A Digital Evidences Preservation Framework for a Logic Based Smart Contract

Wala'a Al-Omari*[1], Khair Eddin Sabri[2], Nadim Obeid[3]

[1, 2] Department of Cyber Security, College of Computer Sciences and Informatics, Amman Arab University, Amman, Jordan.

[3] Department of Computer Information Systems, King Abdullah II School for Information Technology, The University of Jordan, Amman, Jordan.

E-mail: w.alomary@aau.edu.jo, k.sabri@ju.edu.jo, nadim@ju.edu.jo

* Corresponding author

Keywords: blockchain, chain of custody, cybercrime, cyber security, digital evidence, logic-based smart contract

Received: April 21, 2022

Recently, smart contracts were introduced as a necessity to automatically execute specific operations within blockchain systems. The popularity and diversity of blockchain systems attracted intensive attentions from academia, industry and other sectors. Blockchain systems were implemented using different programming languages that used in defining the triggering events and their consequent actions within the smart contract. In this article, we propose a digital evidences preservation framework that supports logic-based smart contracts to manage entries associated with digital evidences. Combining logic-based approach and blockchain systems may result in ensuing contracts that have technical advantages over procedural coding. The paper shows the motivation for choosing logic-based approach to define a smart contract. We introduce the rules and structure of the proposed logic-based contract.

Povzetek: V članku je predlagan okvir za ohranjanje digitalnih pogodb na osnovi logično temeljene pametne pogodbe z veriženjem.

1 Introduction

Cybercrimes have inundated the cyber world creating several challenges for cybersecurity experts. Since a lack of awareness among the end-users creates a gap that can be employed by attackers to exploit them and/or their organizations. Digital forensics is the science of identifying, analyzing, preserving, documenting and presenting digital evidence from electronic devices [1]. It has gained enormous significance in the investigation process of an incident related to cybercrime. The four main phases of a digital forensics investigation are: Identification, Preservation, Analysis and Presentation. The main issue is how to preserve the gathered data and guarantee its integrity. However, with the advancement in digital forensics techniques, hackers are equally exploiting anti-forensics technology to completely erase digital evidence. Due to its volatile nature, digital evidence has to be acquired and analyzed to ensure the maintenance of the Chain of Custody (CoC), which is the process of validating how any kind of evidence has been gathered, tracked and protected on its way to a court of law. Today CoC of digital evidence is essential and considered as the most vulnerable part of digital investigation process [2].

Preserving digital evidence is a challenging task. Digital evidence can be admissible in court only if the CoC can guarantee exactly what was the evidence, why it was collected and processed, and how evidential data was obtained, analyzed, and reported [3]. Furthermore, at each phase of the investigation, the CoC must identify where, when, and who came into contact with the digital evidence. Because of this complexity, establishing and maintaining a reliable CoC is difficult [4].

Blockchain is a data structure that allows creating a digital ledger for recording and storing transactions shared by all participating parties over a distributed network [5]. It makes use of cryptography for protecting the transactions' processes. In relation to CoC, one important feature of blockchain is the way of recording data in an integrity protected, authenticity guaranteed and non-repudiation way; which makes it a suitable choice to apply in digital forensics. These properties enhance the system reliability compared to a centralized approach. With such motivation, a lot of approaches have proposed using CoC as an application of blockchain technique [6]. Blockchain's capability could potentially create a high preservation system for digital evidences that must be tamper-proof.

Smart contracts have recently been developed on blockchain based platforms. These platforms provide developers with simple interfaces to build smart contract applications. Smart contracts are now being advocated as a way to maximize efficiency, security, and impartiality in the execution of a contract, lowering costs and enhancing confidence between parties. Many incumbent blockchain platforms can support smart contracts. The most representative smart contract platforms are: Ethereum, Hyper Ledger Fabric, Corda, Stellar, and Rootstock. Almost, all recently available platforms are implemented in Solidity, Python, C++, JAVA and PHP [7].

Although procedural languages are primarily used to code smart contracts in current blockchain platforms, declarative languages for such contracts, should be considered to better representation. For example, logic-based rule languages bring about a concept known as declarative smart contracts, specifically logic-based smart contracts [8]. Combinations of logic frameworks and blockchain systems may result in smart contracts that are simpler to work with and have technical advantages over procedural coding of the contracts. Working on a logical system is systematic and clear instead of working on programming languages. Comprehensivity, formal verifiability, time ordering and inference ability (inference-ability) are the main distinguished features of logic based smart contract in comparison with procedural smart contract.

We aim in this work to build a digital evidences preservation framework, enabling a logic-based smart contract to manage entries associated to digital evidences. The framework integrates some related aspects in order to achieve its goal. It utilizes the concepts of blockchain, onchain and offchain storage and logic based smart contracts. The framework exploits the immutability and decentralization features of blockchain technology, as well as the scalability and faster, instantaneous, secure, and cheap storage of offchain technology. Moreover, the framework benefits from simplicity and clearance of logic based smart contract. This study explores the gaps between the procedural smart contract and the declarative logic-based smart contracts in digital evidence preservation phase within CoC process.

This paper is organized as follows: section II presents some related works, in section III an overview of the theoretical background is showed, in section IV our methodology is explained, in section V an illustrative example is clarified and finally, in section VI we summarize the work.

2 Related work

Using blockchain systems in storing data has spread significantly recently, hundreds or even thousands of scientific researches in the literature have addressed this technology because of its practical advantages in preserving data from change and manipulation. For example:

In [9], Cabe et. al., presented blockchain based forensic applications for connected vehicles. To enable the vehicular forensics vision, they introduce a novel blockchain forensic framework.

A blockchain based solution to find facts in criminal incidents in IoT-based systems was proposed in [10]. He proposed a forensic framework which employs a public digital ledger to find facts in criminal incidents within different IoT-based systems. It stores evidence in a form of interactions between devices, users, and cloud (e.g., Things to Users, Things to Cloud, and Things to Things interactions) and keeps them secure in the distributed blockchain network. According to the authors, the system is capable of providing integrity, confidentiality, anonymity, and non-repudiation of the publicly-stored evidence.

In [3], Brotsis et. al., proposed a blockchain based solution to cope with the forensics challenges. They claimed that the primary goal of Cyber-Trust in the smart home domain is to accurately detect the local network’s compromised and/or infected devices to apply the appropriate countermeasures. To combat cyber-attacks and assist the evidence collection, devices’ critical information is recorded on the blockchain so that it can be later queried when a verification of proper functioning is needed; their approach fits well within the practices of software distributors that publish hashes of software binaries to allow verifying their authenticity.

