Chain of Custody and Evidence Integrity Verification Using Blockchain Technology

Adir Miller and Avinash Singh
University of Pretoria, South Africa

u20692286@tuks.co.za
asingh@cs.up.ac.za

Abstract: The validity and integrity of digital evidence and the chain of custody are crucial to all digital forensic investigations. All new evidence and access logs of the original evidence should be logged in a document called the 'chain of custody'. This document shows the timeline of any piece of evidence from the time it was recorded until the end of the investigation. In a traditional digital investigation, trusted parties, such as an investigator, are allowed access to the digital evidence and follow a strict process when dealing with data. These trusted parties have the capability to alter the data making the evidence inadmissible in a court of law. Alternatively, these trusted parties may also alter the data accidentally or with malicious intent, due to a lack of transparency and non-repudiation. Blockchain technology can solve this issue, however, existing research shows that adopting blockchain does not provide adequate transparent access control mechanisms. Consequently, this makes blockchain difficult to adopt due to the one-to-one mapping and the inability to easily validate the chain of custody and evidence admissibility. Current methodologies rely on an external off-chain access control mechanism, which, regrettably, remains susceptible to potential breaches that could compromise its integrity and validity. This paper proposes an enhanced model to provide access control through smart contracts, ensuring immutability, flexibility, transparency, and non-repudiation of both the access control mechanisms and the digital evidence itself. This is achieved by moving the access control mechanism to the blockchain. This tracks any changes made through the access control mechanism, further ensuring transparency and integrity. This smart contract-based access control builds off role-based access control, allowing for more complex hierarchies to be used. This model aims to allow for both modularity, making adoption easier for existing digital forensic tools, and encouraging digital investigation and litigation to become more streamlined. Existing tools can easily integrate with the proposed model adding an extra layer of non-repudiation, transparency, and integrity.

Keywords: Blockchain, Chain of custody, Smart contract, Access control, Integrity, Digital forensics

1. Introduction

In the realm of digital data and information, the concepts of digital evidence and integrity stand as paramount pillars, ensuring the reliability and trustworthiness of electronic information in an ever-evolving digital landscape. Digital evidence encompasses data or information stored or transmitted in binary form, serving as a credible and admissible source of evidential information in a court of law (International Organization for Standardization, 2015). Examples of digital evidence include log files, text files, images, and emails. This digital data becomes difficult to preserve owing to no inherent measures to prove its legitimacy (Casey, 2002). A hash function is not sufficient alone as it can only show the file has not been modified since the calculation of the hash (Rivest, 1992).

A strict process is followed when a digital investigation is conducted and is referred to as the Digital Forensic Life Cycle (International Organization for Standardization, 2012). The steps taken include Acquisition, Identification, Collection, Report, and Preservation. During the acquisition and reporting processes, every interaction with a piece of evidence is documented such that the analysis findings can be repeatable and verifiable (Stoykova, 2023). The Chain of Custody (CoC) is a document that tracks all changes in any facet of the investigation. This may include the evidence itself or investigator logs generated by digital forensic software. The integrity of this log must be maintained in order for the evidence to be admissible in a court of law (Prayudi and Sn, 2015). If this log is altered or tampered with in any way, the entire investigation could be put in jeopardy (Sadiku, Shadare and Musa, 2017). The preservation process of digital evidence in regards to storing the chain of custody becomes extremely difficult (Giova, 2011; Prayudi and Sn, 2015). Monitoring the chain of custody during a digital investigation can be quite challenging when attempting to use the standard paper-based approach (Giova, 2011). Therefore, a more reliable and less error-prone approach is to have a digital-based chain of custody. However, this brings challenges regarding its integrity such as ensuring the digital data that was acquired during acquisition did not change during the investigation and was not maliciously modified. This is due to the fact digital data is much easier to modify and can be done remotely with minimal traceability (Cosic and Baca, 2010; Lone and Mir, 2019; Prayudi and Sn, 2015). Hence, a novel model becomes imperative for the safeguarding of chain of custody integrity. Implementing a digital CoC introduces its own unique set of challenges, as there will be a constant need for updates, making it infeasible to securely store a hash in a fixed location, as the hash naturally updates with the ongoing updates to the CoC throughout the investigative process.

Blockchain maintains CoC integrity through digitally signed transactions grouped into blocks, generating a hash. Changes invalidate the chain, causing a snowball effect (Nakamoto, 2008). Existing research (Bonomi, Casini and Ciccotelli, 2018; Burri, et al., 2020; Lone and Mir, 2019; Tian, et al., 2019; Yan, et al., 2020) utilized blockchain as a method for CoC immutability, however, it lacks built-in access control mechanisms as well as integration into other tools. This paper developed a model that allows for more intricate access control mechanisms with the integration of modular storage engines for digital evidence.

