firstly compressed and then stored into a blockchain. Approved modifications to media are stored in the blockchain thus preventing tampering. In (Galiev 2018) the authors describe Archain, a blockchain-based archive system, which stores small-sized records. Multiple roles are defined in the system, thus allowing records creation, approval and removal.

In the tokenized type, records are stored in the blockchain and they are linked to a cryptocurrency. This constitutes an innovative case, where the literature is not consolidated yet. Adding, updating or removing a record has a cost. An example of this type of blockchain is represented by the Ubitquity Project (http://ubitquity.io/brazil_ubitquity_llc_pilot.html), which records land transactions on behalf of companies and government agencies.

III. THE SAFEGUARDING OF THE CULTURAL HERITAGE

The General Catalogue for Cultural Heritage [ ] is a centralized database which collects data about Italian cultural goods. The catalogue is aimed at identifying and describing all

cultural assets for which a specific artistic, historical, archeological or anthropological interest has been recognized.
Cultural goods are properly described by means of specific standards whose definition is led by the "*Central Institute for Cataloguing and Documentation*" (ICCD).
Around May 2016, the ICCD has launched a project called *Openiccd* [,] which aims at sharing, among other things, raw data about cultural goods being catalogued. Open-data are available for each Italian region and refers to the following two *typologies*: "archeological exhibits" and "artworks". Each typology is described according to the so-called standard "*schede di catalogo*". In the most general terms we can state that these are basically sets of of attributes which take into account peculiarities and intrinsic key features of each typology of cultural property.
Initially, for the sake of simplicity, we have limited our catalogue to the ICCD open-data related to archeological exhibits and artworks of Tuscany. However, this amounts to 75.662 records with 51.977 images in the form of URLs and 46 attributes for each record and thus it constitutes a meaningful exercise.
In order to store these data, as we were aiming at a decentralized solution, we chose a *distributed* database system. The choice was furthermore restricted to *NoSQL* databases because of their performance at scale. In this respect, we selected the Elasticsearch [ ] "*document* store"-type (NoSQL) database.
In Elasticsearch, any information item is stored into an *index* which is a collection of documents that have similar characteristics. Therefore, for the storage of Tuscany ICCD open-data we defined two indices, one for artworks and the other for archeological exhibits.

IV. A BLOCKCHAIN-BASED APPROACH FOR DIGITAL ARCHIVES

The reason why using a blockchain-based architecture for digital archives lies in its core value proposition: a shared source of true, non-repudiable and uncensorable system. In other terms a layer of information that everybody trusts. The truth results to be shared because it is distributed among all the nodes in the network. This means that every participant stores its own copy.
Moreover, being intrinsically distributed, having all the information replicated on all the nodes of the network, it gives the guarantee that data are preserved over the time.
Furthermore, it guarantees also decentralization, in the sense that data are not hosted and managed by a single central authority.
Finally, it provides authenticity and integrity because of its intrinsic nature, based on cryptography.
There are different blockchain implementations around [, , ,]. We have considered *Ethereum*, because of its nature of being a public and a Turing complete blockchain.
The presentation of our solution comes in two steps. In the first step, Section A, we propose a straightforward solution that, unfortunately, it is not practically feasible. In the second step, we propose our solution.

A. A straightforward but unfeasible solution

By means of an Ethereum smart contract, an archive can operate programmatically without the need for ownership or control by a particular entity. Furthermore, Ethereum allows storage on the blockchain. This means that smart contracts have their own storage. Therefore, a straightforward solution simply consists in storing data that comprise an archive

within the storage of a smart contract.
However, exactly because all computations, including the storage, have to be replicated on each node, storage of data is really expensive in Ethereum.
In addition, as soon as the size of an archive starts being not negligible, the time required in order to retrieve and store the new archive contents may lead to high latencies.
Things get even worse if we consider that accounting is required every time a new operation alters an archive's current state.
Last but not least, withdrawing that data from the blockchain, at a later time, is also relatively slow.
For all these reasons this solution is unfeasible in practice.

B. The Proposed Solution

Fig. 1. Off-chain operations storage + IPFS links accounting using Ethereum logs.

