Securing, Standardizing, and Simplifying Electronic Health Record Audit Logs Through Permissioned Blockchain Technology

Jessie Anderson
Dartmouth College

Securing, standardizing, and simplifying electronic health record audit logs through permissioned blockchain technology

Jessie Anderson

Senior Honors Thesis

Department of Computer Science, Dartmouth College

Advisor: Sean Smith

Email: jandy145@gmail.com

Technical Report TR2018-854

June 1, 2018

Contents

Abstract

Audit logs perform critical functions in electronic health record (EHR) systems. They provide a chronological record of all operations performed in an EHR, allowing health care organizations to track EHR usage, hold system users accountable for their interactions with patient records, detect anomalous and potentially malicious behavior in the system, protect patient privacy, and develop insight into workflows and interactions among system users. However, several problems exist with the way that current state-of-the-art EHR technology handles audit data. Specifically, current systems complicate the collection and analysis of audit logs because they lack an interoperable audit log structure, spread audit log data from different EHR applications across multiple data repositories, and often fail to record all useful information about events in the EHR. Permissioned blockchain technology offers two opportunities to mitigate these issues. First, smart contracts running on the blockchain can impose an interoperable structure on audit log data, both within single health care organizations and across all organizations participating in the network. Second, the blockchain ledger constitutes a consolidated repository for all audit log data at each organization, simplifying the collection of data for analysis. AuditChain, the prototype system I present in this thesis, leverages Hyperleger Fabric's permissioned blockchain technology to address these issues of audit log interoperability, content, structure, and consolidation. Specifically, AuditChain uses the blockchain ledger and smart contracts to standardize audit log content, simplify access to audit log data, and ensure that audit logs contain all necessary and useful information.

Chapter 1

Introduction

Electronic health record (EHR) systems have gradually supplanted traditional paper-based health record systems in the United States, a shift driven both by the availability of new technology and by federal policy. The Office of the National Coordinator for Health Information Technology (ONC) oversees the Medicare EHR Incentive Program (also known as *meaningful use*), which provides incentives to health care providers that demonstrate meaningful use of EHR technology as defined by ONC [26]. As of 2016, over 95% of all hospitals eligible for federal incentives [24] and over 60% of office-based clinicians [25] showed meaningful use of EHR technology, demonstrating that the majority of all types of health care providers and the vast majority of large-scale health care organizations have integrated EHR technology into their workflows.

Audit logs serve multiple functional and regulatory purposes in EHR systems. They provide a chronological record of all operations performed on an EHR system, allowing for the reconstruction of past states of health records [32]. The documentation of system usage that audit logs provide holds clinicians accountable for handling patients' PHI in an ethical and legally compliant manner, and is sometimes used as legal evidence in medical malpractice cases [12, 33]. Audit logs clarify the origin of problematic or incorrect data in patient records [32], and can be manually or programmatically mined for information about common workflows and personnel interactions in health care organizations that use EHRs [16, 37].

Recognizing the importance and value of well-constructed audit log systems, the US government has released requirements detailing their content and availability in the *Health Insurance Portability and Accountability Act (HIPAA)*. EHR systems must comply with these requirements in order to qualify for meaningful use. These requirements are intended to ensure the usefulness and informativeness of audit log data in allowing health care organizations to successfully monitor system activity [33].

However, audit logging systems in many currently available EHR software systems are confusing, fragmented, and difficult to use . Some systems capture information in a misleading manner; others do not record all necessary or useful audit data [6, 15]. Audit logs even within a single organization may be fragmented across several different EHR applications and recorded in different ways in each of these applications [32]. Due to these drawbacks, tracking and resolving patient metadata across applications proves difficult and time-consuming, as does tracing user activity across each application [6, 12]. Poor-quality audit logs thus inhibit health care organizations from fully capitalizing on the capabilities that audit logs offer and inhibit the effective implementation of security procedures involving audit logs. *Interoperable*, or cross-application and inter-organization compatible, and consolidated audit logs would mitigate or resolve these issues [32].

Permissioned blockchain technology offers capabilities to facilitate the interoperability and consolidation of audit log data. *Smart contracts*, programs that run on the blockchain, provide a framework for standardizing audit log content and ensuring that the provider organizations participating in the blockchain network adhere to these standards. Chaincode defines a limited set of transactions and queries that users can perform on the blockchain ledger, managing and restricting data creation and access [9]. The timestamped hash-chaining functionality of the blockchain ledger, combined with data replication on nodes across the network, ensure that attackers cannot modify audit log data once it has been committed to the ledger [36]. Furthermore, a full copy of all data is available on each of the organizations' nodes, providing a consistent and validated view of the blockchain ledger for all participant organizations. Cryptographic identities maintained on the permissioned blockchain network link each transaction to the user who owns that cryptographic identity, maintaining user accountability [4].

To explore and evaluate the ways that permissioned blockchain technology can improve upon existing EHR systems, I built AuditChain, a full-stack application that puts audit log data on a permissioned blockchain comprised of multiple healthcare provider organizations. *Hyperledger Fabric* provides the underlying permissioned blockchain functionality, and custom chaincode manages audit log creation and access. The append-only nature of the ledger ensures that audit log data cannot be modified once committed to the ledger. Chaincode enforces both legal requirements and professional recommendations for patient data-related audit log records. A simple desktop application demonstrates how information on the ledger might be presented to users in a readable and useful format. AuditChain integrates local user authentication to link user-facing authentication credentials with Hyperledger Fabric's on-chain cryptographic identity management.