2. Related Work

Digital evidence management and access control are core aspects of any digital investigation and several models and strategies have been researched in the field (Chopade, et al., 2019; Lone and Mir, 2019; Singh, Ikuesan and Venter, 2022; 2019). It is evident that blockchain remains the best method for transparency and immutability, however, due to the intricacies of blockchain and the complex technical knowledge and proper use of the technology it becomes poorly adopted. The work by Bonomi, Casini and Ciccotelli (2018), proposed a blockchain-based chain of custody framework, utilizing a private or permissionless blockchain. The solution contains evidence logs, a front end, and a confidential evidence database distributed across trusted entities. One drawback of the solution is the computational overhead when accessing the CoC, as it requires all transactions of the blockchain to be queried. Moreover, a significant issue arises when considering access control, as each individual piece of evidence is solely linked to a single owner, which makes it extremely difficult to work as a team. One possible approach might include distributing the private key among various team members, however distributing a private key is highly risky and can easily result in the key falling into the wrong hands.

Burri, et al. (2020) propose a mixed-chain approach that uses both private and public chains, eliminating the need for a dedicated frontend. A trusted party manages the private chain, while a public chain is used for verification and transparency. This technique creates anchor points, allowing verification of blocks before they are uploaded to the public chain. However, this approach raises integrity concerns as it allows for modification of the private chain without affecting the public chain's immutability and verification.

Yan, et al. (2020) propose a shift from a singular owner model to a revocable ciphertext policy attribute-based encryption, allowing multiple people to collaborate on the same case. This method ensures blockchain integrity and non-repudiation, enhancing teamwork. However, this approach may expose the door for a 51% attack due to the use of attribute-based access control. A hybrid system with a group signature scheme and role-based access control could address this issue. This paper treats each piece of evidence as a transaction, not a block, so each transaction is chained. This type of system makes consensus procedures like mining difficult and computationally expensive because the nonce must be calculated per transaction, not per block.

Current studies have demonstrated the potential of blockchain technology in maintaining the integrity of the CoC. However, there exists a requirement for advanced access control methods that enable collaboration among investigative teams, as well as a faster way to build a CoC. There is also a need for more measures allowing for the user to interact directly with external sub-systems along with the blockchain such as integration with external storage systems.

3. IntegriStore

The proposed model ensures the integrity and non-repudiation of both the chain of custody along digital evidence. It is split into three (3) main components, namely Blockchain, Storage Engine, and a Web Interface as shown in Figure 1. The Blockchain provides the needed integrity, transparency, and non-repudiation, while also containing the information to build and maintain the CoC. The Storage Engine is responsible for the actual physical storage of the digital evidence. The Web interface provides a layer of abstraction for the user and allows them to interact with both the blockchain and the storage engine from a single interface.