A whole blockchain-based decentralized framework for IoT forensics was proposed in [11]. Blocks are divided into two sections: block header and transaction. The block header is divided into block number, Merkle tree hash and timestamp. The block number is a number sequentially assigned to the generated block. The Merkle tree hash is used by investigators or other participants to locate transactions in a configured blockchain. The timestamp stores the time at which the block was generated. In the proposed framework, blockchain is used for ease of integrity in forensic investigation. Thus, a block is a concept of ‘safe’ storage of data that occurs between device and device, rather than an object that a user has to ‘mining’ competitively.

In the field of cybercrimes and digital evidences CoC and evidence preservation, a lot of research has been conducted utilizing blockchain platforms such as in [5] who implemented Ethereum blockchain to secure the CoC. His Forensic-Chain is a blockchain based solution for maintaining and tracing digital forensics CoC.

In their blockchain based digital CoC system, Le and others in [6], define a group of entities in the framework, and a blockchain platform supported by a group of trusted and distributed servers, for recording all events occurred in the life cycle of the evidence, as well as the entities that are related to the evidence. Each entity is identified through a cryptographic public key, obtained during registration. They further define the entity access rights in the proposed blockchain system.

In [12], Tian et. al., proposed a blockchain based solution for CoC life cycle that was built by one or multiple authority organizations and provided services for the public. It supports evidence collection, storage, verification and retrieval. The evidence collection function collects evidences from the evidence provider, the evidence storage function stores the collected evidences, the evidence verification function verifies the evidence and records the verified evidence in the blockchain. About the evidence retrieval function, it provides evidences or evidence information to authorized requesters, e.g., police officers, lawyers, judges, etc. They proposed a mixed blockchain rather than a full blockchain, as well as a corresponding name-based consensus mechanism. The block bodies are distributed to different nodes. Thus, they simplify the blockchain network and assume that each node is known to each other.

A blockchain based solution to secure the CoC for digital evidences was proposed in [13]. Due to its immaterial nature, digital evidence is especially vulnerable to manipulation, and therefore, it has to be extensively documented and protected during all steps of the forensic process. They suggested how by using blockchain technology a certain piece of digital information can be preserved and routed towards its final destination, the court of law.

In the field of smart contract as a new technology, there are several open researches' issues that exist in the wide implementation of it, wherefore [14] adopted a systematic literature review from 30 relevant studies to understand such current issues and the data from them were extracted before identifying the research gaps. This systematic literature review provided five key requirements that should be in a smart contract and proposed the FarMed framework for creating an intelligent framework that will execute Ethereum smart contact-based reputation systems.

A systematic mapping study was conducted in [15] to collect all research that is relevant to smart contracts from a technical perspective. Their aim of doing so was to identify current research topics and open challenges for future studies in smart contract research. They extracted 24 papers from different scientific databases. The results show that about two thirds of the papers focus on identifying and tackling smart contract issues. Also in this paper, they presented a few research gaps in smart contract research that need to be addressed in future studies. The identified gaps are the lack of studies on scalability and performance issues, the lack of studies on deploying smart contracts on different blockchain platforms other than Ethereum, the small number of the proposed smart contract applications, the lack of studies on criminal activities in smart contracts and the lack of high quality research on smart contracts. Moreover, in their study they illustrated some smart contract issues and their proposed solutions in the literature; one related issue is the complexity of the used programming languages and the proposed solution was the study of [16], which proposed the use of logic-based languages.

In their lab tutorial in [17], the authors found other issue related to smart contract as the difficulty of writing correct smart contracts. The use of formal verification methods is one suggested solution proposed by [18] and [19].

Based on the above related works on blockchain and logic based smart contract, we aim, in this article, to integrate those two technologies so that we can utilize advantages from both of them. While almost all existing works that employ blockchain supposed to implement their smart contract using procedural language, in our proposed framework, we exploit the open issues and the lack of studies within the concept of declarative logic based smart contracts to build our framework.

After mentioned significant works that proposed using blockchain in digital forensics, in next table we identify the areas of research in this filed and mentioned some of the advantages and limitations. The other approaches are summarized in survey reviewed by [13]. See table 1.

2.1 Related work discussion

From the above discussion, we can observe that digital evidence preservation or even CoC opens up various lanes to enhancement and improvement. Although, in this work, we emphasize on the blockchain based systems; but in literature we notice that researches' completeness is missing and can be enhanced further. In general we can say that digital evidence's heterogeneity and compatibility are not addressed correctly.

Moreover, smart contract's security features are not employed at all except in very limited and incomplete works that does not cover all the related functions of the CoC. And if it exists, in addition to its deficiency, it is implemented in a complex way through programming languages that are difficult to deal with and understand.

This also urges for a complete solution to handle issues related to digital evidence preservation CoC and leads us to utilize the functionality of blockchain that allows courts and associated personnel the ability to examine historical CoC without accessing data itself. To address these problems we have come up with the proposed solution to provide a logic based framework for digital evidence CoC preservation process.

3 Background

3.1 Digital evidence' CoC

A crime in which a computer is the object of the crime or is utilized as a tool to commit an infraction is referred to as cybercrime [20]. All digital or electronic sources that can be obtained during an investigation and contain any type of information that could be utilized as evidence in that case are referred to as digital evidence. Moreover, it is a branch of computer forensics that employs a variety of techniques for preserving, identifying, analyzing, examining, authenticating, interpreting, and documenting digital data [20]. A legal technique for collecting, analyzing, storing, and reporting digital evidence is known as digital forensics. Collection, examination, analysis, and reporting are the four phases of a typical digital forensics process [21]. Because such evidence is used to convict suspected of crimes, digital forensics is critical in police investigations. Existing studies have used cloud computing to collect data and then used blockchain to ensure that the data is transparent, immutable, and auditable. Unfortunately, such investigations only use a rudimentary security model and do not cover the complete evidence life cycle.

