

Article

Smart Contracts for Managing the Chain-of-Custody of Digital Evidence: A Practical Case of Study

Pablo Santamaría ¹, Llanos Tobarra ¹, Rafael Pastor-Vargas ¹ and Antonio Robles-Gómez ^{1*}

¹ Departamento de Sistemas de Comunicación y Control, Escuela Técnica Superior en Ingeniería Informática, Universidad Nacional de Educación a Distancia (UNED), Madrid 28040, Spain; psantamar4@alumno.uned.es (P. S.); llanos@scc.uned.es (L.T.); rpastor@scc.uned.es (R.P.-V.)

* Correspondence: arobles@scc.uned.es (A.R.-G.)

† The extended manuscript has been an Special Issue editors' invitation from the paper: "Santamaría P., Tobarra Ll., Pastor-Vargas R., Robles-Gómez A. Designing the CoC Process for Blockchain-based Digital Evidences. In Proceedings of the Blockchain and Applications, 4th International Congress, L'Aquila, Italy, 13–15 July 2022".

Abstract: The digital revolution is renewing many aspects of our lives, which is also a challenge in judicial processes, such as the Chain-of-Custody (CoC) process of any electronic evidence. A CoC management system must be designed to guarantee them to maintain its integrity in court. This issue is essential for digital evidence's admissibility and probative value. This work has built and validated a real prototype to manage the CoC process of any digital evidence. Our technological solution follows a process model that separates the evidence registry and any evidence itself for scalability purposes. It includes the development of an open-source smart contract under Quorum, a version of Ethereum oriented to private business environments. The significant findings of our analysis have been: (1) Blockchain networks can become a solution, where integrity, privacy and traceability must be guaranteed between untrustworthy parties; and (2) the necessity of promoting the standardization of CoC smart contracts with a secure, simple process logic. Consequently, these contracts should be deployed in consortium environments, where reliable, independent third parties validate the transactions without having to know their content.



Citation: Santamaría, P.; Tobarra, L.; Pastor-Vargas, R.; Robles-Gómez, A. Smart Contracts for Managing the Chain-of-Custody of Digital Evidence: A Practical Case of Study. *Smart Cities* **2023**, *1*, 1–19. <https://doi.org/>

Academic Editor: Javier Prieto and Roberto Casado Vara

Received: 15 January 2023

Revised: 15 February 2023

Accepted: 17 February 2023

Published:



Copyright: © 2023 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

Keywords: Blockchain; Smart Contracts; Chain-of-Custody (CoC); Zero-Knowledge Proof (ZKP); Alastria; Ethereum; Quorum

1. Introduction

The digital revolution and new digital economy are transforming the basic processes of our society at all levels. This way, creating new smart infrastructures becomes essential in our society. These infrastructures and related applications help cities improve governance and the quality of life of resident citizens, which is included in this transformation [1]. For instance, it is possible to create voting applications [2], auctions [3], multi-signature wallets [4], and so on. These applications are created for different parties which do not trust each other, and the agreed rules of the game must be assured and complied with.

One of the domains that is being digitally transformed is the traditional processes used in trials. One of the processes which clearly can be improved is the case of managing probative evidence at a court. Related to this process, the concept of digital evidence [5] has become very relevant, even more with the exponential growth of crimes in which technology is the protagonist. Therefore, it is now necessary to ensure that the new digital evidence, as a new way of digital interaction among persons and companies, has a similar nature to physical ones. The principal purpose is that they can be admitted with the same value by proving the reality of their occurrence in the digital field during a judicial process.

In addition to this, the forensic process [6] is in charge of collecting this evidence, analyzing it and guaranteeing its relevance, authenticity and credibility with the required

legal guarantees. The Chain-of-Custody (CoC) is one of the related procedures which must ensure evidence has not been altered, disclosed or accessed by unauthorized people, from its acquisition to its availability in a court. This CoC process was traditionally carried out with documentary support and physical custody. The current challenge is to adapt this process to the digital world, in which electronic evidence can be intangible and volatile.

This way, UNE 71506:2013 [7] defines CoC “as the controlled traceability procedure applied to the evidence, from acquisition to analysis and final presentation”. This procedure is focused on performing the documentation to the evidence in chronological events [8,9]. Therefore, the CoC is the most crucial process of evidence documentation. It must assure that, in event of a crime presented to court of law, the evidence is authentic. In other words, it is the same evidence seized for the crime scene. It should reflect all individuals involved in the process of acquisition, collection, analysis of evidence, time records, as well as contextual information (case identification, unit and laboratory participants).

On the other hand, the concept of Blockchain arises for a large amount of security applications to save and protect data, by maintaining its integrity, by means of a chain structure [10]. Data is encrypted during the complete process. The origin of Blockchain is the cryptocurrency [11], and it has evolved towards the concept of Distributed Transaction Ledger (DTL) [12], which is understood as a distributed and digital ledger. This evolution towards enterprise applications is now considered one of the transforming technologies in the near future. The use of Blockchain technology can help us to improve the CoC process and the evidence management life-cycle. Several benefits can be achieved by using a Blockchain network in terms of transparency, fault tolerance, decentralization, and independent verification. This way, each network participant maintains an up-to-date copy of the complete chain, known as a node, being a part of digital elements. The network is also autonomous and sustainable when a node fails since the rest of the nodes can follow with their specific tasks. Writing and reading in the chain are decentralized, so a participant cannot entirely control the chain of blocks by itself. Additionally, any participant can verify the integrity; no intermediary participants are necessary. Security control of data access and traceability is also possible with these kinds of networks.

Therefore, the principal objective of this work is implementing and testing a prototype to manage the CoC process of any digital evidence, which reflects the functionality of the developed smart contract with a distributed architecture. This primary goal leads to the following specific objectives:

1. Three technological architectures for the CoC process of digital evidences are designed: *distributed*, *centralized* and *multi-blockchain*, as well as a set of design principles are proposed to minimize the limitations found in the literature. A review of these gaps is also detailed in this work. Specifically, our proposal has as a starting point the work [13], expanding and adapting it according to our defined principles;
2. A real prototype to manage the CoC process of digital evidences was developed. This prototype is based on the designed distributed architecture by following the proposed model for the CoC process. The prototype also satisfies most of the requirements detailed in the design principles. Moreover, our solution is open-source and compatible with the current top languages used in the Blockchain landscape;
3. A smart contract associated with the prototype was implemented under Quorum, a version of Ethereum oriented to private business environments. The Zero-Knowledge Proof (ZKP) protocol [14] has been employed for this purpose. This protocol allows it to have consensus on transactions and blocks while maintaining data privacy. As a result, entities can see the Blockchain network as reliable since their data are also protected when transmitted through the network without third parties knowing their content.

This work is organized as follows. Section 2 describes the evolution of Blockchain technology, as well as several existing works within the topic of this paper. Section 3 proposes our defined process model for the CoC of digital evidences. Section 4 describes our technological solution, including the different proposed architectures for the CoC process: *distributed*, *central*,

and *multi-blockchain*. Section 5 presents our implemented and tested case of study with smart contracts over Quorum, and Section 6 discusses its limitations and future directions. Finally, Section 7 draws our conclusions and outlines our future work.

2. Literature Review

This section presents the fundamentals of the technologies used for the example case, Blockchain, and reviews the fundamentals of the chain of custody (CoC).

2.1. Blockchain Evolution

Figure 1 shows the Blockchain evolution from cryptocurrencies to applications for a variety of contexts [10], such as finance, supply chain, energy, Internet of Things (IoT), healthcare, pharmaceuticals, intellectual property (IP), digital identity, real estate, government, media, arts, insurance, and justice. Bitcoin was the first implementation of Blockchain in 2008 by an anonymous person under the Satoshi Nakamoto pseudonym [11]. Blockchain was designed as a decentralized payment system with an internal token called Bitcoin.