The scheme in Fig. 1 addresses both the performance and the economic issues described in the previous subsection.
The performance concern has been solved accounting, each time, only the specific operation (insert/update/delete) and the associated data, instead of the whole archive contents.
A JSON object encoding the operation and its data is created each time. In this regard, we have specialized the solution according to our specific use case. In particular, data has been accounted exactly as described in the standard related to the specific typology of cultural good considered. Moreover, if required, data is also encrypted. The JSON data-interchange format has been chosen because the basic unit of information that can be indexed in Elasticsearch, a document, is expressed in JSON.
Furthermore, as already mentioned in Section I, this kind of accounting has been performed using an off-chain data storage approach by means of a content addressed, versioned, peer-to-peer, distributed file system, named *InterPlanetary File System* (IPFS) [ ].
This has enabled storing on the Ethereum blockchain just the immutable, permanent IPFS links in the form of the cryptographic hash of each JSON object, i.e just a fixed amount of data (32 bytes). This has greatly reduced storage costs.
This idea has been further explored so as to end up with the following efficient solution: in the end no smart-contract storage has been used at all, instead IPFS links have been stored in Ethereum *logs*.
Moreover, the following assumption has been made: nodes in the Elasticsearch cluster do not suffer from Byzantine faults, i.e., they can crash but they cannot arbitrarily and maliciously deviate from their specification. The reason is that nodes will be under the supervision of the State Police and, for this reason, they are considered as part of the Trusted Computing Base.
This further specialized the solution. Specifically, it allowed us to reflect each insert/update/delete operation on Elasticsearch right before propagating it off-chain, letting

search operations involve only Elasticsearch. In other terms, the latter acted as a front-end cache system.
Under this assumption, Elasticsearch indices get reconstructed only in the case of nodes crashes within the Elasticsearch cluster.
In order to restore only the specific Elasticsearch indices involved in the fault, we accomodated, within an Ethereum log, also the index on which the operation has been performed.
The index is specified by means of an identifier. Since in general there would be just as many Elasticsearch indices as the number of possible typologies of cultural goods (30), we have defined a lookup table. Keys are the available typologies, values are unsigned starting from 0. Hence, the identifier consists of an unsigned.
The reason why we used an unsigned, instead of a string for example, is due to the cost of generation of a log. A log is generated by the Ethereum Virtual Machine (EVM) every time an *event* is emitted during the contract's code execution.
The generation requires (among other things) also 8 gas for each byte of data. Data is intended as the arguments passed to the event. In our specific case, the arguments are the IPFS link and the identifier. Using an *uint8* type for the identifier, which was more than the required, resulted in saving gas.
We must point out that in order to actually filter logs based on the specific Elasticsearch index (indices) to restore, the (*uint8)* identifier has been defined in the event signature with the reserved keyword "*indexed*".
So to conclude, restoring an index is accomplished by the user specifying the index to restore, thus reading the IPFS links from the blockchain, hence retrieving the associated JSON objects from IPFS. At that time, encoded operations are re-executed *exactly* in the same order they have been previously performed. This is actually shown in Fig.2.

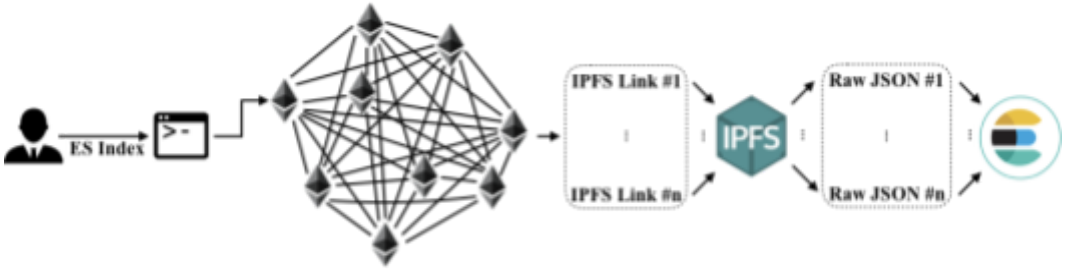

Fig. 2. Restoring an Elasticsearch index

.

It is important to outline that the "*off-chain operations storage and IPFS links accounting using Ethereum logs*" scheme just presented, without considering any of the specializations we made, is actually pretty general and therefore it can be used whatever the database considered.

V. AN EARLY PROTOTYPE

We implemented an early prototype which comes in the form of an Ethereum decentralized application (see Fig. 3).