In sum, AuditChain evaluates the central question of this thesis: how might the properties and capabilities of permissioned blockchain technology, specifically Hyperledger Fabric, be used to improve upon current EHR audit logging technology? I propose that several specific aspects of permissioned blockchain and Hyperledger Fabric - namely, data immutability and non-repudiation, chaincode, ability to integrate with a user-friendly frontend via an API, and ability to integrate on-chain cryptographic identity with off-chain authentication mechanisms - provide unique capabilities that are well-situated to solve issues with audit logs in current EHR sytems.

This thesis presents AuditChain in the context of audit log usage and requirements, current problems with EHR audit logging systems, and prior work integrating blockchain technology with health information technology. It begins with background on audit logs and blockchain technology and presents prior work with blockchain in health care. Next, it presents the prototype implementation and evaluates how the prototype resolves the issues with EHR audit logging previously identified in the paper. Finally, it concludes with directions for future work and a summary overview.

Chapter 2

Background: Audit Logs

An audit trail in an EHR system ensures that actions taken in the system, such as the creation, deletion, viewing, and modification of patient records, can be traced to the user who performed them [13]. Audit logs, individual entries in the EHR's audit trail, connect EHR users to their interactions with health records in the same way that handwritten initials and date connect an entry on a paper chart with the physician who made that entry. Audit logs thus perform the same function in a digital context that manual attributions like dates and initials perform in paper contexts: an outside auditor can determine the origin of information on a patient's chart. In addition to information attribution, audit logs provide a record of who *viewed* a patient's record. Tracking this information ensures that no system user can view a patient's PHI "on the sly" without anyone else being able to find out, and provides information access surveillance resembling the manner in which staff members in a hospital or doctors' office using a paper system manually manage access to patient records and keep track of who looks at those records.

Audit logs therefore hold rich EHR system usage information that serves a variety of purposes in the health care domain, including security audits, system usage analysis, tuning of access controls, monitoring of user behavior, and evidence in legal cases. However, to provide full utility, they must be recorded in a particular manner, contain certain information, and be accessible and understandable. HIPAA specifies requirements detailing the content and availability of audit logs, and both government and non-government bodies have published further recommendations to enhance the utility of audit logs [15, 21–23, 33]. Current systems fulfill these requirements and recommendations to varying degrees, in many cases leaving room for improvement [15].

2.1 Audit Log Usage

Well-constructed audit logs effectively capture complex user activity in EHR systems and serve a variety of purposes in health care organizations. With the information contained in the audit trail, organizations can perform rigorous security audits, monitor system usage for problematic user behavior, and gain deeper insight into general data-access patterns and workflows within the organization [6, 16, 33, 37]. In legal cases, informative audit logs can present evidence regarding patient privacy breaches and provider decision-making [12, 19]. The following section presents specific use cases for audit logs.

2.1.1 Compliance control

Security officers at health care organizations can review audit logs to verify that system users access PHI in permissible and ethical ways. Access needs in a healthcare ecosystem are too complex to be fully captured by programmatic access controls on patient data; to compensate for this complexity while ensuring clinicians have access to all the information they need, health care organizations often provide users with more access than they need. Overly restrictive access controls can limit employees' ability to perform their jobs effectively if these limitations prevent them from accessing necessary information [33]. Audit logs can be used in post-hoc access control measures to ensure that clinicians do not abuse their access to patient data [32, 33].

2.1.2 Legal evidence

Full audit log records may be requested as evidence in court proceedings. For example, a hospital may be required to produce a full record of metadata (including an audit trail) for a patient in a malpractice case. Because audit logs can be used to reconstruct past states of a patient's record, they may support or contradict a defendant's claims about information they did or did not know at a particular point in a patient's care. Therefore, in some cases, audit log data can be used to support the validity of a defendant's medical decisions based on the information known at the time the decision was made; in other cases, it may provide evidence to support malpractice claims [12, 19].

2.1.3 Fine-tuning access controls

Analysis of audit logs can allow organizations to refine access controls within their EHR systems to more accurately reflect the access needs of clinicians in different roles. Hospitals commonly use *role based access control (RBAC)* to manage data access permissions. Under this method, each role is associated with certain access capabilities; users are assigned roles in the system based on the position they hold, which determines what actions they can perform and what data they can view in the EHR system [29]. However, in order to allow flexibility and ease of access to needed patient data, RBAC may in certain circumstances give clinicians more access than they need to patient data, which places PHI at unnecessary risk. Analysis of audit logs can supplement RBAC by providing insight into more fine-grained access patterns [16, 29, 37].

Røstad and Edsberg provide an example of how audit logs may be used to update access permissions in an EHR system [29]. They studied access logs in the EHR System DocuLive EPR, and found that users employed exception mechanisms to gain access to PHI too frequently for these accesses to be considered true exceptions to standard protocol. In examining the audit logs at organizations using DocuLive EPR, the authors found that users cited a common set of reasons for needed exceptional access to PHI. Audit log data thus indicated that users had recurring access needs that DocuLive EPR's access control configuration did not provide. The authors went on to make specific recommendations for changing access controls in DocuLiveEPR to better fit user needs, demonstrating how audit log analysis can be employed to make access control mechanisms in an EHR system more reflective of end-user usage requirements.

2.1.4 Data Mining

Finally, some researchers have mined audit log data to model workflows within health care organizations and to automate detection of anomalous user behavior. Zhang et al. present an algorithm that uses audit data to detect deviations from normal patient care workflows, which may signal unauthorized or inappropriate data access in an EHR system [37]. Malin et al. demonstrate a data mining process that uses audit logs to model user interactions in a health care encironment [16]. Automated data analysis can provide deeper insight into system usage and interaction patterns in health care organizations, which may help inform policy within these organizations. Furthermore, from a security standpoint, automated analysis of audit logs may allow EHRs to automatically detect potential malicious behaviors in the system.