Table1: Related works' summary

Suggested Work	Reference	Features	Limitations
Vehicular forensic investigation	(Cebe, 2018) [9]	The solution eliminates the need of a trusted third party by providing a lightweight privacy-aware mechanism for vehicles. Fragmented ledger is used with consensus.	The weakness is the integrity between the fragmented ledger and the shared ledger is not addressed.
Digital forensics extracted from cybercrimes	(Le, 2018) [6]	Authors proposed a blockchain-based IoT forensics framework with identity privacy. It provides a complete workflow of the evidences from gathering to disposing. Privacy of the identity is imposed using Merkle tree signature.	The heterogeneity of the devices is not addressed in this work.
Forensic-chain	(lone, 2019) [5]	The proposed a blockchain-based solution emphasizing on Chain of Custody (CoC).	The end to end complete framework is missing in this work.
Forensic investigation framework	(Hossain, 2018) [10]	The work uses a public digital ledger to validate criminal events in an IoT-based environment; It collects interactions of IoT entities as evidences and stores them securely as transactions in a blockchain network.	As the IoT devices are resource-constrained, the complexity of the proposed approach is an issue and can be improved.
Secure blockchain-based digital evidence framework	(Tian, 2019) [12]	In this work, the original evidences are stored in a secured storage and evidence information is stored in blockchain. Multi-signature is used for evidence submission and retrieval.	The mapping between the storage and evidence information needs to be more secure.
Blockchain for IoT forensics	(Ryu, 2019) [11]	This work uses a public distributed ledger and all the IoT devices' communications are considered as transactions for storing in blockchain.	The complexity of the approach can be considered as the main weakness of the work.

Billions of devices connected to the internet generate a massive amount of data that must be stored and accessed, providing significant challenges in protecting the integrity and authenticity of digital evidence for its admission in a court of law. Digital evidences provide particular issues since they are latent, volatile, and fragile, can transcend jurisdictional borders fast and readily, and are often time/machine dependent. As a result, in a digital world, ensuring the validity and legality of methods and procedures used to obtain and transfer evidence is a significant difficulty. Digital evidence is considered admissible in the court of law if it meets following criteria: authentic, complete, reliable and believable [3]. The capacity of blockchain technology to provide a comprehensive view of transactions back to that it can solve challenges relating to trust, integrity, transparency, accountability, and reliable data sharing.

In terms of CoC, the blockchain's ability, when combined with cryptographic hashing and encryption, might potentially generate tamper-proof information pertaining to evidence access. The evidence that needs to be kept is first encrypted capability and data would be accessible only to the desired party on the blockchain, but smart contracts would automatically record the time, date, and perhaps user-ID of the accessing party and add it to the unalterable record in the blockchain [4]. The blockchain can be read in a similar way to how the bitcoin blockchain can be decoded using a special function. This feature of blockchain allows judges and other related personnel to analyze the historical CoC without requiring accessing the material itself [9].

3.2 Blockchain and smart contract utilization

A blockchain is a continuously-growing chain of blocks. When a new block is generated, all the nodes in the network will participate in validating the block. Moreover, the privacy of users is also preserved because users transact with their virtual addresses instead of real identities [22]. Blockchain has become very popular technology to digitally create and manage transactions. It is a form of distributed public ledger which analyses and verifies transactions in decentralized manner and data is not under the control of any third party.

The blockchain technology, which was first used for the Bitcoin cryptocurrency, establishes a decentralized fully replicated append-only ledger in a peer-to-peer network. Every node in the network keeps a full local copy of the blockchain. The blockchain is made up of a number of blocks that include the ledger's transactions. As shown in figure 1, each block contains a cryptographic hash of the previous block in the chain, and transactions within blocks are sorted chronologically. As nodes receive transactions, they build new blocks, which are broadcast throughout the network. When a block is finished, miners begin the consensus process in order to persuade other nodes to add it to the blockchain.

Nodes compete with each other by solving a mathematical puzzle to confirm transactions and create new blocks. Solving a block results in mining a specific quantity of bitcoins, which is the reward for block authors, to incentivize such a process (called miners). Multiple miners may generate valid blocks at the same time, resulting in forks in the chain. To resolve forks, only the longest branch is accepted as a valid continuation of the chain [22].

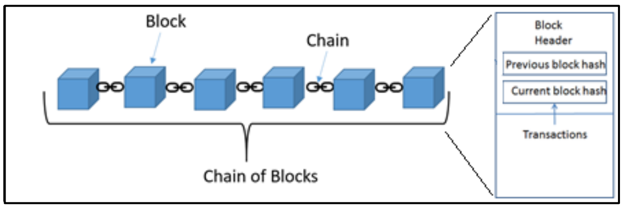

Figure 1: Blockchain Structure

Blockchains are enabling smart contracts. The term smart contract was originally introduced in the 90s by Szabo [16]. A smart contract is a piece of code that resides on a blockchain and is identified by a unique address. It includes a set of executable functions and state variables [23], and implemented on top of blockchains. Smart contracts are a set of Scenario-Response procedural rules and logic, they are decentralized, trusted shared codes that deployed on blockchain and the parties signing a contract should agree on contractual details [24].

Blockchain is a distributed, shared ledger where transactions are digitally recorded. It is considered as a time-stamped collection of immutable data records controlled by a cluster of computers and not owned by a single entity. These contracts are often used to enforce some kind of agreement and to conduct transactions in a distributed environment without the involvement of a trusted third party [25]. In a smart contract, contract clauses written in computer programs will be automatically executed when predefined conditions are met. Smart contracts consisting of transactions are essentially stored, replicated and updated in distributed blockchains. One party who breaches the contract will be punished automatically. It is worth noting that a smart contract consists of a number of declarative statements with logical connections. When a condition is triggered, the corresponding statement will be automatically executed, consequently a transaction being executed and validated by miners in the blockchains. The committed transactions and the updated states have been stored on the blockchains thereafter. Smart contracts generally have two attributes: value and state. The triggering conditions and the corresponding response actions of the contract terms are present using triggering condition statements [26].

Smart contracts guarantee appropriate access control. In particular, developers can assign access permission for each function in the contract. Once any condition in a smart contract is satisfied, the triggered statement will automatically execute the corresponding function in a predictable manner [14]. The key features of the smart contracts are execution in peer to peer mode without the intervention of a centralized third party and service availability without any centralized dependency. Smart contracts can cut down administration and save services costs, improve the efficiency of business processes and reduce the risks.