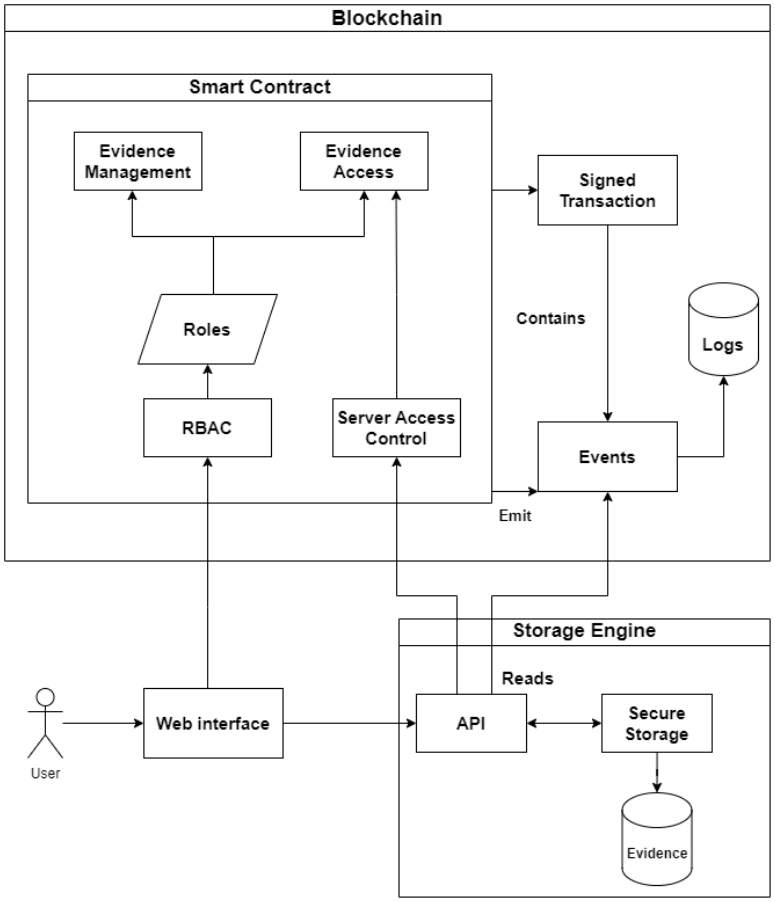

Figure 1: High-level model

3.1 Blockchain

The core of the model is the blockchain, ensuring the integrity of the evidence through cryptographically linked blocks. The consortium blockchain was chosen out of the 4 most common blockchain types (Private, Public, Consortium, and Hybrid (Arooj, Farooq and Umer, 2022)). A public chain uses distributed nodes across members of the public, private is hosted by a single entity, hybrid is a mixture of public and private, and lastly consortium deploys the nodes across trusted parties. Using a public chain is infeasible due to the monetary cost required to execute transactions on the blockchain. Finally, the system would be unsuitable because of the wait times for blocks to be mined on public chains (Doubloin, 2023). A private chain cannot be used since it would significantly reduce the integrity of the system. The hybrid chain suffers from the same cost aspect as the public chain, leaving the consortium blockchain as the best option.

In this case, trusted parties are government institutions such as courthouses and federal buildings. While distributing the chain can increase the risk of someone maliciously getting a copy, it would be difficult to parse as all information is in hash form with no context.

3.1.1 Smart contract

The smart contract handles the role-based access control, the evidence management, server access control and evidence access. Having the contract responsible for such a load places strain on the blockchain. However, this is done so that the chain of custody logging becomes inevitable. In contrast, regular systems can have instances where information and changes do not get logged. Every investigation should have a dedicated smart contract, thereby enabling distinct roles and evidence management for each investigation. This approach introduces a layer of encapsulation, where each investigation is encapsulated within its dedicated smart contract. Even after an investigation concludes and the associated evidence is removed from the storage engine, the smart contract perseveres as a lasting record. This then serves as a permanent history of that investigation which uses minimal storage since evidence is not stored on the chain.

3.1.2 Role based access control

The contract is built by extending the Open Zeppelin Role Based Access Control (RBAC) Scheme (OpenZeppelin, 2023), which inherits and adds more functionality to the access control scheme. In this scheme each role has an admin role, where a role cannot be the admin of itself. Multiple roles can, however, have the same admin, creating a tree-like structure as seen in Figure 2. The root structure of a system terminates with "0x0", and no user can hold this role. A root role is the root of their branch and can add other root roles, but they have no control over them. A user can hold multiple roles, but they cannot hold a role that is the admin of another role they have. A user can assign users to sub-roles but cannot be added to root roles. Any role can be deleted, and once removed, it cannot be added back. This is to ensure proper logging and data integrity. In IntegriStore, integrity and confidentiality can be maintained if a root role is compromised. The storage engine can ignore calls using the compromised contract, and a new contract with new root roles can be made. The integrity is still maintained implicitly, since the logs are immutable, and no one can delete or modify them. The evidence, therefore, is far more likely to be accepted in a court, as the trust in the CoC remains fully intact.

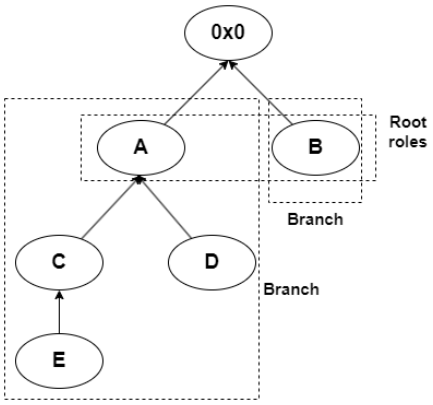

Figure 2: Role

3.1.3 Evidence management

For efficient evidence management, groups were implemented, allowing for the bundling of multiple pieces of evidence together. The smart contract keeps track of which roles have access to which groups. A user that holds a role that has access to a group, can upload an evidence hash to the smart contract. This will trigger an event that acts as the log for the chain of custody. An event tracks who added the evidence, the group it was added to, and the evidence itself in the form of a SHA256 hash. This is the only form of evidence stored on the chain. No other information about the evidence apart from its hash is stored. The contract is not designed to replace the storage/metadata of evidence, rather it is designed to assist the storage engine in adding a layer of immutability. A piece of evidence cannot be “removed” from the contract. If any changes need to be made or the wrong piece of evidence is added, then a new add call needs to be created which will result in a new event. The system is designed to be completely immutable, as any removal or change capability would hinder the credibility of the investigation and logs. A system of giving the user the ability to move evidence as well as remove evidence while logging these changes, may be possible, but is not explored in this paper.