Audit logs thus provide an array of use cases that depend on the usefulness of information contained in the logs. The next section expands on requirements and recommendations that seek to maximize the utility of audit logs in performing the functions outlined above.

2.2 Requirements and Recommendations

Both government and non-government bodies provide recommendations for audit log content that are informed by their use cases. Additionally, federal policies and incentive programs provide requirements that software and organizations must fulfill.

The Health Insurance Portability and Accountability act (HIPAA) requires health care organizations to implement and regularly use mechanisms to monitor the security of patients' PHI [21, 33]. Audit logs play a key role in monitoring security; consequently, the Office of the National Coordinator's (ONC's) Meaningful Use certification requirements include specifications for the content and use of audit logs [22, 23]. (EHR software certified via the meaningful use program qualifies for a federal incentives program that subsidizes providers for using certified technology, so vendors are incentivized to fulfill meaningful use requirements.) Content requirements include [23]:

- Date and time of logged event
- Patient ID whose record was accessed

- User ID who performed the logged event
- Action type (create, delete, change, query, print, or copy)
- Identification of patient whose data was accessed

In addition to these content requirements, ONC specifies that audit logs must be tamper-proof and that the technology used for auditing should be able to detect any unauthorized alteration that does occur [23].

The ONC expands on these requirements with further specifications detailing audit log reporting mechanisms. Specifically, vendors must provide functionality enabling users to create audit reports for a specific time period. These reports must be sortable on each of the required content fields [22].

Other recommendations for audit logs include [15]:

- Recording the method of data entry, particularly when the data is auto-filled or entered via copy/paste
- Recording the ID of the original author when information is entered on behalf of another user

Non-manual data entry in particular can lead to negative care outcomes through entry of misinformation and arbitrary inflation of a patient's record [34]. Therefore, tracking the data entry method in audit logs is vital to ensure the ability to identify the origin of issues introduced through non-manual data entry. Similarly, failing to record the original author of data entered into a patient's record on behalf of another user can have negative consequences for patients, particularly if the information entered is incorrect [8]. Though vendors do not need to record these details to acquire a meaningful use certification, they are aspects of system usage that EHR audit logs should record to enhance the utility of audit log data.

These certification requirements and additional recommendations are intended to ensure that vendor software enables health care organizations to meaningfully and effectively monitor EHR system usage and security.

2.3 Problems with Audit Logs in Current EHR Systems

Audit logs in currently-used EHR systems display some common problems, particularly when evaluated against the requirements and recommendations outlined in Section 2.2 and when considered in relation to the use cases described in Section 2.1. These problems can make audit log data difficult to use or insufficient in some use cases. This section explores specific pain points in greater detail.

2.3.1 Producing Patient Metadata

Full patient metadata, including audit log data, can be expensive, difficult, and time-consuming to assemble. In legal contexts requiring the production of patient metadata, experts are often required to produce and analyze this data [33].

2.3.2 Interoperability

The healthcare sector has not yet adopted an interoperable audit log standard. Interoperable audit logs would help support the secure exchange of health data among separate health care organizations, facilitating better and more effective care for patients. In addition, interoperability would allow health care organizations to track information across multiple systems to construct a unified view of data changes within an organization [32]. Without an interoperable standard, log data remains fragmented both within and across organizations, preventing effective analysis.

2.3.3 Audit Log Content

Audit logs do not reliably contain all required, recommended, or useful data. Boldt et al. found that many of the hospitals they studied recorded incomplete information about system activity, that the logs were difficult to interpret, and that inconsistencies sometimes occurred in the log data (e.g., a user was recorded performing an action after logging out of the system). Due to the poor quality of audit data, few people within the health care organizations studied even knew it existed, and even fewer people accessed or used the data. Therefore, few of the benefits of audit logs described in Section 2.1 could be realized. Though this study took place in the Portuguese health system, it underlines the observation that audit log data must be well-recorded and well-structured in order to be useful, and that poor audit data cannot support the use cases described in Section 2.1 [6].

Though all certified EHR systems in the US must implement baseline requirements for audit logs, as described in Section 2.2, many EHR systems fail to implement useful but non-required recommendations. For example, only 44% of US hospital audit logs track the method used to update patient records [15], even though tracking the data input method is critical in identifying the source of error in a record. For example, copying and pasting information in patient records can lead to record inflation, overdocumentation, and error propagation [34]. Another non-essential but recommended practice is recording the original author of record updates made on behalf of another user, as failing to do so can cause confusion and misattribution later on [8]. However, only 67% of hospital audit logs in the US record the original author of a record modification if the modification is attributed to someone else [15]. Thus, audit logs often leave out important and useful information, particularly if this information is not legally required.

2.3.4 Multiple EHR Applications

Health care organizations often use multiple EHR applications from different vendors that each record audit data in different ways [32]. For example, radiology data may be recorded and managed through one application, while pathology information and procedures are handled by another application. Differing audit log structures and fragmentation of audit data across applications can complicate the collection and analysis of audit log data in health care organizations that use many applications [6]. This lack of cohesion complicates the process of comprehensively tracking user behavior and detecting anomalous accesses within a single organization.