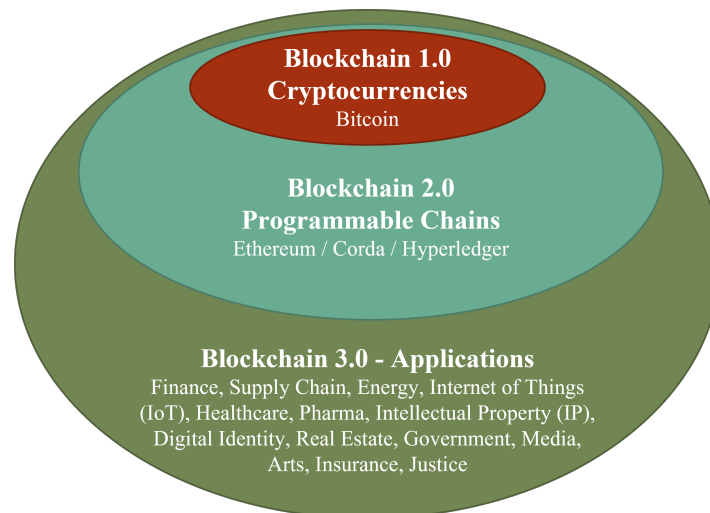


Figure 1. Blockchain Ecosystem [10].

The initial Blockchain network promoted by Bitcoin was public and global. It worked based on a Proof of Work (PoW) consensus [15], which, given its high trading value, allows large mining corporations to associate and invest in controlling the price and profit of the cryptocurrency. This network cannot be used for other applications since it does not have the capacity to be a programmable chain.

Vitalik Buterin conceived a platform for “Programmable Money” with smart contracts [10,16] to add the functionality of being a programmable chain. Furthermore, he proposed integrating a complete Turing language into the Bitcoin scripting system as an enhanced protocol. In addition, Gavin Wood is largely credited for making Ethereum a general-purpose computing platform. Currently, a smart contract is implemented with an Ethereum Virtual Machine (named EVM), which allows code to be executed as a complete Turing machine.

The transactions that change the state of the network consume GAS, a unit used to measure the exchange of an own currency of Ethereum, named Ether. This mechanism involves the control of how a contract consumes more resources than the allowed ones or else is saturated due to inadequate implementation. Ethereum was conceived as a speeder network than Bitcoin to launch these applications over Blockchain networks with many more users and a variety of services. However, the costs of deploying and invoking a smart contract can be high [17], which has a non-negligible FIAT value in the cryptocurrency market.

Despite this, smart contracts can reduce administration and save service costs, improve the efficiency of business processes, and minimize privacy risks, among others. Therefore, a great variety of opportunities and challenges appears with smart contracts [18] in the IoT field, distributed security/privacy systems, finance, data provenance, sharing economy, the public sector, etc. Some works are also studying ways of reducing transaction costs in smart contracts. For example, the authors of [19] identify the benefits of tradable permit schemes in smart contracts for increasing relevant and reducing irrelevant information, correcting asymmetries among actors, minimizing intermediaries, decreasing costs, improving trading quality, etc.

Other smart contract considerations include the design and testing of tasks for smart contracts as part of Blockchain's distributed applications. These considerations are also challenging from the point of view of software engineering. According to [20], there is no effective way to guarantee the code security of a smart contract. This way, debugging tools such as code auditing and formal verification techniques on software design patterns are highly desired to develop high-quality smart contracts [21]. Furthermore, programming languages, such as Solidity and VMs, are still limited and inefficient from the point of view of performance. Our technological prototype has been designed with principles which minimize these drawbacks—as detailed in Section 4—such as a secure, simple and standard control logic implementation; robust authentication and authorization management; periodic audit; separation of any evidence and the CoC registry; safe, reliable, durable and low-cost warehousing; among others.

It was also concluded in [20] that there is a lack of best practices, example codes, community support, and standardization. Alastria [22] was established as the Spanish Blockchain consortium in 2017 to minimize these drawbacks in Spain. It currently has more than 511 adhered members and supports Ethereum. Quorum is employed for our solution, a version of Ethereum oriented to business environments with privacy features based on the ZKP protocol that the parent version does not have. It is possible to limit who can participate in the transactions, who can write to the chain, which nodes can know the content of the transactions and those who cannot. For validation purposes, a seven-node network of the official Quorum distribution has been tested, and the smart contract has been deployed in the Alastria testnet.

2.2. The CoC Background

From the beginning of Blockchain development, the CoC problem has been observed as a classic example to be solved with its intrinsic characteristics [23,24]. The most specific studies on the use of Blockchain as a support for the digital custody process date from 2011 [9]. This research line has been growing with related concepts and approaches.

Bonomi et al. proposed the B-CoC [13] framework aimed at guaranteeing the audit integrity of digital evidence collection, and its traceability by third parties. However, multiple stakeholders have not been considered nor complemented by an adequate evidence management system. Our solution is based on their proposal, expanding and including additional features. For instance, an evidence invalidation process has been included in our work, several possible architectures and specific control functions to manage participants at the CoC level and with granular access to evidence, among other design principles.

The work of [25] presents a CoC architecture in the context of Blockchain-based forensics. However, it does not offer a complete vision of the transactions nor put special attention on the preservation of evidences or the integrity guarantee mechanisms avoiding any modification. This work presents an evolution of a previous work [26], where an architecture based on Ethereum Hyperledger oriented to the forensic CoC was presented without considering a permission management.

In [27], the creation of a Blockchain-based incident response system is combined for easing a subsequent forensic analysis. They propose managing the evidence in a distributed ledger and facilitating the chronological storage of the associated documentation. However,

it is not based on the fundamental aspect of guaranteeing all the necessary properties for a CoC process.

The LEChain [28] framework is one of the more complete proposals found in the literature. It presents a Blockchain-based evidence management model that includes mechanisms for the supervision of evidence collection and the protection of the identities of witnesses and jurors (and their specific votes). It even addresses the management of evidence in the cloud, although it does not offer full-support for transparency and audit mechanisms. Additionally, it does not cover the full-life cycle of digital evidence.

On the other hand, there are existing commercial applications, such as Khipus [29]. This app offers a personal evidence custody service over the Alastria network, for instance, with WhatsApp or email. According to their website, they are a service for the issuance of electronic stamps that are not qualified in time. The purpose of this solution differs from the principal objective of this work, since it intends to act as a custody and third party service, but it is an example of the possibility that it can offer society.

Finally, three basic architectural designs with different challenges are proposed in [30]. A discussion about the need to promote a nationwide smart contract standardization for the CoC process is also detailed. Its principal conclusion is that the smart contract must be open-source and employed in consortium environments, where reliable, independent third parties validate the transactions without knowing their specific content. This work is an extended version in which a concrete technological solution is fully detailed from three possible architectures. Once the use of the distributed one is justified, a case of study is presented, including the implementation and testing of a real prototype.

As for the CoC process model, previous studies determine that a Blockchain-based network can be a great approach to leveraging this process. Blockchain guarantees transparency, authenticity, security and traceability. However, it is essential to avoid the problem of excessive growth of the chain, known as “Blockchain bloat”. These previous works agree on the need to decouple the evidence registry from the custody of the evidence itself. The balance between privacy and traceability is also highlighted in them. The CoC process is required to be verifiable, auditable and transparent, but this could compromise the privacy of legal procedures. Therefore, mechanisms to ensure that only legitimate parties access specific data are needed. The design principles of our proposed process model consider these functional requirements, being incorporated into our technological solution provided in this work. They are detailed in the next section.

3. Proposed Process Model

Researchers found many problems when accessing digital information during the CoC process. Digital evidences are duplicated, altered or modified, not allowed to keep their integrity. Roles and permissions over a digital element can also be not the corresponding ones. Additionally, the evidence storage could not be secure or sensitive along the time. Even digital evidences can suffer several regulations from different countries.

For these reasons, a CoC process of digital evidences is more complex than traditional evidences. Once studied the current situation in the literature is, the following process is proposed, which defines and implements a generalization of the process that can be adapted to any part of the world.

3.1. The CoC Process

Each participant involved in the CoC process must be registered with a particular role in the system. The *creator* role is for the forensic staff that collects the evidence from the scene and is responsible for it. Their functions will be to create a record in the CoC, assign an owner, manage participants with access to the evidence and notify those exceptions that occur throughout the cycle-life of the evidence by including marking it as invalid in case it is authorized its destruction. Additionally, the *owner* of the evidence can be the person (or people) who guarantee the custody of the digital evidence and act as a security officer for the credential generation to the participants to allow them access to the corresponding evidence.