3.3 Logic-Based smart contract

When programming in a procedural language, the programmer must write each step that the smart contract must take, which can be time-consuming and error-prone because the order of instructions affects the correctness of the smart contracts. As a result, some researchers propose using declarative languages to represent and reason about smart contract rules [27]. With the logic approach, contractual clauses are rephrased into explicit formal statements which are separated from the embedding program, and the program has inferential functionalities to reason upon these statements. In practice, the contractual clauses would be encoded into logic rules, and a rule-based engine would reason upon the rules.

3.4 Why Logic-Based smart contract?

To shed light on the features of logic based contracts, this section will explain the life cycle of smart contract, and through its stages, what distinguishes logic based smart contracts from procedural ones [16]:

1- Creation/ Negotiation: A smart contract, like any other ordinary contract, is developed through negotiation. Before deploying a contract on a blockchain system, agreement on what the contract should do is defined. The contract establishes legal connections between the parties after it is created and it is frequently written in natural language before being transformed into a smart contract. The procedural coding of a smart contract may appear hard to comprehend and slowing the formulation of the contract because it appears to be complex and one would doubt if the contractual terms are appropriately coded. Per contra logic statements used in logic-based smart contracts may be simpler and understood as high-level specifications and provide executable specifications that can be directly performed, that reduce the chance of implementation errors.

2- Validation: A logic representation can make validation easier by utilizing approaches like formal verification to determine if particular properties are true. According to [28], a logic approach of programming smart contracts has technical advantages over procedural coding of the contracts by including inherently self-verification of the smart contracts logic.

3- Contract Storage: Contract storage is synonymous with the storage of smart contracts on database systems. The smart contract can be stored on a blockchain system. So, the logic-based contract may reduce the cost of storage, because logic statements are generally more compact than their procedural counterpart. According to [28], code length in smart contacts has to be short pieces of code. With the increasing length or complexity of a smart contract, the time required for committing the associated transactions is higher and such is the gas consumption.

4- Modification: A contract cannot be amended in current blockchain systems, although the data held in it can be updated using some paradigms that allow modification of smart contracts. The statements can be implemented as 'public' variables in logic-based smart contracts, allowing for more fine-grained modifications. Furthermore, with the procedural model, the sequencing of instructions and procedures is critical, and the paradigm may pose some complications in this regard. So, a logic-based language can be extremely effective when dealing with modifications, because the order of statements has no bearing on the inferences that can be drawn from them. According to [28], if the smart contract is supposed to modify the state of the Blockchain and generate other transactions, these have to be carefully ordered in order to avoid unwanted transfers. Moreover, logic based smart contracts can be distinguished from procedural ones through other aspects, such as:

5- Dispute Resolution: Smart contracts can be considered legally legitimate in theory, regardless of whether they are developed using an imperative or declarative language. However, because certain imperative code might be difficult to understand, it's possible that the control structures of these smart contracts thwart jurists' interpretations of the contract [16]. On the contrary, because logic rules are intended to mirror contractual contracts, their representation in logic will make it easier for jurists to formulate, assess, and compare legal arguments based on formal statements. The logic rules may thus make the contract's implementation or interpretation easier, but they may not be close enough to natural language, especially for those who aren't technical specialists.

6- Vulnerability: Comparing to smart contracts written in Solidity, logic based smart contracts are not vulnerable to out of order execution (reentrancy attacks by breaking the sequence of functions or cross calling of functions) [7].

4 Methodology

4.1 The proposed blockchain framework

In this research we proposed to store the acquired digital evidences on offchain, but the logs (transactions) of storing and retrieving data from offchain will be stored in the blockchain, in other words, to use offchain to store data while using blockchain to monitor and restrict access to this data, supposing that approach solves many of the challenges that faced blockchain and enhances some security properties.

Since transactions can be costly, it is often advised that heavy computation should occur ‘offchain’ instead of ‘on-chain’. In offchain scenarios, computation is performed outside the blockchain based system, while, in onchain scenarios, computation is performed and validated in the blockchain based system. Of course, offchain computation results can be recorded in a blockchain. We used an offchain centralized data store for private data and only used blockchain application for storing public data. A reference is then recorded on the blockchain once these set of transactions have successfully completed. A hash for the offchain item is generated and that is what will be stored in the blockchain. It is expected that the required storage for offchain data will exceed the needs of blockchain storage.

To elaborate more on building the proposed blockchain framework; we gave a general description of the essential elements constitute a blockchain framework: the data to be stored, transactions, smart contract, blocks, consensus and the mining process:

- The Data: The data is the digital evidences and their attributes. Digital evidence is any provable information stored or transmitted in digital form that a party may use in a trial to a court case. Digital evidences are collected by authorized parties (usually police officers and digital evidences analysts). Digital evidence is divided into expressive evidence and computerized evidence. Expressive evidence is an item or archive viewed as illustrative proof when it straightforwardly shows a reality. It's a typical and dependable sort of proof. Instances of this sort of proof are photos, video, sound chronicles and diagrams. While computerized evidence can be such an advanced record from an electronic source. This incorporates email, instant messages, texts, records and archives extracted from hard drives, GPS locations, logs, addresses, sensors data, electronic monetary exchanges and any media documents.

- Transaction: Each transaction corresponds to an action performed by one or multiple participants. Its structure carries not only necessary information to describe the action (e.g. evidence submission), but the unique identifier to differentiate itself from others. To provide a detailed description of an action, the transaction type, time stamp and other supplementary data are included to the transaction.

- Smart Contracts: Each transaction could be associated with one or more smart contracts. The smart contract first retrieves the identity of the submitter along with the evidence data, and then initiates a case instance with the information obtained from the submitted transaction.

- Block: Block is a data structure bearing a number of verified transactions performed during a specific time frame. The formation of blocks is executed by the miners, by collecting and verifying all newly submitted transactions in the network. Each block contains block identifier, body, payload, and the digital signature field.

- **Consensus:** To ensure that the entire community has a unified view of the current network state; we suggest utilizing a consensus protocol called Byzantine Fault Tolerance (BFT), which is typically used in a permissioned blockchain. For each pre-defined epoch, the system selects one "leader" from the designated entities. This leader then collects the unconfirmed transactions, forms a block, and includes its ID into the miner ID field. This particular block is then broadcast to the entire network and is verified by the community. Once the number of successful verification passes a pre-defined threshold, this particular block is considered as valid and written into the immutable ledger. The "proof-of-consensus" to be included into the blocks is the digital signatures generated by the entities who have successfully verified the blocks.