2.4 Audit Logs: Summary

In summary, audit logs constitute a vital aspect of EHR technology that enables security officers to monitor systems for privacy and security. Additionally, they provide a mechanism to improve access control models, to automatically detect anomalous and potentially malicious behavior within the system, and to gain insight into personnel interactions and workflows in hospitals. For audit logs to be usable in these applications, they must contain vital information, be understandable, and be well-formatted. To this end, the ONC has released both recommendations and requirements detailing the content and availability of audit logs. However, not all hospitals use software that implements these guidelines. In particular, EHR systems often fail to implement recommendations that are not required to qualify for incentive programs. In addition to lack of required information, audit logs currently lack an interoperable structure, can be difficult to consolidate for a single patient, and tend to be fragmented across data repositories in health care organizations that use multiple EHR applications. Each of these issues demonstrably hinders the effective use of audit logs. Improvements upon the existing state-of-the-art are therefore necessary to fully realize the functionality and insights that audit logs offer.

Chapter 3

Background: Blockchain

As the *Economist* succinctly states, albeit in a non-technical manner, the blockchain is "The great chain of being sure about things" [7, 30]. Since its invention in 2008 [20], the blockchain (also called *distributed ledger technology*) has been lauded for removing the need for a trusted third party in overseeing asset transactions due to its inherent properties of data nonrepudiation and immutability. More recently developed blockchain systems, such as Ethereum [35], have extended the functionality of a blockchain network to handle arbitrary computation in a network of distributed trust. Though the blockchain began as a method of supporting cryptocurrencies, researchers and businesses are increasingly exploring ways to apply blockchain technology to areas outside cryptocurrency and financial assets, focusing on a variety of domains such as IoT devices, health care, and supply chain tracking [3, 5, 30, 31].

The blockchain can be conceptualized as a state machine that runs on a network of computers, or *nodes*. The blockchain's state is stored in a *ledger*, which constitutes a full record of all *transactions*, or state transitions, that have ever occurred on the network.

Each peer on the network holds a full copy of the ledger, which the peer every time new transactions occur and are broadcast to the peer network. The ledger stores collections of transactions as *blocks*; in addition to transaction records, each block contains a timestamp of when that block was created as well as a cryptographic hash of the data contained in the previous block. Because every block $n+1$ contains a hash of the data in the previous block n, blocks are chained together in a chronological sequence, hence the name *blockchain*.

Many blockchain systems perform transactions via *smart contracts*, programs that are deployed to the peer network and that can perform computations and hold internal state. Different blockchain networks implement smart contracts in different ways. For example, Ethereum runs smart contracts in a special-purpose Turing-complete language that allows users to perform arbitrary computations in return for *gas*, Ethereum's native cryptocurrency [35]. Hyperledger Fabric, on the other hand, runs smart contracts in the general-purpose programming language Go [4]. For the output of running a smart contract to be accepted as a valid transaction, the network must reach consensus on the computational result of executing the smart contract and on the validity of that result.

Consensus protocols differ among different blockchain implementations. Bitcoin and Ethereum use a computationally expensive puzzle called *proof-of-work* to reach consensus. Nodes simultaneously work to solve the puzzle, and the node that solves it first creates a block with the puzzle output, a hash of the previous block, a timestamp, and the results of transactions in the network, broadcasts the block to the network, and receives *mining rewards* in the form of cryptocurrency. Nodes reach *consensus* by accepting the proof-of-work output as correct and verifying the validity of transaction output. (In Bitcoin, for example, transactions are only valid if they do not constitute double-spending and if the assets transferred in the transaction are in fact available.) Once consensus is reached, each peer adds the new block to its ledger [20, 35]. Transactions become immutable once they are appended to the ledger due to the computational infeasibility of re-mining all the blocks in the ledger to accommodate the modification [20, 35].

The blockchain's append-only distributed ledger thus supports data non-repudiation and immutability, and distributed consensus protocol removes the need for a trusted third party to oversee asset transfers. These aspects of the blockchain have compelled both researchers and enterprises to explore various applications for the new technology. The expansion of interest in blockchain has led to the development of permissioned and permission-less blockchain systems, two distinct flavors of distributed ledgers that lend themselves to different types of applications and that have differing technical and practical implications.

3.1 Permission-less vs. Permissioned Blockchain Systems

The fundamental difference between permissioned and permission-less blockchain systems lies in the underlying assumption about the relationships among participants in the network; this underlying assumption leads to various practical and technical differences that inform the use cases of each type of system [36]. Permission-less blockchains assume a trust-less model in which no node on the network knows any other node's identity, in which no node trusts any single other node, and in which any node is allowed to join the network at will without special access rights or permissions. More recent implementations of blockchain have developed a permissioned model, which assumes that all nodes in the network know the identity of all other nodes, even if mutual trust does not exist between any two nodes.

“Gateway authorities” control membership in a permissioned network, ensuring that the only nodes that can join the network are nodes whose identities are known to the other participants. These two differing trust models lend themselves to different use cases; the permission-less model is most often used for cryptocurrency exchange, while permissioned blockchains generally have business applications [20, 35, 36].