3.1.4 Server access control

In instances where the external storage engine needs to make requests and transactions to the blockchain, the normal RBAC scheme will not work due to the storage engine having different actions from the user. To solve this, the contract keeps a list of addresses that are supported “servers”, i.e. addresses that are trusted servers that may perform server-related actions. This list can be updated, however, only root admins can add/remove these servers from the contract. One example of an operation that only a server can perform, is calling the *saveEvidence* function. This function indicates a particular piece of evidence has been saved to the database of a storage engine. The necessity for the storage engine to call the contract to provide a confirmation upon saving a piece of evidence, stems from the two-stage process of evidence storage. Initially, the hash is uploaded to the blockchain, and only once this is completed will the server accept and securely store the corresponding evidence. This deliberate two-stage design guarantees the comprehensive and accurate logging of all pieces of evidence within the chain of custody.

3.1.5 Evidence access

Since the actual evidence is stored off-chain in a storage engine, the investigator will need to interface with this storage engine to download and upload evidence. If the storage engine were to only use its internal access control mechanisms, it would defeat the entire purpose of the smart contract. In the case of downloading / uploading evidence, the smart contract determines if the user has access to a piece of evidence or the ability to upload it to a particular group. This gives rise to a minor challenge concerning how a user provides verification to the storage engine. To resolve this issue, the "personal_sign" feature of MetaMask is used (Consensys, 2023). To guarantee that the request of a user to store evidence is logged, the user must first make a call to the contract to request for evidence, after which the process as highlighted in Figure 3 is then carried out. The unique hash, generated for each request along with the *Evidence Access* event, prevents against any replay attacks.

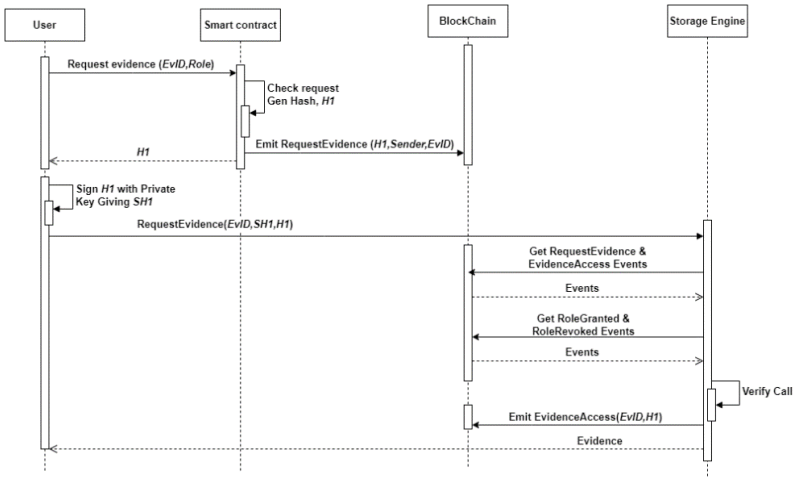

Figure 3: Sequence diagram for evidence access

3.1.6 Events

An event is an immutable log that is attached to a signed transaction when it is emitted from a smart contract. The event log (along with the transactions) implicitly becomes the CoC. Figure 4 shows an example of a decoded event on the blockchain, using Ganache (ConsenSys Software Inc, 2022). This event log is immutable as it is attached to the transaction. It is comprised of a signature (A in Figure 4.), and the parameters (C-D in Figure 4.). If the parameters are indexed, it means that the blockchain can be queried to only return events that match a particular filter. This could be used, for example, to get all pieces of evidence uploaded by a particular person. While log provision is one of the primary functions of these events, they serve another pivotal role. Every alteration made to the contract's state is associated with a prescribed event, allowing the event history to effectively encapsulate and describe the current system state. This allows the contract state to be calculated by replaying the events in chronological order. The API will use these events to verify a request from a user during the evidence access and upload stage as shown in Figure 3.