- **Mining Process:** The block is broadcasted to all nodes in the network. One of the nodes validates the block (which called mining in bitcoin) and broadcasts it back to the network. The nodes add the block to their chain of blocks if the block is verified and the block correctly references the previous block.

In order to control access to data; we enabled smart contract, a set of permissions that can be granted to the parties involved. Smart contracts used for writing data and reading from the offchain database. We identified the permissions of reading and writing using smart contracts. All the logs of reading and writing will be stored in the blockchain. We analyzed the literature to consider the most important contracts to be included in the blockchain.

Without compromising generality, and for simplicity of presentation, we considered single digital evidence (DE) collected by an authorized entity (EN) that holds its ownership. During investigations, several authorized entities (e.g., police offices, lawyers, judges, magistrates, etc.) may need to access, acquire and/or own temporarily DE. The set of authorized entities that can interact with DE is denoted with (DE-EN). Each authorized entity has a unique identifier known to all and he/she owns credentials that allow him/her to be authenticated and take actions in the DE CoC process.

At each time (T), DE can have just one owner and the owner must belong to DE-EN. If an authorized entity (ENi) needs to acquire and own DE, the current owner needs to issue a transfer request towards ENi. The change of ownership happens if and only if the new owner belongs to the set of authorized entities in the framework (ENi ϵ DE-EN), and the transfer record is written permanently in the evidence log.

4.2 The system architecture

The proposed framework's architecture bases on a private and permissioned blockchain. This choice has been driven by the authentication requirement of the process that does not allow un-authorized and un-trusted parties to manage digital evidences.

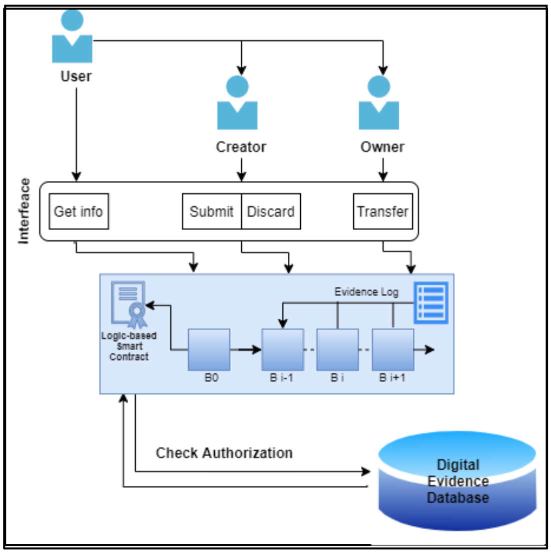

Figure 2: The whole proposed framework's architecture

As shown in figure 2, the framework is composed mainly of three components: The Evidences Database (DB), the Evidence Log and the Interface.

- **Evidence DB.** The Evidence DB is an ordinary offchain database or file repository where we store the actual digital evidences, while related data are stored in the Evidence Log, which is implemented through the blockchain technology. We proposed to store the acquired digital evidences on offchain, in order to some of the challenges which face blockchain and enhance some security properties. First of all, evidences can be too large to be efficiently stored in the blockchain (for example, an evidence may be several TBs of capacity). Secondly, and most importantly, if evidences were stored in the blockchain, every node in the blockchain network would have access to them, while by storing them in an offchain database; only authorized nodes should be allowed to acquire evidence. Therefore, we store in the blockchain only the information regarding the DE CoC process and a hash of the evidence which allows verifying evidences integrity during acquisition.

The original digital evidence is stored along with an identifier ID, obtained as the hash of the evidence and a nonce (to guarantee uniqueness of IDs). This database is distributed and is managed by trusted entities. Moreover, each access is executed only if the requesting entity is authorized to perform such access according to its role.

In the next phase in this work, we will investigate the different attributes and relations of the stored digital evidences in order to build a comprehensive Ontology. The goal is to represent the data in a way that is correct, complete, and secure using Ontology. We will represent the Ontology as description logic to remove ambiguity and to make some reasoning and moreover, to be combatable with our proposed logic based smart contract. The data is the collected digital evidences; this data will be stored offchain in the Evidence DB to maintain scalability. The logs (transactions) of storing and retrieving data from offchain will be stored in the blockchain within the Evidence Log. In other words, to use offchain to store data while using blockchain to monitor and restrict access to this data. In this way we maintain the integrity of data and verify its owner. We plan to identify the stored digital evidences with their unique IDs which will be a “Data Property” for each evidence. The IDs will be connected somehow to other objects, such as: crime, source and related persons.

- Evidence Log. The Evidence Log is represented through the blockchain technology, for each evidence, its ID, a description, the identity of the submitter (which we call creator) and the complete history of owners up to the current one, including the time at which changes of ownership occurred. While the evidence itself is not stored in the blockchain, the ID allows verifying that the evidence has not been tampered with, provided that a robust cryptographic hash function is used to generate it.

The evidence log is implemented on top of a peer-to-peer network composed by all authorized entities. Such network can be decomposed in two sets of nodes:

- Validator nodes: this is the set of nodes that must be preventively authorized with the role of validators in the permissioned blockchain.
- Lightweight nodes: they can be seen as clients of the chain since they simply issue transactions and need to rely on validators for adding and validating their transactions.

The Evidence Log comprises four basic functions for Submitting, Discarding, Transferring and Getting information of evidence from the Blockchain (The Evidence Log). These functions can be viewed as transactions triggered by the participants in the network. Constraints such as who should access what function and under what conditions access should be granted to participants, are all defined in access control rules.

- Interface. It is a mediator that stands between the framework and its users. It is a local instance runs on each node and interacts with the Evidences DB and the Evidence Log (through a local blockchain node).

4.3 Logic-based smart contract

The Evidence Log runs a logic-based smart contract which will be a set of permissions that can be granted to the parties involved. Smart contracts will be used for writing data and reading from the offchain database. All the logs of reading and writing will be stored in the blockchain.

The mediator interface generates the ID for DE using a nonce N, stores (ID; N; DE) in the Evidence DB and issues the Submit transaction in the Evidence Log. As already discussed; the submitter is also registered as the first owner in the blockchain.

The creator of an evidence DE can request to discard it from the system (e.g., because it is no more legally valid). If he/she is authorized to do so, the corresponding entry is removed from the map of evidences by issuing the Discard transaction. If the transaction succeeds, the corresponding evidence is deleted from the Evidence DB.