The differing trust models lead to architectural and implementation differences between permission-less and permissioned blockchain systems. Because nodes in a permission-less blockchain do not know or trust one another, the integrity and accuracy of the ledger relies on the computational power necessary to mine blocks. In a permission-less blockchain, nodes must perform a computationally expensive task in order to create a new block, which ensures that the data on the ledger will be correct as long as the majority of computational power is controlled by honest and non-colluding nodes. The computationally expensive task of creating a block is called *block mining*. During the mining process, nodes vote to determine which representation of the ledger is correct in a process called *consensus*. As a reward for block mining, the node creating the block accepted by the network is rewarded in some way, usually via the blockchain network’s native cryptocurrency. After block creation, the majority of nodes must verify that each transaction in the block constitutes a valid transition of the ledger’s state before the block is accepted and added to the ledger. Therefore, a permission-less blockchain network is open, trustless, and decentralized. Anyone is able to join the network; everyone is able to see all transactions; no trust is assumed in either a central authority or between any two nodes in the network; and network computing costs to protect against ledger modification by attackers [20, 35, 36].

By contrast, permissioned blockchain networks eliminate the need for computationally expensive mining processes through the assumption of mutual knowledge of identity among nodes in the network. Because all nodes know one another, and actions on the network are cryptographically linked to the nodes that performed them, nodes can be held legally accountable for actions taken on the network. The lack of computationally expensive mining removes the need for mining rewards; instead, a relatively small subset of nodes perform and validate transactions, and broadcast the results across the network. Permissioned blockchains are generally industry- and/or purpose-specific, and handle off-chain assets rather than on-chain cryptocurrencies [36].

Permissioned and permission-less blockchains thus handle data storage, transaction verification, block creation, and trust in different ways. These differing properties make them suitable for different applications. In general, permissioned blockchains tend to have industry-specific applications handling assets that are held off-chain, while permission-less blockchains handle on-chain asset transfers among unknown and untrusted participants.

3.2 Hyperledger Fabric: A Permissioned Blockchain System

Hyperledger is an open-source permissioned blockchain initiative under IBM and the Linux Foundation; *Hyperledger Fabric (HLF)*, a blockchain implementation within this project, employs a novel approach to consensus, smart contracts (called *chaincode* in HLF), and transaction validation. Consensus is customizable rather than built-in (Bitcoin and Ethereum's blockchains, by contrast, allow only one type of consensus that is built into the system). In addition, the method that Fabric uses to update the ledger allows for the parallelization to transaction processing, reducing computational overhead and latency [2].

Fabric separates the ledger update process into three steps [2].

- *Endorsement:* First, a client submits a transaction to *endorsing* peers. These are peer nodes on the network, each of which contains a full copy of the ledger. These peers execute chaincode to get the result of the transaction, sign the result, and pass this signed result back to the submitting client. No changes are made to the ledger at this stage.

- *Ordering:* The client then submits the transaction results to an *ordering service*, which takes care of block creation. The ordering service does not have a copy of the ledger, as this is not needed for ordering the results of transactions into blocks. After creating a new block, the ordering service broadcasts it to all peers on the network.

- *Validation:* Following the broadcast of new blocks, all peers *validate* the orderer's output, ensuring that the signatures of endorsing peers are valid, that the *endorsement protocol* [1] for the network has been fulfilled, and that the transactions contained in the blocks have not been invalidated. If the new block contains only valid transactions, peers accept it and append it to their local copy of the ledger.

HLF's purpose in separating ledger updates into these steps is to allow parallel rather than sequential transaction processing and to remove the need for every peer on the network to execute transactions [2]. This reduces computational overhead and transaction processing

[1] *Endorsement protocol* in HLF specifies the number and type of peers that need to provide signatures on a transaction result in order for that result to be validated later in the process. Endorsement is flexible and customizable on a network-by-network basis so that each network can implement the endorsement protocol that best fits its needs.

latency, and is appropriate in the context of a permissioned network in which all participants are known to one another.

Participants in a HLF network are called *organizations*, which are able to combine to form *channels* and *consortiums*. Each participant organization generally maintains a peer node that processes transaction requests and stores a copy of the blockchain ledger. In addition to the peer node, each organization usually owns a Certificate Authority that provides cryptographic identity management for that organization. Member organizations collectively manage an orderer node, which is responsible for ordering transactions that peer nodes have processed, forming blocks containing valid transactions, and broadcasting new blocks that peer nodes use to update their local copy of the ledger [4].

Ledger state is represented by a set of key-value pairs; the ledger contains the full history of the creation, value updates, and deletion of each key. The most recent values for each key are stored in a key-value store, an instance of which is maintained separately on each peer [11].

Peer nodes use chaincode to execute transactions in the phase of ledger updates. Chaincode comes in two flavors: system chaincode, which is deployed upon network creation, and custom chaincode that can be deployed to the network after startup [10]. Client applications can invoke both types of chaincode. Unlike smart contracts on other permissioned and permission-less systems, chaincode may be written in a general purpose programming language [2]; at the moment, HLF mainly supports chaincode written in Go.

Hyperledger Fabric is therefore a modular and customizable blockchain system for running distributed applications in a permissioned context. As a permissioned blockchain system with flexible architecture and a novel approach to consensus, HLF is a platform that is well-suited for a variety of business applications; one domain that has received particular attention is health care and health care information technology.

Chapter 4

Health Care and the Blockchain: Prior Work

Both the academic and corporate world have developed a variety of blockchain-based EHR solutions running on both permissioned and permission-less blockchain systems, ranging from proof-of-concept prototypes to production-level data storage and sharing mechanisms. Four trends emerge from current work on blockchain in the health care space. The first trend uses the blockchain to support a patient-centric health data management system. The second utilizes the blockchain to facilitate interoperability of health data among providers. The third trend leverages the immutability of transaction data on the blockchain to increase trust in the security, validity, and ownership of medical data. Research and enterprises following the fourth trend use the blockchain to allow patients to access their health data more quickly and easily. Several blockchain-based EHR solutions incorporate aspects of all four trends, using the blockchain to address multiple pain points simultaneously.