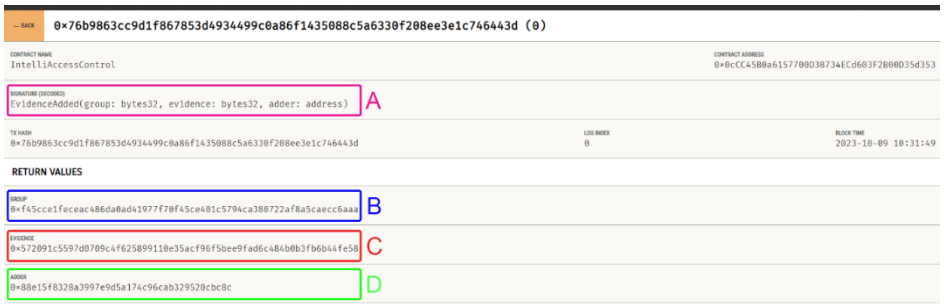

Figure 4: Example event on Ganache

3.1.7 Signed transaction

Each call to the contract is essentially a signed transaction that is sent to the blockchain. If the transaction is verified, it gets accepted and put into a block, making it secure and immutable. Since the user must sign this

transaction with their private key it means that each transaction in the blockchain can be traced back to an address allowing for the needed aspect of non-repudiation.

3.2 Web Interface

To improve adoption and assist the user in interacting with the smart contract, a web interface was utilized. A web interface removes the need of a user to install any application as well as improving device compatibility. The web interface is required to conceal the complexity and intricacies of dealing with the smart contract functions. It is responsible for abstracting the connection between blockchain and user, and user and API. Abstracting into a single interface, allows the user to focus solely on the investigation while the web interface manages the underlying complexity of simultaneously dealing with the 2 sub-systems (Blockchain and Storage Engine). It is crucial that the web interface allow the user to interact with the blockchain directly. The alternative would require the user to pass their private key along, which could negatively impact the safety of the system.

3.3 Storage Engine

The storage engine is responsible for the physical storage of evidence. Storing the evidence on a chain would be too costly. Storing the evidence in any highly distributed manner creates issues of privacy. Highly distributed storage works extremely well for its use case, however, when dealing with investigations where data is extremely sensitive and potentially voluminous, it becomes a breach of privacy. The storage engine, therefore, needs to resemble a highly secure centralized system with redundancy. The access control of this system is achieved through the blockchain, as described in Figure 3. The storage engine does not verify the user from a username and password. All verification is achieved through the private key of the user, the events, and the smart contract. To further improve security, more access control can be added by means of Multi-Factor Authentication. This paper does not define a particular storage engine. It is designed to be modular and integrate with any storage engine, provided the base API functionality to interface with the smart contract is present. Although trusted parties receive access to the blockchain, this does not imply that they also have authority over the storage engine. The blockchain may be hosted by one entity, and the storage engine by another. This makes it possible for a single blockchain to increase in efficiency and integrity, while supporting various storage engines installed or used by digital investigators. However, the storage engine also needs to have its own processes for evidence storage, as well as privacy preservation. For this existing secure storage models such as SecureRS (Singh, Ikuesan and Venter, 2022) and Digital Witness (Nieto, Roman and Lopez, 2016) can be used.

4. Prototype

The prototype was developed using Ganache, an Ethereum-based blockchain, to facilitate the development and deployment of smart contracts. The contract was developed using tools like Hardhat (Nomic Foundation, 2023) and Truffle (ConsenSys Software Inc, 2022), with a solidity-based contract. The web interface, API, and storage engine were built on a single Python flask (Ronacher, 2023) server, with the connection between the user and blockchain carried out client-side through Web3js (ChainSafe, 2023) and EthersJs. MetaMask was used as a wallet provider and for 'personal_sign' features. A straightforward interface was designed using HTML and Bootstrap to hide the complexities associated with both systems from the user. This paper uses the Flask server's local filesystem for file management, allowing direct access control through the API. Web3.py (Ethereum Foundation, 2023) is used for communication between the API and the blockchain. The API is streamlined, with a simplified database containing hashes and their corresponding keys. This approach enhances usability by storing role and group names as hashes on the blockchain, providing a more contextual understanding of data.