The rest of the participants are either *investigators/researchers* who need to examine the evidence, or other forensic analysts, court staff, or law enforcement personnel that intervene in the different parts of the judicial process and need to examine the evidence. Additionally, the figure of the *master* participant is necessary for the CoC process, who is responsible for the deployment of the smart contract and the maintenance of the participants in the complete process. This way, the owner authorizes those participants' access to one or another piece of evidence. This custodian figure is essential to defining the final process to be implemented.

Figure 2 shows the proposed CoC process model for the evidence management, which includes a set of phases:

- *Acquisition*. The chain of custody begins at the evidence acquisition phase. Once collected, the process of creating the evidence must be instantiated in the chain of custody;
- *Analysis*. It invokes the process of analyzing the evidence access by the corresponding participants;
- *Report*. It invokes the process of reporting the evidence access by the corresponding participants;
- *Destruction*. In case the destruction is requested, invoking the method destruction of evidence will be needed.

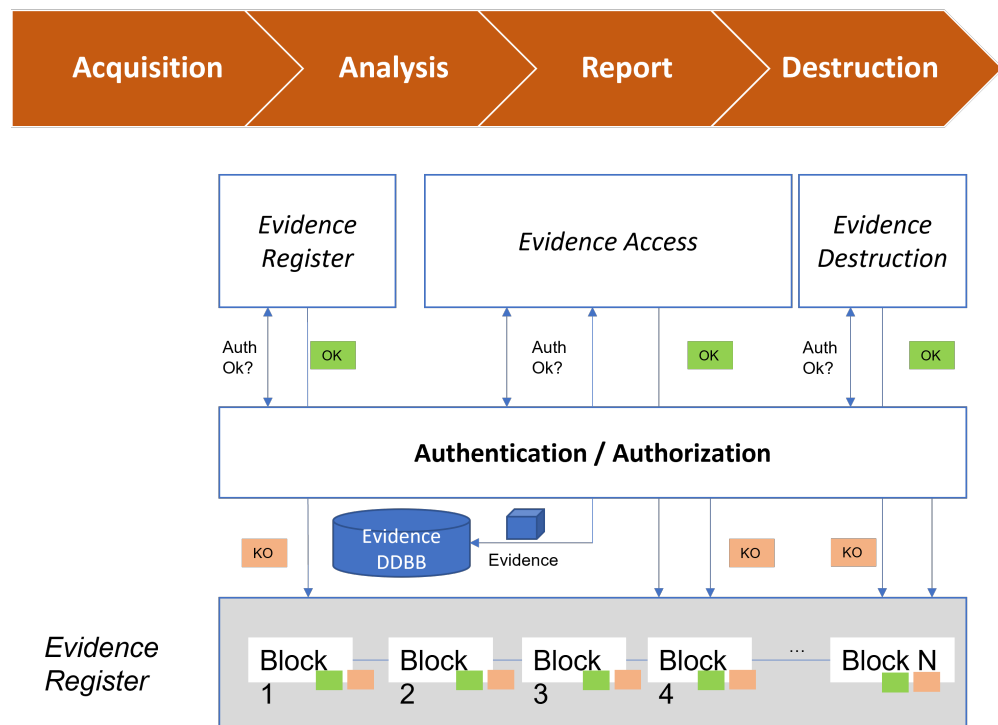


Figure 2. Proposed CoC Process Model.

3.2. Evidence Services

The CoC process requires defining the following services on the digital evidence to guarantee the traceability of actions that happens to the evidence:

- *Create evidence*. The person responsible for acquiring the evidence signs the record with all the circumstances related to the acquisition process. Then, these data are used to generate the corresponding record in the Blockchain network. Therefore, the person who is responsible for the act of acquisition is the one who must create the CoC evidence;
- *Invalidate evidence*. If necessary, who, when, how and why the destruction of the evidence was authorized must be recorded;
- *Obtain evidence*. The registration of all participants requesting access to the evidence;

- *Transfer evidence.* It is necessary to delegate the management of the evidence participants may transfer that responsibility to another competitor;
- *Show the register evidence.* Report on all entries in the log about a particular piece of evidence.

The creator of the evidence, once the digital evidence and the certificate of acquisition have been acquired, stores the evidence in the repository compatible with the architecture, if possible. In this process, the fingerprints are calculated as hash summaries that will validate that the evidence has not been modified since its acquisition. The evidence is securely stored in the secure evidence store, and it will generate a record of evidence with the creation.

Different participants in the investigation or judicial process will want to examine what has occurred during the life of the evidence. For this reason, who accessed the evidence needs to be registered, as well as when, where and how they accessed it, and what is the specific reason for consulting the evidence. The data that can be stored on access will depend on the availability of the information available on the network Blockchain (account identifier, IP address, etc.). It is not always possible to have connection source information.

When the competent authority determines the destruction of the evidence, all the circumstances must be recorded and invalidated in the evidence record. The services must be only consumed in a timed manner by the participants accredited for it. That is why it is necessary to check that participants have the necessary permits and log, including failed attempts.

The described requirements in this section have been incorporated into the technological solution provided in this work to manage the CoC process.

4. Technological Solution

Once our process model has been defined, we have to detail the technological solution employed for this work. This solution considers the lack of maturity and homogeneity of Blockchain technology, as well as the current evolution of many companies towards a cloud service model, so the specific needs of the CoC process are not affected by those technological changes. Specifically, when hardware and software requirements for our infrastructures depend on cloud services (or even a hybrid local-cloud infrastructure), it would be very fast to evolve or scale our architecture and applications, respectively. As a result, our infrastructures would not become obsolete since the services are paid per consumption. It would also be easy to be tolerant of failures, for instance. These aspects are more noticeable for disruptive technologies, such as Blockchain, in which fast technological evolution is possible. These changes are also subject to possible consortiums, companies, the scientific community, etc.

4.1. Design Principles

In this era of transformation, it is necessary to have a modular approach with clear and simple design principles that allow each module to evolve separately without changing the focus of the CoC process or affecting the recorded evidences. The design principles followed for our technological solution are the following:

- *A smart contract that implements the logic of a secure, simple and standard process of CoC.* An example of a smart contract can be found in [31]. It is imperative to reuse verified code, such as Open Zepellin [32] libraries. Programming language can be vulnerable [33,34], and they are specialized in generating safe libraries for Solidity contracts. A smart contract must be open-source and standardized where they gain the endorsement and scrutiny of the entire community.
- *The data related to the CoC are only accessed through the smart contract.* The architecture in these networks are layers and interactions with the contract, but it should never maintain or manage the chain information without invoking a smart contract function. The rest of the architecture must facilitate access but also guarantee the independence of CoC validation for third parties, if needed, with the appropriate permissions through the interfaces of the Blockchain network itself.

- *Deployment of the contract in one or more consortium of Blockchain.* Our developments must be technologically prepared for this deployment.
- *Permissive technologies with ZKP consensus support.* That is, the participation of public or private consortiums that guarantee independence from the third parties. On the other hand, a licensed technology is essential to allow, if necessary, privacy and confidentiality without losing the network's consensus on the validity of the transactions.
- *Custody of the evidence in a separated way from the evidence in the CoC record.* The size of the evidence makes it unfeasible to use the Blockchain itself as a secure custody repository. However, the fingerprint of the evidence is stored in the Blockchain registry. This way, any tampering with the evidence would be detected. As a consequence, the resulting architecture is more complex.
- *Robust identity management and authentication method.* The NIST Guide Digital Identity Guidelines 800-636 [35] is the main reference for this property. Having the maximum certainty of a virtual correspondence to the real identity of the participant is essential for reliability purposes. The distributed architecture achieves this with wallets that store credentials based on public key cryptography. Private key custody is essential to avoid losing virtual currencies, and mechanisms have been designed (software, hardware, storage) that allow users to be the only ones who physically have it. Like others based on multiple authentication factors, this mechanism is sufficient to authenticate. There is no concept of identity; the person who has the private key is the owner. That is, there is only one authentication identifier. Regarding the custody process, it is also necessary to validate that the digital identity corresponds to the real identity of the participant. This way, the appropriate processes for managing the identities associated with these accounts and private keys must be defined.
- *The use of safe, reliable, durable and low-cost warehousing.* Any evidence must be available during the open judicial period, lasting even years. Traditionally, the solution has been to make copies of the evidence on various magnetic disks, encrypt them with keys and physically store different copies (by the entity, a judicial authority, or a third party). However, there is no guarantee that these copies will be accessible in the future, in addition, to managing the custody of the encryption keys. The cloud is another option for having the capacity of security, reliability and redundancy without having to maintain the infrastructure for large volumes that are occasionally recovered. Again, however, this is not a suitable option for frequent access.
- *Electronic evidence format following an open standard.* Since there is no consensus, we will focus more on the process of generating the trace than on imposing a standard evidence format. Our proposal leaves the CoC format open. Moreover, it is very important to know how the fingerprint is generated, leaving open to any kind of electronic evidence with a standard fingerprint, such as SHA-256 or SHA-512. We also cover in this work SHA-1 and MD5 for backwards compatibility. It is mandatory to generate a minimum footprint based on SHA-256. On the other hand, due to changes in forensic techniques over time and the availability of forensic tools, the evidence acquisition record must be included in the record. This record includes all the details and the reproducible property of the evidence. It is good practice to include in the evidence file or an attachment the artifacts in source code or object mode of the tools used to generate it.
- *Periodic accreditation of external security audits.* Our technological solution is robust to be audited periodically by third parties with applicable normative and regulatory requirements.