MedRec is a working prototype that seeks to achieve interoperability among providers and to grant patients more comprehensive and granular control over who is able to access their data [3]. MedRec runs on the Ethereum blockchain and models health records as assets; smart contracts control access to these assets. The prototype represents medical records on-chain as generic query strings that can be executed against a provider's database to get the full record; patients can invoke these smart contracts through a user interface to control who can see the records that belong to them, and which aspects of those records are viewable. Thus, MedRec leverages a permission-less blockchain for two main purposes: giving patients more control over their health data, and allowing data to be transferred more easily among different providers via interoperable query strings.

HealthChain models *personal health information (PHI)* as digital assets whose ownership may be transferred between patients and providers [1]. Unlike MedRec, these digital assets are stored on a *permissioned* blockchain. Patients are able to create and modify their own records in HealthChain, as well as perform transactions to transfer ownership of health records to a provider of their choosing. Like MedRec, HealthChain uses blockchain technology to give patients greater control over who can access their data.

Blockchain innovations in health care extend to the private sector as well. The Estonian health care system now operates on a blockchain infrastructure provided by the company Guardtime [28]; Gem, a US startup, is using blockchain technology to increase data access for health care providers and improve cross-organization interoperability [18]; Patientory leverages the Ethereum blockchain to give patients immediate access to their data, control over sharing their data, and real-time claim adjudication [17]. These enterprises constitute just a few representative examples within a burgeoning ecosystem of healthcare-related blockchain initiatives and products.

Each of the examples presented above use blockchain technology to address concerns with the interoperability of PHI, the availability of PHI to patients, the integrity of health data, and patient control over access to PHI. However, none of them leverage blockchain technology to target problems with EHR audit logs specifically. I argue that a permissioned blockchain is well-suited to handle the problems with audit logs described in Section 2.3.

AuditChain, the prototype presented here, adds to existing work on blockchain and health IT by demonstrating how a permissioned blockchain can be used to improve upon the way that current EHR systems handle audit logging.

Chapter 5

AuditChain Prototype

I built AuditChain to answer the core question of this thesis: how can permissioned blockchain technology, specifically Hyperledger Fabric, solve current problems with audit logs in EHR systems? Specifically, how well does Hyperledger Fabric's implementation of permissioned blockchain technology solve the problems outlined in Section 2.3?

Several of the properties and features of permissioned blockchain technology, and specifically Hyperleger Fabric's implementation of these features, present a viable means to solve the problems specified in Section 2.3. Chaincode handling the creation and retrieval of audit log data would impose a standard structure on this data, achieving interoperability for participating provider organizations. This standardized audit log structure would be customizable to fulfill both legal requirements and best-practice recommendations. Data from all applications would be contained on a single blockchain, simplifying and streamlining audit log reporting and analysis and allowing administrators to implement audit procedures more frequently and effectively. Finally, because all health care organizations participating in the blockchain network would know one another's identity, the permissioned blockchain offers a viable solution that is more scalable than the permission-less alternative.

AuditChain is a full-stack application that leverages a permissioned blockchain to secure, standardize, and simplify EHR audit logs. The base of the stack is a blockchain network of peers running Hyperledger Fabric. These peers run custom *audit log chaincode* written in Go that formats audit log data, returns audit log data to clients in response to queries, and adds new audit log entries to the ledger. A NodeJS backend application communicates with the blockchain network via a set of APIs provided by Hyperledger Fabric. Frontend clients can communicate with the NodeJS server through API endpoints that handle updating and querying ledger data. To demonstrate querying functionality, I built a desktop application that uses the querying API endpoints to fetch and display audit log data to end users. In total, the project contains approximately 4.9K lines of code.

Each component of AuditChain fulfills a specific purpose that relates to the issues described in Section 2.3. Storing data on the blockchain consolidates audit records in one location on each peer, meaning that audit metadata for a single patient can be retrieved with a simple query. Chaincode facilitates interoperability by imposing the same structure on all audit log data from each participant organization. The NodeJS server abstracts communication with the blockchain, consolidating this logic within one backend application to avoid fragmenting communication across multiple frontend applications. Finally, the desktop application I built demonstrates how audit log data would travel from from the blockchain to the end user, and how users can interact meaningfully with the blockchain network to retrieve information. All components of AuditChain work together to address my core aim: to use permissioned blockchain technology to solve issues with state-of-the-art audit logging systems.

5.1 Architecture Overview

5.1.1 Multi-Organization Architecture

AuditChain runs a permissioned blockchain network of health care provider organizations such as hospitals, urgent care clinics, and doctors' offices. Each participant organization owns a peer node and a certificate authority; all nodes share an ordering service. As described in Section 3.2, each peer node holds a full copy of the ledger containing audit log data. Certificate authorities handle new user registration and enrollment, allowing new end users to register on the network at each participant organization.

Each participant organization runs a client that sends transaction proposals, endorsed transactions, and queries to its peer. Transactions consist of audit log data; queries consist of parameters specifying the properties of audit log data being requested by the end user. Currently, AuditChain's endorsement policy specifies that only one peer from one organization needs to endorse the proposed transaction. Therefore, only the organization's peer needs to execute the transaction in order for validating peers to accept the transaction later in the block creation process.