Fig. 3. Proof-of-Concept prototype

The backend code are three smart contracts written in Solidity.
A smart contract named "*AccessRestriction"* allows only the authorized entity to record new items. The contract implements a "*restriction access*" pattern by means of an "*onlyOwner"* Solidity function modifier.
Another contract called "*Mortal"* (as it is usually referred in the jargon), which inherits from *AccessRestriction*, accounts for the specific need of removing the code and storage from the blockchain for whatever reason, at a point in time.
Finally, the "*CulturalGood"* smart contract. It encodes the business logic of the application by exposing a method which basically causes the EVM emitting the event, so as to let the correspondent log to be generated.
The contract inherits from *Mortal*, which in turn inherits from *AccessRestriction*, hence it provides both the two functionalities previously described.
The backend code has been deployed and run on a private and local Ethereum blockchain named "*ganache-cli*". The term private in this context only means isolated, rather than protected or secure. With respect to the client side, instead, we implemented a Web application running on Google Chrome. It is written in HTML5, CSS3 and JavaScript. The jQuery library has also been used.
The next component is "*Metamask"* which allows us to connect, from within the browser, to a custom RPC endpoint exposed by ganache-cli. In this way, it is possible to import ganache-cli defined accounts. This enables the user to sign transactions. The last two components are the IPFS service and the Elasticsearch service.
The prototype we just presented has been deployed within a *containerized* dev-environment with "*Docker"*. Specifically, a *multi-container* Docker application has been built.

VI. COST EVALUATION

To evaluate the cost of our solution we need to consider two costs: a) the cost for the deployment of the *CulturalGood* smart contract, and, b) the cost related to insert, update or delete operations.
With respect to the first case, the *contract creation transaction* accounts for 289154 gas. The latter is due to the cost of several EVM operations that the transaction affects. The smart-contract cost is so composed: 32000 gas for a Gcreate, 200 gas per byte of calldata (Gcodedeposit), 32000 gas for a Gtxcreate, 21000 gas for a Gtransaction, 68 gas per non-zero byte of code (Gtxdatanonzero) and 4 gas per zero byte of code (Gtxdatazero).
The cost for the contract deployment has been evaluated on 16 October 2018. Specifically,

considering a gas price equals to 10 Gwei (< 2 minutes confirmation time) and the exchange at that specific time (1 ETH = 183 EUR), the actual cost for the contract deployment was approximately: 289154 × 〖10〗^(-8) × 183 ~ EUR 0,53.

As to the operations cost, instead, when a new cultural property is inserted, or when an existing one is altered or deleted, the user gets prompted with a Metamask notification in order to sign the transaction. When the transaction gets signed, it is then broadcasted. Broadcasting the transaction causes the *CulturalGood* smart contract method to be invoked. The function call once compiled into transaction data (*calldata*) accounts for 68 bytes overall. The first 4 bytes represent the method's signature followed by parameters in chunks of 32 bytes. In our case, the parameters are such that the first 32 bytes encode the IPFS link whereas the subsequent encode the unsigned.

Whatever the operation performed, calldata is such that the method's signature is always the same, what can vary are only the parameters. In the *worst case*, in terms of gas consumption, we end up with only 31 bytes out of 68 being zero. These corresponds to the 31 most significant bytes of the unsigned.

So, we can compute the upper bound of the amount of gas to pay. It turned out that, in the *worst case*, the transaction accounts for 25642 gas. This cost value can be split as follows: 375 gas for the LOG operation (Glog), 8 gas per byte in the LOG operation's data (Glogdata), 375 gas for each topic of the LOG operation (Glogtopic), 21000 gas for a Gtransaction, 68 gas per non-zero byte of code (Gtxdatanonzero) and 4 gas per zero byte of code (Gtxdatazero).

Following the same reasoning as above, the cost for storing within a log an IPFS link and an (*uint8*) unsigned, in the *worst-case* amounted to: 25642 × 〖10〗^(-8) × 183 ~ EUR 0,047.

The actual overall cost will depend on the number of insert, update, delete operations that will be performed over time. In this regard, we try to give an *estimate* based on both the information available in ICCD public reports and the total number of cultural goods actually stored (~ 2,7 Million). According to the just mentioned, every eight years roughly 1,35 Million cultural goods get indexed.

If we suppose the same for the next eight years (2019-2026), i.e. *considering only insert operations* (plus considering the same exchange rate as before), the cost to pay will be: 0.047 × 1.35 × 〖10〗^6 = EUR 63.450.

VII. CONCLUSIONS & FUTURE WORKS

According to last available public ICCD survey [,] (2011) , the total expenditure for cataloguing from 2002 to 2009 amounted to EUR 12.495.757. The estimate of 63.450 EUR we gave in Section VII actually accounts only for 0.5% of that cost. In other words, even if the solution has been designed to be proposed to one specific entity, the *State Police*, it may be even worthwhile for the ICCD bearing this cost.