4.2. Architectures

Three technological architectures have been designed for the CoC process of digital evidence. Self-manipulation is not possible since all the information and the CoC trace resides in the own Blockchain network. It is not manipulable unless the active help of most consortium nodes.

The three designed architectures have several common elements, which are:

- *Safe Storage.* This artifact is a safe storage of evidences, protected and only accessible to legitimate CoC participants. User tracking is also considered for monitoring. Activity logs must also be recorded in the evidence registry for monitoring.
- *Identity Node.* There is an identity node per participant in a Blockchain network. Each participant must have their own node, if desired, to privately deploy a smart contract, and the rest of the nodes validate the transactions, but without knowing specific details.
- *Smart Contract.* This contract implements the CoC management processes and traces all events related to the digital evidence.
- *Blockchain Consortium.* Networks where the smart contract is deployed to generate consensus and chain of blocks (such as Alastria, R3, etc.), to guarantee the integrity and immutability of each evidence during the whole CoC process.
- *Identity Manager.* It manages the authentication and authorization processes in order to guarantee that each participant has access and permission to carry out the actions which correspond to their accreditation. All accounts and accreditation of identity are recognized as a person.
- *Gateway Safe Storage.* It is an interface for the uploading, downloading and additional controls for the evidence files.
- *User Interface.* It is a centralized or distributed web/mobile application that interacts with participants, in order to access the CoC processes.
- *Contract Development and Debugging.* It is the development environment of the smart contract with the capacity of compilation, debugging and deployment in the considered Blockchain network.
- *Crypto Service (Key Management System, KMS).* This component establishes a secure communication channel among the CoC participants and Identity Nodes. In particular, it is a dedicated cryptographic system with certified hardware for key custody, encryption and secure signature. This way, the security risks of having these keys in configuration files or memory in the systems are avoided.

The final architectural choice depends on the specific requirements of a service provider to its clients (in terms of costs, performance, etc.). These are:

- *Distributed architecture.* This proposed Blockchain-based architecture inherits from public networks (see Figure 3). Clients access to the network through a wallet app that stores the public cryptographic key-pair. The network stores the public key of the account and verifies the transactions signed from that wallet. Connections to the network depend on the protocol used—XML-RPC in the case of Ethereum networks, as our case is. This client must interact with the browser in order to integrate network data into a distributed application. An example is the Metamask extension [36] and the WEB3J library [37] for Ethereum networks.
- *Centralized architecture.* This architecture is service-oriented (see Figure 4). Access to the Blockchain network is shielded with a layer of services or micro-services that access the Blockchain network by delegation. In this case, it is necessary to implement a robust authentication and identity management system for the Blockchain network. Once authenticated, the private key can be retrieved on the server side (from a KMS) to be able to sign the network transactions. However, any third party with a provisioned account could verify the CoC process, independently.
- *Multi-blockchain centralized architecture.* It is a variation of the centralized solution by abstracting the network technology used (see Figure 5). The advantage is the redundancy of the chain of custody in different networks. It is a solution, for example, to provide a CoC service to third parties. This redundancy in different networks provides a higher degree of reliability and validation.

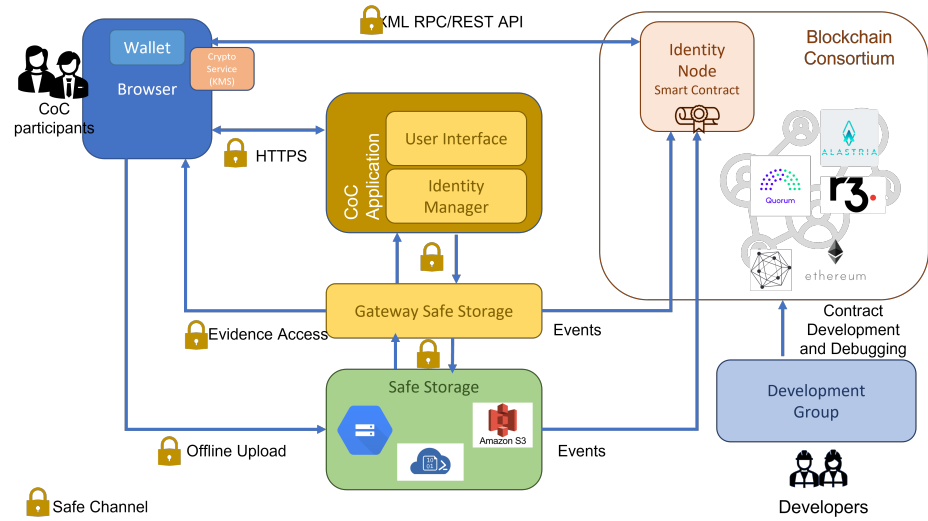


Figure 3. Distributed Architecture.

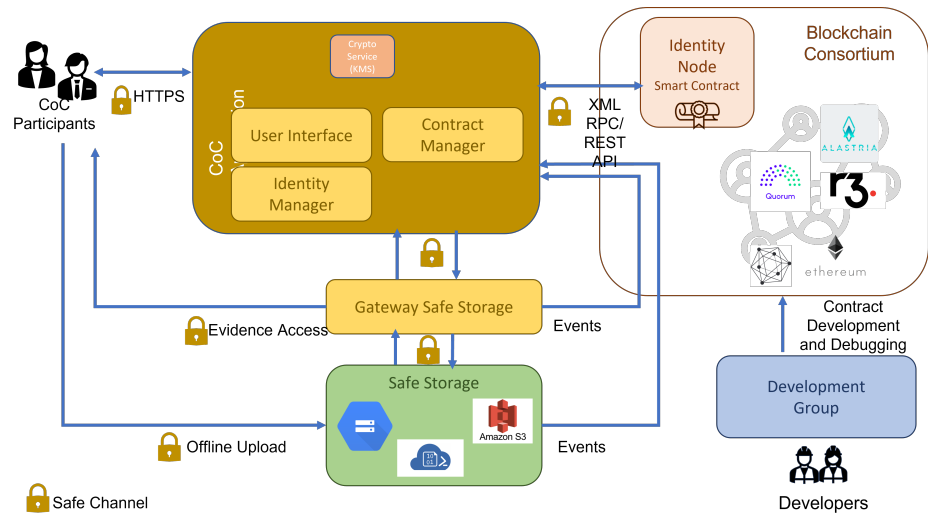


Figure 4. Centralized Architecture.