Since the storage engine is not the focus of this paper as it is designed to be interchangeable, the local filesystem of the Flask server was used. This allowed the file management to be done directly through the API with only one extra layer of abstraction, which was the access control of the API. The API itself is built into the flask server. Web3.py (Ethereum Foundation, 2023b) was used for the communication between the API and the blockchain. The API did not make any transactions itself, instead it uses Web3.py (Ethereum Foundation, 2023b) to get events from the blockchain. Web3.py (Ethereum Foundation, 2023b) provides the ability to query the blockchain for all events that contain a particular parameter or match a filter. For example, these events were used for the “Verify call” section of Figure 3. This design choice facilitated the creation of a highly streamlined API with a simplified database containing hashes and their corresponding keys (names). This approach greatly enhanced usability, as both the role and group names are stored solely as hashes on the blockchain. Consequently, the system can seamlessly replace the hash with the associated string, to provide users with a more contextual understanding of the data.

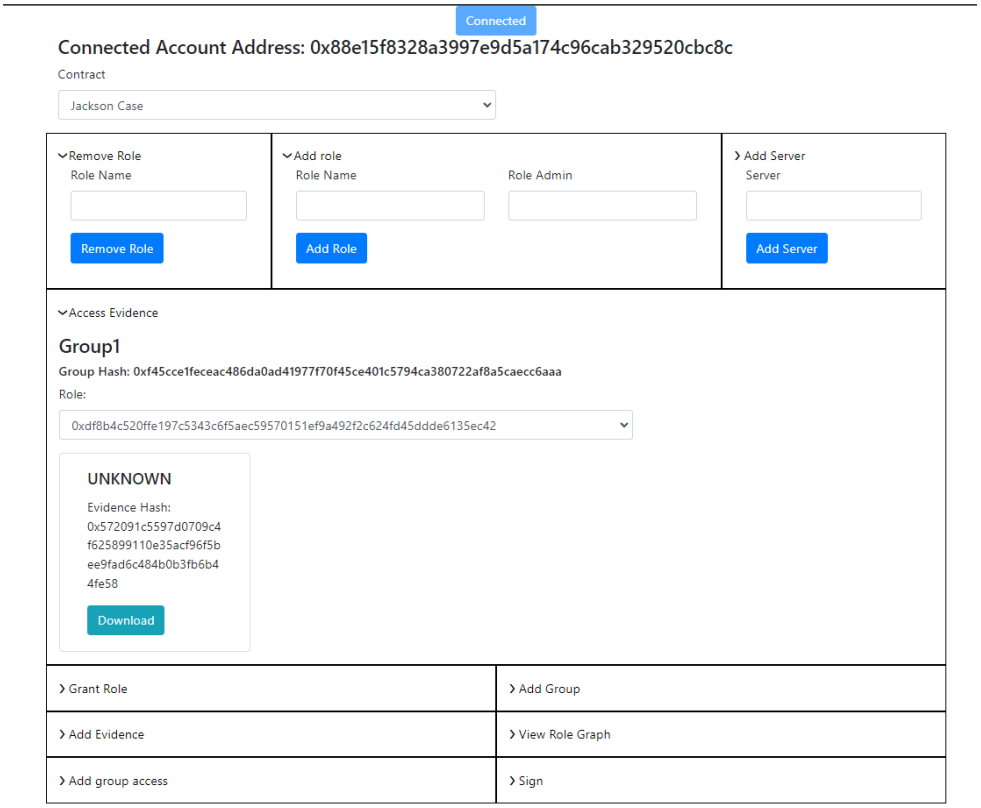

Figure 5: Web Interface

5. Discussion

To benchmark the proposed model and prototype, a comparative analysis was carried out comparing existing research. From Table 1, it is observed that IntegriStore is feature rich and is the only model that allows for role-based access control and contract access control. It is also suggested that the model is unique in allowing for the handling of access control directly through a smart contract, forcing it to be logged directly onto the CoC. IntegriStore is a model that allows for the integrity and non-repudiation of any form of digital data. Since the contract only keeps track of the hash and not the metadata, there is no limitation on the type of data. Any form of data that can produce a hash, may be tracked with the contract. This implies that the model has the capability to store and process more than just digital evidence. One example scenario involves using the model to control physical access control, wherein access cards employing a private key, are utilized to authenticate individuals. In this context, rather than downloading evidence, the focus lies on gaining entry to an evidence room. The model is built around its modularity and its ability to be attached to existing storage engines. Different digital forensic investigations and accompanying tools may differ in the way that they are conducted. The role of the digital forensic investigator and the tools necessary to manage the system, will evolve over time. IntegriStore is designed to evolve and adapt with necessity and change. The storage engine that is required to work with the system is considered external to the actual model, allowing it to be swapped out for whatever tool is needed by the user. It is also possible for the storage engine to be swapped out during an investigation. This would involve taking the current storage engine offline and moving the evidence to a new secure storage engine. After the move, the new storage engine should report the hashes of the evidence it received to the blockchain where it can be ensured no evidence was altered or removed. This allows for evidence integrity and preservation to be maintained when evidence is moved to a new storage engine. This is useful if a vulnerability is detected in the current storage engine as it can be swapped out while the CoC and Evidence Hashes remain secure on the blockchain.