If the participant organization's client sends a transaction proposal to the peer to add new audit log data to the ledger, the peer executes the audit log chaincode to get the transaction result and proposed update to the ledger. If an error occurs, the proposal is rejected. If the transaction is successful, the peer sends the signed transaction back to the client, which then sends the transaction to the ordering service. The ordering service places the transaction in a new block, which it broadcasts to all peers on the network. If enough peers validate the new block to fill the network's endorsement policy, the each peer appends the block to its copy of the ledger, permanently committing the new audit log entry.

If the client simply sends a query to the peer, the transaction proposal, endorsement, ordering, and validation process is no longer necessary; the peer can just execute this query against its copy of the ledger and return the results to the client.

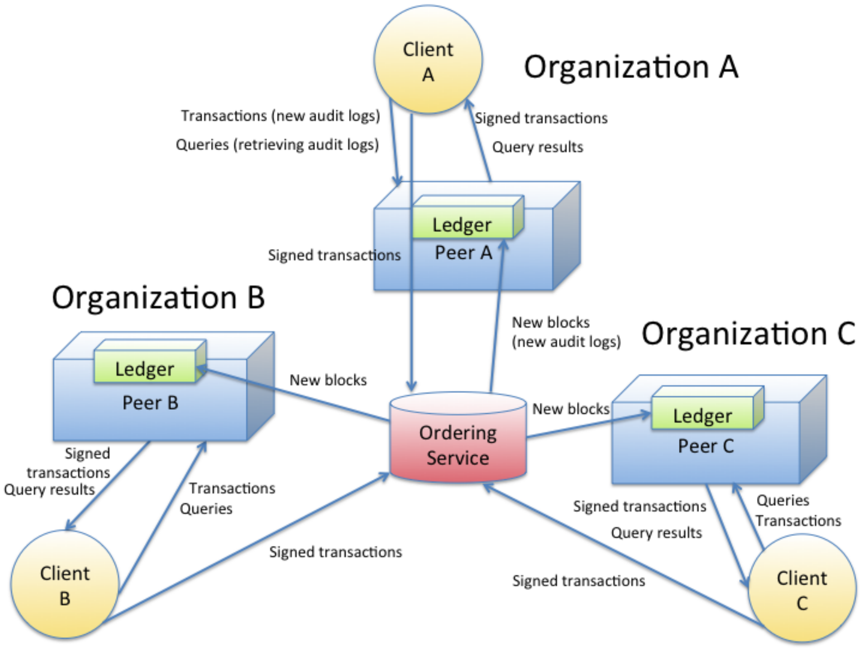


Figure 5.1: Multi-Organization Architecture of AuditChain. Displays how the client application at each participant health care organization communicates with its peer, how each peer executes chaincode and responds to each client, how each client communicates with the network's ordering service, and how the ordering service communicates with the peers to update the ledger on each peer.

5.1.2 Architecture Within Participant Organizations

At the participant organization level, AuditChain consists of four main components. The organization's peer communicates with the blockchain network as described in the previous section. A NodeJS server running at the organization acts as both a client to the organization's peer node and as a server for EHR applications within the organization: EHR applications make a call to the server to create new audit log entries or to query log data, and the server in turn makes a call to chaincode on the peer to perform a transaction committing the new entries on the ledger or to run a query on the ledger. When an application queries audit log data, the server sends the result it receives from the peer back to the requesting application. In addition to acting as a proxy between frontend applications and the blockchain for queries and transactions, the NodeJS server handles end-user authentication to the blockchain network. A MongoDB database stores local user credentials end users authenticate with; the database links these local authentication credentials to the user's cryptographic identity on the blockchain network. Finally, a desktop application provides a user interface that handles authentication, queries and displays audit log data, and supports new user enrollment and registration.

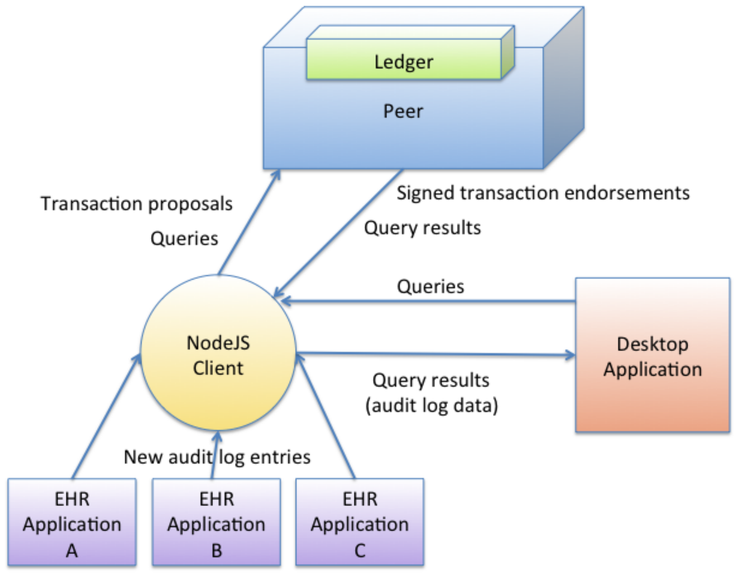


Figure 5.2: Organization-Specific Architecture of AuditChain; displays communication among all components in the system. The EHR applications at the health care organization (not part of AuditChain) use AuditChain's API to send new audit log data to the client, which then invokes chaincode on the peer to perform a transaction updating the ledger with the new audit log entries. The desktop application communicates with the client, which in turn queries the peer to retrieve audit log data to send back to the desktop application.