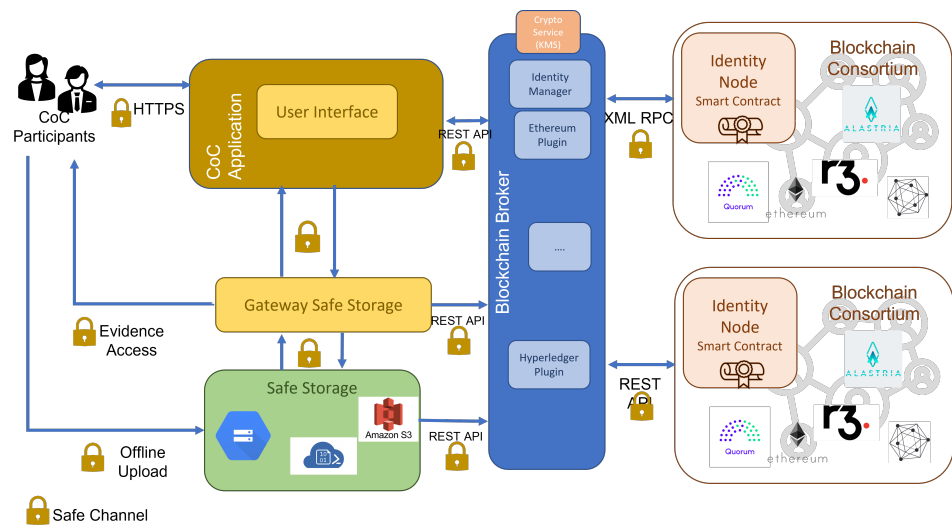


Figure 5. Multi-Blockchain Architecture.

4.3. Principal Differences

In addition to this, the specific components and actions involved in the distributed architecture, which are not presented in the other two ones, are:

- *Browser.* Participants connect through a specific browser to the CoC application and, as a consequence, to her/his specific Identity Node. This connection is performed via HTTPS, depending on the device (mobile or a workstation). The options available must be adapted to the channel through which is requested. The application sends the browser the necessary JavaScript to interact with the Blockchain network client.
- *Wallet.* It is needed for the CoC communication process among the CoC participants and Identity Nodes. Furthermore, for the application to access the wallet, a browser extension and a JavaScript library that interacts with it are necessary. For Ethereum networks, it is common to use a combination of WEB3J and Metamask.
- *Communication with the Blockchain network.* The entity node, depending on the underlying technology, will have an interface for interaction with the outside. In the case of Ethereum networks and the particular case of Quorum, it is XML-RPC [38], and with Hyperledger, it is an API based on REST (REST API) [39].
- *Communication of Safe Storage events.* The Secure Store must have the ability to generate events against the node to report access to the stored evidence. A proposal for how to do this is by implementing a FaaS (Function as a Service) in services such as AWS Lambda, which reads the log of events generated by the storage system and generates an audit log directly in the smart contract.
- *Interaction within the Gateway Safe Storage.* The application must manage the uploading or download of evidence from or to the client's device. It will mediate communication with the different secure stores, so having credentials in the secure store will not be necessary. So, access will be made with service accounts with granular permissions to the evidence managed from the identity management module. In the case of downloading, a secure URI (Uniform Resource Identifier) must be generated for downloading from the client.
- *Offline Upload of evidences.* Formalize a protocol for sending the evidence to the offline provider for uploading or massive data migration to avoid problems and cost overruns.
- *Development and Debugging environment.* Depending on the network, it is necessary to have tools and environments to build and debug the smart contract. In the case of Ethereum networks, the most suitable options are Truffle suite [40] and Remix [41]. Hyperledger has its own SDKs and projects for this purpose, as was Hyperledger-Composer at the time, but it is now discontinued.

The main differences of the centralized architectures with the distributed architecture are next:

- In this case, the application's access does not need extra extensions, and the complete identification, authentication and authorization processes are being managed in a centralized way.
- The application interacts with the network using the network account of the authenticated user. For this, it is necessary to implement the identity management module and a cryptography server to safely store the keys.
- Safe Storage must generate the event against the application to generate the audit record directly against the smart contract.

Lastly, the multi-Blockchain architecture is the most oriented approach to being a CoC service provider. It applies to institutions that desire redundancy at the Blockchain and the associated smart contracts to avoid problems related to having only one service provider. The principal difference with the distributed architecture is the need to separate the connection services layer from the application with the different smart contracts and identity management.

The last difference would be, in the case of wanting to provide services to several entities, the need for the application and the entire infrastructure to be multi-tenant, in order

to be able to instantiate different entities, whose information and access are kept isolated from each other on the same infrastructure.

5. Case of Study: Smart Contracts and Ethereum for the CoC Process

5.1. The Developed Prototype

This section describes the Proof of Concept (PoC), built for didactic and feasibility purposes to implement a complete CoC process with smart contracts. The prototype is open, secure, and robust. After reviewing the literature review exhaustively, several kinds of technological architectures were proposed: *distributed*, *centralized*, and *multi-blockchain*. For the purposes of this PoC, the distributed approach has been chosen by satisfying the proposed CoC model and design principles. As justified in our review, our architectural solution is based on the proposal by Bonomi et al. [13], adding several additional requirements for our practical case. The concrete architecture of the proposed solution can be observed Figure 6. It has been built under Quorum since it was one of the technologies chosen by Alastria.

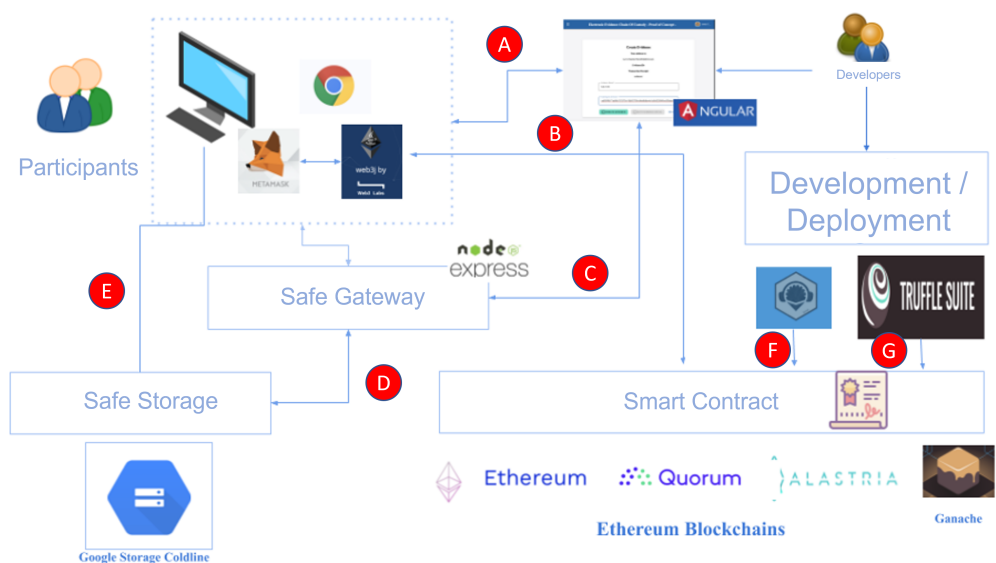
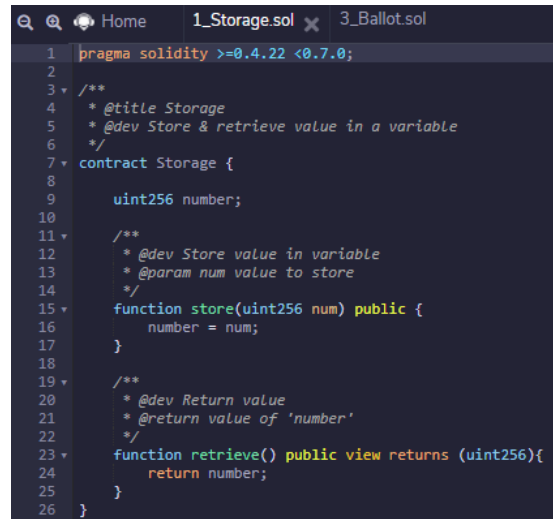


Figure 6. Architecture of the Prototype Developed.

Our prototype implements the CoC process model detailed in a previous section. From the point of view of smart contracts, we have also followed the studied features found in the literature and the gaps detected in them. Our solution includes the creation of any evidence, management of participants, audit processes, and invalidation of any evidence, among others. In addition, a *Safe Storage* service offered by Google—named *Google Storage Coldline*—is employed for the secure storage of evidences. The interaction with the secure storage is conducted through a gateway component, *Safe Gateway*, which implements a primary REST interface by using *Node Express* and the Google Storage API SDK offered by Google [42]. As observed in Figure 6, our solution interacts with Ethereum networks. In particular, the Ganache implementation and a 7-node test network of the official Quorum distribution have been used for this practical case. It has also been successfully deployed in the Alastria testnet. Quorum incorporates the ZKP protocol for the authorization of transactions and privacy purposes.