When a user wants to acquire an evidence DE, the interface sends a request to the Evidences DB which will serve the request only if the user is the current owner of DE. This check is performed by interacting with the Evidence Log. The change of ownership of an evidence DE is performed by issuing Transfer transaction specifying the new owner.

Finally, every user in the network can query the Evidence Log to get the entry of evidence (which contains all relevant information except the evidence itself). This is performed by simply issuing the Get-Info transaction.

As previously illustrated, smart contract manages entries associated to digital evidences]. In this section we will explain parts of the smart contract:

Evidence definition contains the ID, the creator's address, the owner's address, a string field to hold the evidence description, and two arrays taddr and ttime that store the evidence handovers and the times at which they occurred, respectively. From the creator to the current owner, these arrays are sorted chronologically. When an authorized user submits a new digital evidence DE to the system, he/she takes the role of creator of DE. All evidence items are stored in a map indexed by evidence IDs stored in the DB.

The Evidence Log primitives are implemented by four functions in the smart contract:

The Submit (ID, description) function creates a new Evidence entry with the specified ID and description, and the address of the related transaction sender as the creator and current owner of the evidence.

Submit function takes Evidence ID and Evidence Description as input and submits the evidence to the blockchain (The Evidence Log). Other attributes like creator and owner are also set to participant address that created it first time. Participant address is pushed to taddr array thereby indicating it is the creator as well as first owner of the digital evidence. Evidence creation time is pushed to ttime array.

Note, Evidence Submit function first checks whether the evidence exists with the same ID, if so it returns without creating the duplicate evidence. Pseudocode of the function is presented below in the form of algorithm [5].

```
Algorithm 1: Submit new evidence
Input: Evidence ID, Evidence Description
Result: Creates the evidence with appropriate values in The Evidence Log
if evidence_exists then
return
else
    Set the evidence attributes with Evidence Description
    Set participant (who invoked this function) address as creator and owner of the evidence
    Push the address to taddr array
    Push the current time to ttime array
```

The Transfer (ID, newowner) function transfers the ownership of the evidence identified by ID to the entity identified by the address newowner. Note that only the current owner of evidence can transfer ownership.

Evidence Transfer method takes Evidence ID and address as input and in return transfers the ownership to address supplied. The function first checks whether evidence exists and the participant who invokes the function is the owner of the evidence if so, it sets the evidence owner to the new owner. It also pushes a new owner to the taddr array and current time to ttime array thereby maintaining the auditable chain pertaining to evidence transfer. Pseudocode of the function is presented below in the form of algorithm [5].

```
Algorithm 2: Transfer evidence
Input: Evidence ID, Address
Result: Transfers the evidence to the appropriate address in The Evidence Log
if evidence_exists & owner then
        Set the evidence owner to new owner.
        Push the address to taddr array
        Push the current time to ttime array
else
return
```

<u>The Discard (ID)</u> function removes evidence from the map of evidences. No further operations can be performed on removed evidence. Note that only the creator of evidence can remove the evidence.

Evidence Discard function takes Evidence ID as input and deletes the corresponding evidence from the map of evidences. If the transaction succeeds, the corresponding evidence is deleted from the Evidence DB. Evidence Log. It first checks whether evidence exists and participant who invokes it is the creator of the evidence if so it removes the evidence entry from map. Pseudocode of the function is presented below in the form of algorithm [5].

```
Algorithm 3: Discard evidence
Input: Evidence ID
Result: Removes the evidence from the evidences' map
if evidence_exists & creator then
Remove the evidence entry from the evidences' map
else
return
```

<u>The Get-Info (ID)</u> function returns all fields of an evidence entry. This function takes Evidence ID as input and returns the evidence information from Blockchain. The only check this function does is to ensure evidence already exists. Pseudocode of the function is presented below in the form of algorithm [5].

```
Algorithm 4: Get-Info of evidence
Input: Evidence ID
Result: Displays the appropriate evidence instance from The Evidence Log
if evidence_exists then
Return the evidence view from The Evidence Log
else
return
```

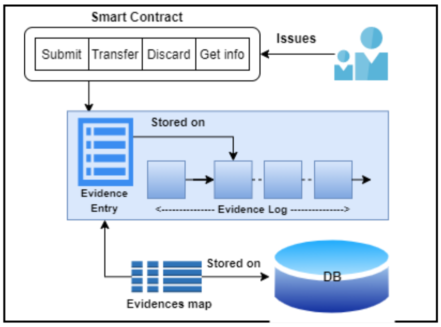

Figure 3: Detailed architecture

The details about the proposed framework's architecture are illustrated in next figure 3.

As illustrated previously in fig.2, in this proposed framework we have three types of users: general user, evidence creator and evidence current owner. Privileges are granted to their roles, and roles are granted to those users, to specify the operations that they can perform on digital evidence. These privileges enhance the authorization system as follows:

TABLEL 2: FRAMEWRORK'S USERS PRIVILEGES

Users	Privileges
General user	Get-Info
Creator	Submit, Read, Transfer, Discard
Owner	Read, Transfer

We should differentiate between Get-Info and Read privileges. Get-Info returns all fields of a previously defined evidence entry except the evidence itself, but suppose that one authorized entity wants to acquire one of the stored digital evidences and possess it in any stage for investigation purpose not just viewing its definition, that means he/she wants to examine the digital file itself, so we need Transfer function to allow any authorized entity to have access to own and inspect the evidence which we call it (Read) privilege. Only authorized entities can issue Transfer transaction and own the evidence. Those authorized entities are directly involved in investigation process, so no need to allow the creator to revoke his/ her transfer of ownership and be the owner again because this may disrupt the investigation process.

5 Illustrative example

A well-known example and one of the most controversial criminal cases is a murder crime for which evidences were obtained from Fitbit data, which is an IOT health care device [29], the acquired data was used as digital evidences for this murder case that took place in Connecticut in December 2015 [30]. Events can be summarized as follows [31]:

- Connie Dabate was a victim of a murder crime and her husband Richard Dabate was a suspect of this murder.

- Connie Dabate was wearing Fitbit on the day of the crime, which is a Health Care Device.
- Data retrieving tool was used as a forensic tool by police investigators.
- Police investigators analyzed Fitbit, retrieved Fitbit Data and considered them as Digital Evidences of this crime [32], which are:
  - Fitbit Time Record was a retrieved record.
  - Fitbit Location was a retrieved GPS location.
  - Fitbit Movement was a retrieved sensor data.
- Two police investigators were investigating this crime: investigator A & investigator B, and one lawyer and one judge.