Table 1: Comparative Analysis of IntegriStore

Feature	IntegriStore	(Bonomi et al., 2018)	(Yan et al., 2020)	(Burri et al., 2020)	(Santamaría et al., 2023)
Evidence Integrity	✓	✓	✓	✓	✓
Chain of custody Integrity	✓	✓	✓	✓	✓
Nonrepudiation	✓	✓	✓	✓	✓
Role-based access control	✓	X	X	X	X
Group evidence Ownership	✓	X	✓	✓	X
Contract Access control	✓	X	X	X	X
Multi Key signature	X	X	✓	X	X
Smart contract	✓	✓	X	X	✓
Storage Engine Integration	✓	X	X	✓	✓

6. Conclusion and Future Work

This study introduces an innovative model designed to ensure the integrity and preservation of digital evidence and the chain of custody, ultimately increasing the likelihood of digital evidence being deemed admissible in a court of law. The foundation of this model relies on blockchain and smart contract technology, where IntegriStore comes into play. IntegriStore incorporates smart contract-driven role-based access control, eliminating the constraint of a single owner per piece of evidence. This, in turn, fosters improved collaboration among team members during digital forensic investigations. Furthermore, by entrusting access control to the smart contract, the logging of the chain of custody becomes guaranteed. What is particularly advantageous is that the system can read the blockchain state directly from indexed events, allowing for seamless integration with external subsystems without burdening the blockchain with additional overhead. The web interface offers users direct interaction with both the blockchain and the storage engine, simplifying the overall experience. However, a notable limitation of this model pertains to the potential complexity and size of the smart contract, which may present challenges in terms of management and navigation. For future work, it is recommended that efforts be directed toward creating a more streamlined, smart contract structure. This would help optimize the management of the blockchain while maintaining the effectiveness of the model. A proof of concept has been developed to showcase IntegriStore's feasibility and its ability to preserve the chain of custody, providing a strong foundation for future refinements and expansion of the model. Future research could explore ways to make the model more scalable and adaptable to different use cases, potentially offering a valuable tool for the digital forensic investigators' community.

References

Arooj, A., Farooq, M.S., and Umer, T., 2022. Unfolding the blockchain era: Timeline, evolution, types and real-world applications. Journal of Network and Computer Applications. 207, 103511. https://doi.org/10.1016/j.jnca.2022.103511

Bonomi, S., Casini, M., and Ciccotelli, C., 2018. B-coc: A blockchain-based chain of custody for evidences management in digital forensics. ArXiv Prepr. ArXiv180710359. - [online] Available at: <https://drops.dagstuhl.de/storage/01oasics/oasics-vol071-tokenomics2019/OASIcs.Tokenomics.2019.12/OASIcs.Tokenomics.2019.12.pdf>

Burri, X., Casey, E., Bollé, T., and Jaquet-Chiffelle, D.-O., 2020. Chronological independently verifiable electronic chain of custody ledger using blockchain technology. Forensic Science International: Digital Investigation. 33, 300976. https://doi.org/10.1016/j.fsidi.2020.300976

Casey, E., 2002. Error, uncertainty and loss in digital evidence. International Journal of Digital Evidence . vol 1.

ChainSafe, 2023. web3.js - Ethereum JavaScript API — web3.js 1.0.0 documentation [online] Available at: <https://web3js.readthedocs.io/en/v1.10.0/> [Accessed 20 October 2023].

Chopade, M., Khan, S., Shaikh, U., and Pawar, R., 2019. Digital forensics: Maintaining chain of custody using blockchain. In:IEEE, *Third international conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC).* Palladam, India, pp. 744–747. https://doi.org/10.1109/I-SMAC47947.2019.9032693

Consensys, 2023. The crypto wallet for Defi, Web3 Dapps and NFTs | MetaMask [online] Available at: <https://metamask.io/> [Accessed 20 October 2023]10.20.23).

ConsenSys Software Inc, 2022. Ganache - Truffle Suite [online] Available at : <https://trufflesuite.com/ganache/> [Accessed 20 October 2023].