On the other hand, the implemented user-interface is a web application developed with *Angular* [43] and the *Web3j* library. The connection to the Blockchain is completely distributed through an Ethereum client. In our case, Metamask has been chosen with an Ethereum-compatible wallet. The smart contract itself has been implemented in Solidity using the development and debugging environment offered by Remix [41]. It generates compiled code to *Bytecode* to be executed on EVMs. An example is shown in Figure 7. TruffleSuite has been used for the deployment of the contract within the Blockchain network.



```

1  pragma solidity >=0.4.22 <0.7.0;
2
3  /**
4   * @title Storage
5   * @dev Store & retrieve value in a variable
6   */
7  contract Storage {
8
9      uint256 number;
10
11     /**
12      * @dev Store value in variable
13      * @param num value to store
14      */
15     function store(uint256 num) public {
16         number = num;
17     }
18
19     /**
20      * @dev Return value
21      * @return value of 'number'
22      */
23     function retrieve() public view returns (uint256){
24         return number;
25     }
26 }

```

Figure 7. Programming Environment.

5.2. Component Interactions

The workflow of the CoC process for the components belonging to the developed prototype must be established. The concrete component interactions are also detailed in Figure 6 with red circles and letters. Each interaction is described next:

- *Interaction A (Client application <-> Distributed application)*. The client application, through a Chrome browser, or another compatible browser with the extension provided by Metamask, accesses the distributed application by an HTTP communication. The browser downloads JavaScript code from the Node server, including the interface and style sheets. The web server does not interact with the smart contract or the Blockchain network.
- *Interaction B (Client application <-> Smart Contract)*. The application client interacts directly with the smart contract deployed in the network node. The client browser can do this through the Web3j JavaScript library and the Metamask extension.
- *Interaction C (Distributed application <-> Safe Gateway)*. The safe storage is organized as a server that provides different REST services. In our case, it provides a service for uploading/downloading information. The specific interaction between the application and the gateway is to adjust the upload/download parameters so that the browser can manage this information directly from the safe gateway.
- *Interaction D (Safe Gateway <-> Safe Storage)*. The safe gateway is in charge of collecting/sending the information from/to the safe storage. Through this option, the file requested to be downloaded or sent by the client is collected by the gateway. With the corresponding credentials, this is in charge of remotely storing files. This fact allows a control and abstraction layer of the safe storage without the client knowing the store's details or credentials. As further work, the gateway could perform additional controls on the information managed. For example, recalculating fingerprints of evidence, encrypting data, etc.
- *Interaction E (Client application <-> Safe Storage)*. The volume of electronic evidence could be Gigabytes, Terabytes or Petabytes. In these cases, it is reasonable for the client to interact directly with the store with his/her specific native credentials and tools. For example, Google's Gsutil tool or the offline data migration available from most cold cloud storage providers, such as Amazon AWS [44].
- *Interaction F (Development environment (Remix) <-> Smart Contract)*. The Remix environment [41] is ideal for the development and debugging tasks of the smart contract. An Ethereum virtual machine can be used, or well, accessed with a client via Web3J Metamask to interconnect with Ethereum. It has a multitude of options to be able to develop, compile, debug and test the smart contract.

- *Interaction G (Compilation and deployment <-> Smart Contract)*. For the compilation and deployment of the contract, we use the Truffle suite [41] that allows the deployment of Ethereum-based Blockchain networks. The application deployment is also used to compile and migrate the contract to the corresponding network.

5.3. The CoC Smart Contract

The developed smart contract is a program that implements the business logic of the CoC process to manage digital evidences, with a set of specific principal capabilities. The list of the principal implemented functions are:

- *CreateEvidence*. It creates a new evidence, in case it does not exist in the Blockchain network. As an example, the source code for this function is included in Figure 8. This function creates new evidences with a specific filename and an SHA-256 hash. Furthermore, some additional parameters are defined, such as its identifier, the creator and owner, etc. It is essential to be a participant of the network to perform this action.

```
function CreateEvidence ( string memory _evidence_filename,
                        string memory _evidence_hashdigest_SHA256
)
public
ParticipantExists(msg.sender, true)
returns(bytes32)
{
    bytes32 _EvidenceID = GenerateEvidenceID(_evidence_filename,
        _evidence_hashdigest_SHA256);

    bool exists = evidences[_EvidenceID].EvidenceID != 0$ \times $0;

    if (!exists && _EvidenceID != 0$ \times $0) {

        evidences[_EvidenceID].EvidenceID = _EvidenceID;
        evidences[_EvidenceID].owner = msg.sender;
        evidences[_EvidenceID].creator = msg.sender;
        evidences[_EvidenceID].isValid = true;
        evidences[_EvidenceID].evidence_filename = _evidence_filename;
        evidences[_EvidenceID].evidence_hashdigest_SHA256 =
            _evidence_hashdigest_SHA256;
        evidences[_EvidenceID].hasAccess[msg.sender] = true;

        string memory audit_description="Evidence creation";
        EmitAuditRecord(_EvidenceID,audit_description);
        return(_EvidenceID);
    } else {

        string memory audit_description="Evidence exists!";
        EmitAuditRecord(_EvidenceID,audit_description);
        return(_EvidenceID);
    }
}
```

Figure 8. Source Code to Create an Evidence in the Smart Contract.

- *GetParticipantList*. It returns the list of all participants who are in the Blockchain network.
- *AddMasterParticipant*. It adds the master participant data. This action is only possible if the participant did not exist in the Blockchain network previously.
- *AddParticipant*. It adds the data of any participant. This action is only possible if the participant did not exist in the Blockchain network previously.

- *GetParticipant*. It obtains the participant data required by another participant located in the Blockchain network.
- *EmitAuditRecord*. It adds to the evidence the record of what has happened, and it sends an event to the Blockchain network.
- *GenerateEvidenceID*. It is a public evidence ID generator. It can be used for any participant belonging to the Blockchain network.
- *SetParticipantToEvidence*. It adds a participant to the evidence. Only the evidence owner can perform this action.
- *RemoveParticipantFromEvidence*. It adds a participant to the evidence. Only the evidence owner can perform this action.
- *TransferOwnership*. An owner can transfer the owner role of evidence to another participant belonging to the Blockchain network.
- *InvalidateEvidence*. The owner can invalidate evidence in the Blockchain network.
- *NotifyException*. The owner can notify an exception about evidence in the Blockchain network.
- *SetEvidenceHashdigestMD5*. An evidence creator can add an MD5 hash to this evidence.
- *SetEvidenceHashdigestSHA1*. An evidence creator can add a SHA1 hash to this evidence.
- *SetEvidenceFormatVersion*. An evidence creator can add the format version of the object that contains this evidence.
- *SetEvidenceFormatType*. An evidence creator can add the format type of the object that contains this evidence.
- *SetEvidenceObjectType*. An evidence creator can add the object type that contains this evidence.
- *SetEvidenceUriToSecureVault*. An evidence creator can add its URI to the safe storage.
- *SetEvidenceFilepath*. An evidence creator can add its file path.
- *SetEvidenceDescription*. An evidence creator can add its description.
- *SetAcquisitionStatementUriToSecureVault*. An evidence creator can add the URI path of its acquisition document into safe storage.
- *SetAcquisitionStatementHashdigestSHA256*. An evidence creator can add the SHA256 hash of its acquisition document into safe storage.
- *SetAcquisitionStatementFilename*. An evidence creator can add the filename of its acquisition document into safe storage.
- *SetEvidenceDetailAll*. An evidence creator can update all its data from a smart contract.
- *GetEvidenceAudit*. Any authorized participant in the Blockchain network can access evidence data for audit purposes.
- *GetEvidence*. Any authorized participant in the Blockchain network can access the evidence itself.
- *GetEvidenceDescription*. Any authorized participant in the Blockchain network can access an evidence description.
- *GetEvidenceUriToSecureVault*. Any authorized participant in the Blockchain network can access the URI location of evidence.
- *GetEvidenceObjectType*. Any authorized participant in the Blockchain network can access the object type containing evidence.
- *GetEvidenceFormatType*. Any authorized participant in the Blockchain network can access the format type of evidence.
- *GetEvidenceHashdigestSHA1*. Any authorized participant in the Blockchain network can access the SHA1 hash of evidence.
- *GetEvidenceHashdigestSHA256*. Any authorized participant in the Blockchain network can access the SHA256 hash of evidence.
- *GetEvidenceHashdigestMD5*. Any authorized participant in the Blockchain network can access the MD5 hash of evidence.
- *GetEvidenceAcquisitionStatementFilename*. Any authorized participant in the Blockchain network can access the filename of the acquisition document of evidence.
- *GetEvidenceAcquisitionStatementUriToSecureVault*. Any authorized participant in the Blockchain network can access the URI path of the acquisition document of evidence from the safe storage.