We will give notations IDs for the digital evidences (DE) and authorized entities (EN) as follows:

TABLEL 3: NOTATION IDs

Item	Notation ID
Time Record	***DE1***
GPS Location	***DE2***
Movement sensor data	***DE3***
Investigator A	***EN1***
Investigator B	***EN2***
Lawyer	***EN3***
Judge	***EN4***

As previously explained, digital evidences are collected by authorized parties who become their temporary (first) owners. During investigation, several authorized entities may need to access, acquire and/or own temporarily DE; the set of authorized entities that can interact with DE is denoted with (DE-EN). So, in this example:

DE-EN = { EN1, EN2, EN3, EN4 }

At each time, DE can have just one owner and the owner must belong to DE-EN.

The smart contract was used for writing and reading data from the off-chain database. All the logs of reading and writing were stored in the Evidence Log that represented by blockchain. The interface generates the ID for DE using a nonce N, stores (ID; N; DE) in the Evidence DB and the submitter is registered as the first time owner in the Evidence Log. In this example we will suppose that investigator A submitted a Fitbit time record and GPS Location, while investigator B submitted a Fitbit movement sensor data. We can summarize that as follows:

TABLE 4: DIGITAL EVIDENCES' SUBMITTERS

Digital Evidence ID	First Owner (submitter)
DE1	EN1
DE2	EN1
DE3	EN2

The Evidence Entry to The Evidence Log contains the evidence ID, a string field to hold the evidence description, and two arrays (handovers-addresses) array and (handovers-times) array that store the evidence handovers and the times at which they occurred, respectively. From the creator to the current owner, these arrays are sorted chronologically.

All evidence items are stored in a map called Evidences [], indexed by evidence IDs. For example to insert digital evidence (DE1) to this map, we need to define it as follows:

```
Evidence Definition:
ID = DE1;
Description = “Fitbit Time Record”;
taddr = DE1- taddr [];
ttime = DE1- ttime [];
```

And then index its ID to the Evidences [] map by:

```
Mapping Evidences [DE1];
```

Then to investigator A can submit DE1 to the Evidence Log, the interface issues the Submit function at time T0:

```
Submit (DE1, “Fitbit Time Record”):
Evidence Log [on available block] =
Evidence. ID = DE1;
Evidence .Owner address = EN1-address;
Evidence. Creator address = EN1-address;
Evidence. Description = “Fitbit Time Record”;
Evidence. taddr address = Push. DE1- taddr [EN1-address];
Evidence. ttime = Push. DE1- ttime [now T0];
```

Suppose that the Lawyer (EN3) at time T1 wants to acquire evidence DE1 for any reason, the interface sends a request to the Evidences DB which will serve the request only if the user is the current owner of DE. This check is performed by interacting with the Evidence Log. The change of ownership of evidence DE1 is performed by issuing Transfer transaction specifying the new owner. So, EN1, which is the current owner of DE1, must send transfer transaction which transfers the ownership of DE1 to the new owner entity EN3 by his address. Bellow, the interface issues the Transfer function at time T1:

```
Transfer (DE1, EN3-address):
Evidence [DE1] .Owner address = EN3-address;
DE1- taddr [] = Push. DE1- taddr [EN3-address];
DE1- ttime [] = Push. DE1- ttime [now T1];
```

Suppose that the Lawyer (EN3) which is now the current owner of DE1 wants to transfer the evidence (DE1) to Investigator B (EN2), he must apply Transfer function too at time T2 as bellow:

```
Transfer (DE1, EN2-address):
Evidence [DE1] .Owner address = EN2-address;
DE1- taddr [] = Push. DE1- taddr [EN2-address];
DE1- ttime [] = Push. DE1- ttime [now T2];
```

After completing his task, Investigator B (EN2) which is the current owner of DE1, must return the ownership of DE1 to its creator which is Investigator A (EN1) to discard it from the database if he wants, by applying Transfer function also. The creator of an evidence DE can request to discard it from the system, the corresponding entry is removed from the Evidence map. Note that only the creator of evidence can remove it.

Suppose that at any time after submitting (GPS Location) as second digital evidence (DE2), Investigator B (EN2) which is the creator of it, wants to delete it from the Evidence map and DB. The interface issues the Discard function which discards DE2 from the map of evidences as bellow:

```
Discard (DE2):
Delete    Evidences [DE2] ;
```

Moreover, every user in the framework can query the Evidence Log to get the entry of evidence (which contains all relevant information except the evidence itself); this is performed by simply issuing the Get-Info transaction.

Suppose that Judge (EN4) at any time wants to get information about DE1, EN4 must issues the Get-Info function which returns all fields of DE1.

```
Get-Info (DE1):                        = View:
Evidences [DE1]. ID;                   = DE1
Evidences [DE1]. Owner;                = EN2
Evidences [DE1]. Creator;              = EN1
Eevidences [DE1]. Description;         = “Fitbit Time Record”
DE1- taddr [];                         = [T0, T1, T2]
DE1- ttime [];    = [T0= Investigator A Submission
                     T1= Investigator A to Lawyer Transfer
                     T2 = Lawyer to Investigator B Transfer]
```

6 Conclusion & future work

This research proposed a digital evidences preservation framework, enabling a logic-based smart contract to manage entries associated to digital evidences. One phase in CoC process is how to preserve digital evidence during its way to a court of law. To achieve this goal, this work exploited the immutability and decentralization features of blockchain technology. Moreover, the work benefited from simplicity and clearance of logic based smart contract and explored the gaps between the procedural smart contract and the declarative logic based smart contracts. Furthermore, the paper showed the motivation of choosing logic approach to define the contract and introduced the rules and structure of the proposed logic based smart contract.

As a next work, any other related issues such as analyzing duplicating and inconsistency between records may be considered as a promising future works. Also, to build a better authorization system we can add more fields that would allow us to provide an extra privilege to a user on all information related to a specific case or from a specific source. In our next work we will give full specification about the proposed logic based contract.

Moreover and as a future work, we will investigate different attributes and relations about digital evidences in order to build a comprehensive Ontology of the collected stored digital evidences. We propose to represent the Ontology as description logic to remove ambiguity, make some reasoning and to be combatable with our proposed logic based smart contract.