Cosic, J. and Baca, M., 2010. A framework to (im) prove" chain of custody" in digital investigation process. In: *Central European conference on information and intelligent systems.* Faculty of Organization and Informatics Varazdin, p. 435.

Doubloin, 2023. How long will an Ethereum (ETH) transaction stay pending? [online] Available at: <https://www.doubloin.com/learn/how-long-ethereum-eth-transaction-pending> [Accessed 18 October 2023].

Ethereum Foundation, 2023a. Contracts — Solidity 0.4.24 documentation [online] Available at: <https://docs.soliditylang.org/en/v0.4.24/contracts.html#events> [Accessed 19 October 2023]

Ethereum Foundation, 2023b. Web3.py [online] Available at: <https://web3py.readthedocs.io/en/stable/> [Accessed 20 October 2023].

Giova, G., 2011. Improving chain of custody in forensic investigation of electronic digital systems 11. International Journal of Computer Science and Network Security, VOL. 11 No. 1, January 2011

Lone, A.H. and Mir, R.N., 2019. Forensic-chain: Blockchain based digital forensics chain of custody with PoC in Hyperledger Composer. *Digital. Investigation.* 28, 44–55.

Nakamoto, S., 2008. Bitcoin: A peer-to-peer electronic cash system.- [online] - Available at: < https://bitcoin.org/bitcoin.pdf>

Nieto, A., Roman, R. and Lopez, J., 2016. Digital witness: Safeguarding digital evidence by using secure architectures in personal devices. *IEEE Networks.* 30, 34–41. https://doi.org/10.1109/MNET.2016.1600087NM

Nomic Foundation, 2023. Hardhat | Ethereum development environment for professionals. [online] Available at: <https://hardhat.org/> [Accessed 20 October 2023].

OpenZeppelin, 2023. Access Control - OpenZeppelin Docs. - [online] Available at :< https://docs.openzeppelin.com/>

Prayudi, Y. and Sn, A., 2015. Digital chain of custody: State of the art. International Journal of Computers and Applications114, 1–9. https://doi.org/10.5120/19971-1856

Rivest, R., 1992. The MD5 message-digest algorithm. - [online] - Available at: < https://www.rfc-editor.org/rfc/rfc1321>

Ronacher, A., 2023. Welcome to Flask — Flask Documentation (3.0.x) [online] Available at: <https://flask.palletsprojects.com/en/3.0.x/> [Accessed 20 October 2023].

Sadiku, M.N.O., Shadare, A.E. and Musa, S.M., 2017. Digital chain of custody. International Journal of Advanced Research In Computer Science and Software Engineering. 7, 117. https://doi.org/10.23956/ijarcsse.v7i7.109

Santamaría, P., Tobarra, L., Pastor-Vargas, R. and Robles-Gómez, A., 2023. Smart contracts for managing the chain-of-custody of digital evidence: A practical case of study. *Smart Cities* 6, 709–727. https://doi.org/10.3390/smartcities6020034

Singh, A., Ikuesan, A.R. and Venter, H.S., 2019. Digital forensic readiness framework for ransomware investigation. In: F. Breitinger and I. Baggili, eds. *Digital forensics and cyber crime, lecture notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*. Cham: Springer International Publishing. pp. 91–105. https://doi.org/10.1007/978-3-030-05487-8_5

Singh, A., Ikuesan, R.A. and Venter, H., 2022. Secure storage model for digital forensic readiness. *IEEE Access* 10, 19469–19480. https://doi.org/10.1109/ACCESS.2022.3151403

Stoykova, R., 2023. The right to a fair trial as a conceptual framework for digital evidence rules in criminal investigations. *Computer Law Secure* Review. 49, 105801. https://doi.org/10.1016/j.clsr.2023.105801

International Organization for Standardization, 2015. *ISO27043, Information technology — Security techniques — Incident investigation principles and processes.*

International Organization for Standardization, 2012. *ISO27037, Information technology — Security techniques — Guidelines for identification, collection, acquisition and preservation of digital evidence.*

Tian, Z., Li, M., Qiu, M., Sun, Y. and Su, S., 2019. Block-DEF: A secure digital evidence framework using blockchain. *Information Science*. 491, 151–165. https://doi.org/10.1016/j.ins.2019.04.011

Yan, W., Shen, J., Cao, Z. and Dong, X., 2020. Blockchain based digital evidence chain of custody. In: *Proceedings of the 2020 2nd international conference on blockchain technology*.Hilo HI, ACM. https://doi.org/10.1145/3390566.3391690