- *GetEvidenceAdquisitionStatementHashdigestSHA256*. Any authorized participant in the Blockchain network can access the SHA256 hash of the acquisition document of evidence from the safe storage.

6. Discussion and Limitations

The digital revolution is renewing many aspects of our lives. This new reality is also being reflected in current judicial processes for the CoC evidence process in a similar way to traditional evidences. For this reason, it is essential to have a CoC management system that ensures the integrity and privacy of any digital evidence in a court. This is a necessary condition, though not sufficient, to decide on the admissibility and probative value of digital evidence. Knowing, without a doubt, who, when, how, and why somebody could access the digital evidence. Blockchain can become a suitable solution to improve this CoC process, where integrity and privacy must be guaranteed between untrustworthy parties. Several benefits can be achieved in terms of transparency, fault tolerance, decentralization, and independent verification, as in our case is.

From the business perspective, the costs of a CoC process can become very relevant. Remind, our technological solution consumes the GAS unit to measure the exchange of Ether tokens during transactions. However, smart contracts can reduce administrative procedures and save costs. In our case, the access functions do not have costs since the evidence registry, and any evidence itself are separated in different secure storage, as detailed in previous sections. Audit and event generation would generate GAS costs in the implemented smart contract for the PoC proposed in this work.

Our developed prototype has been built for didactic and feasibility purposes to check the end-to-end implementation of the CoC process with smart contracts in an open, secure, and robust way. It implements both the design principles and the features of the distributed architecture detailed in Section 4, including the creation/access/destruction of the evidence, the management of participants, the audit procedure, etc. The distributed application has been developed is suitable for mobile devices and native applications. The technological decisions of this work have been subrogated to the strategic decisions of Alastria.

The smart contract of our prototype is prepared for both single and multiple entities. Each of them could share the smart contract or have a dedicated one. The principal aim of our solution is to optimize its size and to keep the evidence integrity and privacy during all steps in the CoC process. Performance time is not a very relevant indicator since it is not a real-time process to access or store any evidence, among other available functions. The smart contract manages all events by including associated data, authentication/authorization permissions, evidence registry, and access to any evidence itself. Additionally, the implemented prototype includes the audit and tracking procedures.

There is an opportunity to standardize smart contracts for this CoC process with the support of both the scientific community and the competent authorities. This standardization will provide society with the characteristics of reliability and security. Standardization can be focused on the most used programming languages for current smart contracts, such as Solidiy or Chaincode. The multi-blockchain approach could also be considered at the architectural level since Blockchain technology is not yet mature at a business level.

Another challenge is the need to deploy these contracts in a consortium since the CoC process also demands a holistic approach at the process and participant levels. Blockchain is a technology that fits perfectly for the required technical purposes if a robust process of evidence management and participants accompanies it. Otherwise, the scenario would not be different from using other database technology. Blockchain autonomously does not eliminate the risk of intentional manipulation of data. Our presented prototype also tries to reflect these aspects as a proof of concept.

At a technological level, Alastria has also incorporated in its roadmap the incorporation of Hyperledger. It was decided to add Hyperledger Besu, an Ethereum client project to its capabilities, as a possible alternative. Additional advances of our proposed prototype can

also be included according to new Alastria requirements since our technological architecture is modular and flexible.

7. Conclusions and Future Works

A real fully-functional prototype to manage the CoC process of any digital evidence has been built and validated in this work. Three designed architectures (distributed, centralized and multi-blockchain) and several principles to be incorporated into our prototype are first specified. Its associated smart contract has been developed under Quorum, a version of Ethereum oriented to private business environments, which is open-source and compatible with the current top languages, to evolve to standardize smart contracts for the CoC process. Promoting the standardization of smart contracts in a secure and simple way is essential. Independent third parties can validate the Blockchain transactions without having to know their content thanks to ZKP protocols and the designed principles of our prototype.

As a next step, our prototype could be improved with the incorporation of a multi-blockchain approach, as well as include support for Hyperledger Besu. Other further tasks may be to improve the user interface of our solution. The interoperability of the smart contract to interact directly with third parties, with the use of "Oracles" gateways, can be another option for improvement.

Author Contributions: All authors were involved in the foundation items. All authors wrote the paper. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Not applicable.

Acknowledgments: Authors would like to acknowledge the support of the I4Labs UNED research group, the CiberGID UNED innovation group with the CiberScratch 2.0 project, the SUMA-CITeL research project for the 2022-2023 period (096-043077), as well as the E-Madrid-CM Network of Excellence (S2018/TCS-4307). The authors also acknowledge the support of SNOLA, officially recognized Thematic Network of Excellence (RED2018-102725-T) by the Spanish Ministry of Science, Innovation and Universities.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CoC	Chain of Custody
DTL	Distributed Transaction Ledger
EVM	Ethereum Virtual Machine
FaaS	Function as a Service
HTTP	Hypertext Transfer Protocol
ID	Identification
IoT	Internet of Things
IP	Intellectual Property
KMS	Key Management System
MD5	Message Digest Algorithm 5
PoC	Proof of Concept
PoW	Proof of Work
SDK	Software Development Kit
SHA	Secure Hash Algorithm
URI	Uniform Resource Identifier
XML-RPC	eXtensible Markup Language-Remote Procedure Call
ZKP	Zero-Knowledge Proof