References

[1] M. Alghamdi, "Digital forensics in cyber security—recent trends, threats, and opportunities", Cybersecurity Threats with New Perspectives, IntechOpen, 2021.

[2] S. Bonomi, M. Casini and C. Ciccotelli, "B-coc: A blockchain-based chain of custody for evidences management in digital forensics", arXiv preprint arXiv:1807.10359, 2018.

[3] S. Brotsis, et al., "Blockchain solutions for forensic evidence preservation in IoT environments", 2019 IEEE Conference on Network Softwarization (NetSoft), IEEE, 2019.

[4] F. Alruwaili, "CustodyBlock: a distributed chain of custody evidence framework", Information 12.2: 88., 2021.

[5] Lone and R. Mir, “Forensic-chain: Blockchain based digital forensics chain of custody with PoC in Hyperledger Composer”, Digital investigation, 28, 44-55., 2019.

[6] D. Le, H. Meng, L. Su, S. Yeo and V. Thing, “Biff: A blockchain-based IOT forensics framework with identity privacy”, In TENCON, IEEE Region 10 Conference (pp. 2372-2377), IEEE, 2018.

[7] Z . Zheng, “An overview on smart contracts: challenges, advances and platforms”, Future Generation Computer Systems, 105, 475-491, 2020.

[8] G. Governatori, et al., “On legal contracts, imperative and declarative smart contracts, and blockchain systems”, Artificial Intelligence and Law, 26(4), 377-409, 2018.

[9] M. Cebe , E. Erdin , K. Akkaya , H. Aksu and S. Uluagac, “Block4forensic: An integrated lightweight blockchain framework for forensics applications of connected vehicles”, IEEE Communications Magazine, 56(10), 50-57, 2018.

[10] M. Hossain, Y. Karim and R. Hasan, “FIF-IoT: A forensic investigation framework for IoT using a public digital ledger”, In 2018 IEEE International Congress on Internet of Things (ICIOT) (pp. 33-40), IEEE, 2018.

[11] J. Ryu, P. Sharma, J. Jo and J. Park, “A blockchain-based decentralized efficient investigation framework for IoT digital forensics”, The Journal of Supercomputing, 75(8), 4372-4387, 2019.

[12] Z. Tian, M. Li, M. Qiu, Y. Sun and S. Su, "Block-DEF: A secure digital evidence framework using blockchain", Information Sciences, 491, 151-165, 2019.

[13] M. Stoyanova, Y. Nikoloudakis, S. Panagiotakis, E. Pallis and K. Markakis, "A survey on the internet of things (IoT) forensics: challenges, approaches, and open issues", IEEE Communications Surveys & Tutorials, 22(2), 1191-1221, 2020.

[14] Almasoud, F. Hussain and O. Hussain, "Smart contracts for blockchain-based reputation systems: A systematic literature review", Journal of Network and Computer Applications, 170, 102814, 2020.

[15] M. Alharby and A.Van Moorsel, "Blockchain-based smart contracts: A systematic mapping study", arXiv preprint arXiv: 1710.06372, 2017.

[16] F. Idelberger, G. Governatori, R. Riveret and G. Sartor, "Evaluation of Logic-Based Smart Contracts for Blockchain Systems", 9718. 10.1007/978-3-319-42019-6_11, 2016.

[17] K. Delmolino, M. Arnett, A. Kosba, A. Miller and E. Shi, "Step by step towards creating a safe smart contract: Lessons and insights from a cryptocurrency lab", In international conference on financial cryptography and data security (pp. 79-94), Springer, Berlin, Heidelberg, 2016.

[18] K. Bhargavan, et al., "Formal verification of smart contracts: Short paper", In proceedings of the 2016 ACM workshop on programming languages and analysis for security (pp.91-96), 2016.

[19] G. Bigi, A. Bracciali, G. Meaccia and E. Tuosto, "Validation of decentralised smart contracts through game theory and formal methods", In Programming Languages with Applications to Biology and Security (pp. 142-161), Springer, Cham, 2015.

[20] Pandey, et al., "Current challenges of digital forensics in cyber security", 10.4018/978-1-7998-1558-7.ch003, 2020.

[21] R. Montasari, R. Hill, S. Parkinson, A. Hosseinian-Far and A. Daneshkhah, "Digital forensics: challenges and opportunities for future studies", International Journal of Organizational and Collective Intelligence, 10. 10.4018/IJOCI.2020040103, 2020.

[22] J. Muhr and T. Laurence, Blockchain for Dummies, 2rd ed., John Wiley & Sons Incorporated, 2017.

[23] Bahga and V. Madisetti, "Blockchain platform for industrial internet of things", Journal of Software Engineering and Applications, 9(10), 533-546, 2016.

[24] S. Wang, et al., "An overview of smart contract: architecture, applications, and future trends", In 2018 IEEE Intelligent Vehicles Symposium (IV) (pp. 108-113), IEEE, 2018.

[25] S. Rezaei, E. Khamespanah, M. Sirjani, A. Sedaghatbaf, and S. Mohammadi, "Developing safe smart contracts", In 2020 IEEE 44th Annual Computers Software and Applications Conference (COMPSAC) (pp. 1027-1035), IEEE, 2020.

[26] M. Li, C. Lal, M. Conti and D. Hu, "LEChain: A blockchain-based lawful evidence management scheme for digital forensics", Future Generation Computer Systems, 115, 406-420, 2021.

[27] J. Hu and Y. Zhong, "A method of logic-based smart contracts for blockchain system", In Proceedings of the International Conference on Data Processing and Applications (pp. 58-61), 2018.

[28] Stancu and M. Dragan, "Logic-based smart contracts", In World Conference on Information Systems and Technologies (pp. 387-394), Springer, Cham, 2020.

[29] Rodis, "Fitbit Data and the Fourth Amendment: Why the Collection of Data from a Fitbit Constitutes a Search and Should Require a Warrant in Light of Carpenter v. United States" Wm & Mary Bill Rts, 29, 533, 2020.

[30] T. Khairallah and A. Shamlawi, "Wearables as Digital Evidence", 2019.

[31] D. Altimari, "A marriage marked by secrets, a murder case months in the making", [Online] Available: http://www.courant.com/news/connecticut/hc-ellington-murder-fit-bit-20170422-story.html.

[32] N. Black, "Fitbit data, other digital evidence used by prosecution in murder case" [Online] Available: http://www.legalnews.com/washtenaw/1443244