It should be noted that in our proof-of-concept prototype, as shown in Fig. 3, data gets supplied to one single IPFS node only. Even if data cannot get garbage collected, since automatically pinned after being added to IPFS, we still have data on one single IPFS node. In order to improve the data redundancy, one possible enhancement may be keeping several IPFS nodes online pinning the data. The more the IPFS nodes pinning content, the better the redundancy.

In this respect, there are some tools being developed which can help in sharing the costs of bandwidth and disk-storage. Among all possible implementations *IPFS Cluster* seems to be the most promising. One possible future work may involve designing a solution of this kind.

Finally, another possible future work may involve, instead, designing a scheme in which nodes in the Elasticsearch cluster are actually considered byzantine.

ACKNOWLEDGEMENT

This work has been partially supported by the Italian Ministry of Education and Research (MIUR) in the framework of the CrossLab project (Departments of Excellence), and the University of Pisa in the framework of PRA 2019.

REFERENCES

Bhowmik, D. and Feng, T. (2017). The multimedia blockchain: A distributed and tamper-proof media transaction framework. In 2017 22nd International Conference on Digital Signal Processing(DSP), pages 1–5.

Galiev, A., Prokopyev, N., Ishmukhametov, S., Stolov, E.,Latypov, R., and Vlasov, I. (2018). Archain: a novel blockchain based archival system. In 2018 Second World Conference on Smart Trends in Systems, Security and Sustainability (WorldS4), pages 84–89.

Garcia-Barriocanal, E., S´anchez-Alonso, S., and Sicilia, M.-A. (2017). Deploying metadata on blockchain technologies. In Research Conference on Metadata and Semantics Research, pages 38–49. Springer.

Lemieux, V. L. (2017). A typology of blockchain record-keeping solutions and some reflections on their implications for the future of archival preservation. In 2017 IEEE International Conference on Big Data (BigData), pages 2271–2278.

Liang, X., Shetty, S., Tosh, D., Kamhoua, C., Kwiat, K., and Njilla, L. (2017). Provchain: A blockchain-based data provenance architecture in cloud environment with enhanced privacy and availability. In 2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID), pages 468–477.

Vishwa, A. and Hussain, F. K. (2018). A blockchain based approach for multimedia privacy protection and provenance. In 2018 IEEE Symposium Series on Computational Intelligence (SSCI), pages 1941–1945.

[] Catalogo Generale dei Beni Culturali. Retrieved from: http://www.catalogo.beniculturali.it/sigecSSU_FE (Accessed: March 13, 2019).

[] Openiccd. Retrieved from: http://www.iccd.beniculturali.it/getFile.php?id=4082 (Accessed March 13, 2019).

[] Openiccd. Retrieved from: http://www.catalogo.beniculturali.it/opendata/ (Accessed March 13, 2019).

[] ICCD price survey of cataloguing in Italy. Retrieved from: http://iccdold.beniculturali.it/getFile.php?id=1669 (Accessed: March 13, 2019).

[] Elasticsearch. Retrieved from: https://www.elastic.co/guide/en/elasticsearch/reference/current/getting-started.html

[] G. Wood, “Ethereum: A secure decentralised generalised transaction ledger,” last revision 2019. [Online]. https://ethereum.github.io/yellowpaper/paper.pdf

[] E. Androulaki, A. Barger, V. Bortnikov, C. Cachin, K. Christidis, A. De Caro, D. Enyeart, C. Ferris, G. Laventman, Y. Manevich, et al. “Hyperledger fabric: A distributed operating system for permissioned blockchains.” arXiv preprint arXiv:1801.10228, 2018.

[] R.G. Brown, J. Carlyle, I. Grigg, and M. Hearn. “Corda: An introduction.” R3 CEV, August, 2016.

[] D. Schwartz, N. Youngs, A. Britto, “The Ripple Protocol Consensus Algorithm” Ripple Labs, Inc., San Francisco, CA, USA, Tech. Rep., 2014. [Online]. Available: https://ripple.com/files/ripple_consensus_whitepaper.pdf

[] J. P. Morgan Chase. “A Permissioned Implementation of Ethereum.”. Accessed Feb 20, 2018. [Online]. https://github.com/jpmorganchase/quorum

[] J. Benet, “IPFS - Content Addressed, Versioned, P2P File System (DRAFT 3)”. Accessed 2014. (no publishing or posting data available). https://arxiv.org/abs/1407.3561v1