References

1. Clim, A.; Toma, A.; Zota, R.D.; Constantinescu, R. The Need for Cybersecurity in Industrial Revolution and Smart Cities. *Sensors* **2023**, *23*, 120. <https://doi.org/10.3390/s23010120>.
2. Benabdallah, A.; Audras, A.; Coudert, L.; El Madhoun, N.; Badra, M. Analysis of Blockchain Solutions for E-Voting: A Systematic Literature Review. *IEEE Access* **2022**, *10*, 70746–70759. <https://doi.org/10.1109/ACCESS.2022.3187688>.
3. Wu, S.; Chen, Y.; Wang, Q.; Li, M.; Wang, C.; Luo, X. CReam: A Smart Contract Enabled Collusion-Resistant e-Auction. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 1687–1701. <https://doi.org/10.1109/TIFS.2018.2883275>.
4. Selvi, S.S.D.; Paul, A.; Rangan, C.P.; Dirisala, S.; Basu, S. Splitting and Aggregating Signatures in Cryptocurrency Protocols. In Proceedings of the 2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON), Newark, CA, USA, 4–9 April 2019; pp. 100–108. <https://doi.org/10.1109/DAPPCON.2019.00021>.
5. Stringer-Calvert, D.W. Digital Evidence. *Commun. ACM* **2002**, *45*, 128. <https://doi.org/10.1145/505248.505280>.
6. Ajjola, A.; Zavarovsky, P.; Ruhl, R. A review and comparative evaluation of forensics guidelines of NIST SP 800-101 Rev.1:2014 and ISO/IEC 27037:2012. In Proceedings of the World Congress on Internet Security (WorldCIS-2014), London, UK, 8–10 December 2014; pp. 66–73. <https://doi.org/10.1109/WorldCIS.2014.7028169>.
7. Asociación Española de Normalización. Information Technologies (IT). Methodology for the Digital Evidences Forensic Analysis. Standard, UNE. Normalización Española. 2013. Available online: <https://www.une.org/encuentra-tu-norma/busca-tu-norma/norma?c=N0051414> (accessed on 10 February 2023).
8. Adam, I.; Varol, C. Intelligence in Digital Forensics Process. In Proceedings of the 2020 8th International Symposium on Digital Forensics and Security (ISDFS), Beirut, Lebanon, 1–2 June 2020; pp. 1–6. <https://doi.org/10.1109/ISDFS49300.2020.9116442>.
9. Giova, G. Improving Chain of Custody in Forensic Investigation of Electronic Digital Systems. *Int. J. Comput. Sci. Netw. Secur.* **2011**, *1*, 1–9.
10. Swan, M. *Blockchain: Blueprint for a New Economy*; O'Reilly Media: Sebastopol, CA, USA, 2015.
11. Nakamoto, S. *Bitcoin: A Peer-to-Peer Electronic Cash System*; Washington, DC, USA, 2008.
12. Clint, R.V. DLT/Blockchain as a Building Block for Enterprise Transformation. *IEEE Eng. Manag. Rev.* **2019**, *47*, 24–27. <https://doi.org/10.1109/EMR.2019.2895303>.
13. Bonomi, S.; Casini, M.; Ciccotelli, C. B-CoC: A Blockchain-based Chain of Custody for Evidences Management in Digital Forensics. *arXiv* **2018**, *arXiv:1807.10359*.
14. Tyagi, S.; Kathuria, M. Role of Zero-Knowledge Proof in Blockchain Security. In Proceedings of the 2022 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COM-IT-CON), Faridabad, India, 26–27 May 2022; Volume 1, pp. 738–743. <https://doi.org/10.1109/COM-IT-CON54601.2022.9850714>.
15. Ouyang, Z.; Shao, J.; Zeng, Y. PoW and PoS and Related Applications. In Proceedings of the 2021 International Conference on Electronic Information Engineering and Computer Science (EIECS), Changchun, China, 23–26 September 2021; pp. 59–62. <https://doi.org/10.1109/EIECS53707.2021.9588080>.
16. Kushwaha, S.S.; Joshi, S.; Singh, D.; Kaur, M.; Lee, H.N. Ethereum Smart Contract Analysis Tools: A Systematic Review. *IEEE Access* **2022**, *10*, 57037–57062. <https://doi.org/10.1109/ACCESS.2022.3169902>.
17. Baird, K.; Jeong, S.; Kim, Y.; Burgstaller, B.; Scholz, B. The Economics of Smart Contracts. *arXiv* **2019**, *arXiv:1910.11143*.
18. Zheng, Z.; Xie, S.; Dai, H.N.; Chen, W.; Chen, X.; Weng, J.; Imran, M. An overview on smart contracts: Challenges, advances and platforms. *Future Gener. Comput. Syst.* **2020**, *105*, 475–491. <https://doi.org/10.1016/j.future.2019.12.019>.
19. Shahab, S.; Allam, Z. Reducing transaction costs of tradable permit schemes using Blockchain smart contracts. *Growth Chang.* **2020**, *51*, 302–308. <https://doi.org/10.1111/grow.12342>.
20. Zou, W.; Lo, D.; Kochhar, P.S.; Le, X.B.D.; Xia, X.; Feng, Y.; Chen, Z.; Xu, B. Smart Contract Development: Challenges and Opportunities. *IEEE Trans. Softw. Eng.* **2021**, *47*, 2084–2106. <https://doi.org/10.1109/TSE.2019.2942301>.
21. Górski, T. The $k + 1$ Symmetric Test Pattern for Smart Contracts. *Symmetry* **2022**, *14*, 1686. <https://doi.org/10.3390/sym14081686>.
22. Alastria. 2019. Available online: <https://www.alastria.io/> (accessed on 10 February 2023).
23. Li, X.; Jiang, P.; Chen, T.; Luo, X.; Wen, Q. A survey on the security of blockchain systems. *Future Gener. Comput. Syst.* **2020**, *107*, 841–853. <https://doi.org/10.1016/j.future.2017.08.020>.
24. Lusetti, M.; Salsi, L.; Dallatana, A. A blockchain based solution for the custody of digital files in forensic medicine. *Forensic Sci. Int. Digit. Investig.* **2020**, *35*, 301017. <https://doi.org/10.1016/j.fsidi.2020.301017>.
25. Lone, A.H.; Mir, R.N. Forensic-chain: Blockchain based digital forensics chain of custody with PoC in Hyperledger Composer. *Digit. Investig.* **2019**, *28*, 44–55. <https://doi.org/10.1016/j.diin.2019.01.002>.
26. Lone, A. Forensic-chain: Ethereum blockchain based digital forensics chain of custody. *Sci. Pract. Cyber Secur. J.* **2017**, *1*, 21–27.
27. Al-Khateeb, H.; Epiphaniou, G.; Daly, H. Blockchain for Modern Digital Forensics: The Chain-of-Custody as a Distributed Ledger. In Blockchain and Clinical Trial: Securing Patient Data; Springer International Publishing: Cham, Switzerland, 2019; pp. 149–168. https://doi.org/10.1007/978-3-030-11289-9_7.
28. Li, M.; Lal, C.; Conti, M.; Hu, D. LEChain: A blockchain-based lawful evidence management scheme for digital forensics. *Future Gener. Comput. Syst.* **2021**, *115*, 406–420. <https://doi.org/https://doi.org/10.1016/j.future.2020.09.038>.
29. ioBUILDERS Blockchain Tech & Ventures. Khipus, deja huella con tu móvil. 2018. Available online: <https://khipus.io/> (accessed on 10 February 2023).

30. Santamaría, P.; Tobarra, L.; Pastor-Vargas, R.; Robles-Gómez, A. *Designing the Chain of Custody Process for Blockchain-Based Digital Evidences*. In *Blockchain and Applications*, 4th International Congress; Prieto, J., Benítez Martínez, F.L., Ferretti, S., Arroyo Guardado, D., Tomás Nevado-Batalla, P., Eds.; Springer International Publishing: Cham, Switzerland, 2023; pp. 225–236.
31. Anne, V.P.K.; Ayyadevara, R.C.; Potta, D.; Ankem, N. *Storing and Securing the Digital Evidence in the Process of Digital Forensics through Blockchain Technology*. In *Proceedings of the International Conference on Data Science, Machine Learning and Artificial Intelligence, DSMLAI'21*, Windhoek, Namibia, 9–12 August 2021; Association for Computing Machinery: New York, NY, USA, 2021; pp. 272–276. <https://doi.org/10.1145/3484824.3484899>.
32. OpenZeppelin Contracts. Available online: <https://github.com/OpenZeppelin/openzeppelin-contracts> (accessed on 10 February 2023).
33. Arroyo, D.; Rezola, A.; Hernández, L. *Principales problemas de seguridad en los smart contracts de Ethereum*. XII Jornadas STIC CCN-CERT, Madrid, Spain, 12–13 December 2018.
34. Manning, A. *Solidity Security: Comprehensive List of Known Attack Vectors and Common Anti-Patterns*. 2018. Available online: <https://blog.sigmaprime.io/solidity-security.html> (accessed on 10 February 2023).
35. Grassi, P.A.; Garcia, M.E.; Fenton, J.L. *NIST Special Publication 800-63. Digital Identity Guidelines*. 2017. Available online: <https://pages.nist.gov/800-63-3/> (accessed on 10 February 2023).
36. Metamask. Available online: <https://metamask.io> (accessed on 10 February 2023).
37. Labs, Web3 Web3j Project. Available online: <http://web3j.io> (accessed on 10 February 2023).
38. JSON-RPC for Ethereum. Available online: <https://github.com/ethereum/execution-apis> (accessed on 10 February 2023).
39. Open Blockchain. Available online: <https://openblockchain.readthedocs.io/en/latest/API/CoreAPI/> (accessed on 10 February 2023).
40. Truffle Suite. Available online: <http://www.trufflesuite.com> (accessed on 10 February 2023).
41. Remix Ethereum. Available online: <http://remix.ethereum.org/> (accessed on 10 February 2023).
42. Google. Google Cloud Storage: Node.js Client. Available online: <https://github.com/googleapis/nodejs-storage> (accessed on 10 February 2023).
43. Angular-Truffle-Dapp. Available online: <https://github.com/ng-es/Angular-Truffle-Dapp> (accessed on 10 February 2023).
44. AWS Amazon. Cloud Data Migration. Available online: <https://aws.amazon.com/es/cloud-data-migration/> (accessed on 10 February 2023).

Disclaimer/Publisher’s Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.