5.2 Chaincode

In AuditChain, chaincode enforces an interoperable structure for audit log data both within and across health care organizations, addressing the problems I identify in Sections 2.3.3, 2.3.2, and 2.3.4. Audit log structure is programmed into the chaincode, ensuring that all useful and necessary data is recorded in each log. Furthermore, the chaincode records audit log data in the same way for all applications running within a single participant organization, enabling organizations to draw insight from audit log data across each of its EHR applications. Finally, chaincode supports interoperability among all organizations running on the same network, ensuring that each participant health care organization will record audit data in the same way.

AuditChain's chaincode handles the creation and querying of audit log data. Chaincode state represents the most recent audit log entry for each record, for each patient, and for each user. New audit log entries are formed every time the state for a key referencing a particular record, user, or patient is updated.

The audit log chaincode provides an entry point for the client application to update and access the blockchain ledger, standardizing the way that audit logs are recorded across the network of participating organizations. The content of audit log entries presented here is informed by the requirements and best practice recommendations laid out in Section 2.2; standardizing this content via chaincode programmatically ensures that participant organizations comply with these standards and practices. The content of audit logs could be updated in the chaincode. Chaincode thus constitutes the basis of AuditChain's solution to the issues I identitied with interoperability, data consolidation, and audit log content, and is a customizable and flexible way to impose agreed-upon standards on the network of participant organizations.

5.2.1 Creating audit log entries

The audit log chaincode expects the following information to be sent with each transaction request that creates a new audit log entry:

- *Action Type (actionType):* The Action Type of an audit log is one of the following: edit, delete, create, or view. This information is passed from the client, and is determined by the EHR system of the organization that the user is a member of.
- *User ID (userId):* The ID of the user performing the action recorded in the audit log. This is the ID of the user in the local database holding the authentication credentials.
- *Patient ID (patientId):* The ID of the patient whose record is referenced by the record ID of the audit log. As with the User ID, this ID corresponds to the patient's ID in the local database where authentication credentials are stored.
- *Record ID (recordId):* The ID of the health record that is being edited, deleted, created, or viewed.
- *Data Type (dataType):* The type of the data that the audit log pertains to, e.g. chart, prescription, diagnosis, etc. This information is provided by the EHR system of the user's organization.
- *Original Author ID (originalAuthorId):* This field pertains to audit logs originating from software with "make-me-the-author" functionality, in which one user may enter information on behalf of another.
- *Data Field (dataField):* The field on the record that was modified, e.g. height, weight, etc.

- *Data (data):* The value of the data entered, e.g. numeric value of height in inches, etc. It is assumed that the EHR will send an obfuscated version of this data, either encrypted or hashed, so that the plaintext information does not get stored on the ledger.

- *Entry Method (entryMethod):* This is the method that the user employed to enter the data into the record. In software systems, many such methods exist, including copying-and-pasting, automated templating, manual entry, and others. Some methods, including templating and copy/pasting, can lead to problems such as data inaccuracy, incorrect recording of events during a patient encounter, and overly long records containing redundant information. Recording the user's data entry method can help clarify the source of such problems.

- *User's National Provider Identifier (NPI) (userNpi):* RTI International recommends that this piece of information be recorded with the user's actions in the system [15].

- *Original Author NPI (originalAuthorNpi):* Useful in the case of make-me-the-author, as explained under *Original Author ID*.

- *Organization NPI (organizationNpi):* Also recommended by RTI International [15].

- *Time (time):* The time that the event occurred.

The audit log chaincode makes three separate updates to the state of the ledger for each entry, indexing the data in three different ways to allow for easy retrieval from queries later:

1. Updates the key *recordId:*{*recordId*} with the remainder of the values in a JSON-like object

2. Updates the key *patientId:*{*patientId*} with the remainder of the values in a JSON-like object

3. Updates the key *userId:*{*userId*} with the remainder of the values in a JSON-like object

5.2.2 Querying audit log data

The audit log chaincode can query the ledger and filter audit log data in a variety of ways to obtain audit log information that has been committed to the ledger. The following queries are possible:

1. *getAllLogsForTimeRange:* Retrieve all audit logs created within a specified time frame

2. *getAllLogsForUserForTimeRange:* Retrieve all audit logs for a specified user within a specified time frame

3. *getAllLogsForPatientForTimeRange:* Retrieve all audit logs for a specified patient within a specified time frame

4. *getAllLogsForRecordForTimeRange:* Retrieve all audit logs for a specified record within a specified time frame

5. *getAllLogsForQueryForTimeRange:* Retrieve all audit logs for specified users, patients, and records within a specified time frame

These particular queries are representative examples of the type of data retrieval that is possible in AuditChain, and map directly to use cases in filtering and displaying audit log entries on AuditChain's desktop application. New queries returning information in different ways could easily be introduced in a new version of the audit log chaincode.

5.3 Client Application

The client application acts as a proxy between frontend EHR applications and chaincode, managing end user authentication to the blockchain network and supplying audit log creation and querying functionality to frontend applications. This means that frontend applications do not need to invoke chaincode directly, and that all logic for communicating with the blockchain is contained in one server rather than being fragmented across multiple frontend applications. The client application is therefore a necessary layer in AuditChain's architecture that connects end-user applications with the blockchain and chaincode functionality.

The client application sends transaction and query requests to the blockchain ledger via the HLF API on behalf of an authenticated user. In AuditChain, the client application layer is implemented as a NodeJS server with endpoints that correspond to chaincode functionality. This client handles end-user authentication via a local MongoDB database holding user credentials and information about each user's on-chain cryptographic identity. Endpoints that communicate with the blockchain are protected and available to authenticated users only.

5.3.1 User Authentication

The client application's user authentication system allows end-users to access their on-chain cryptographic identities in order to gain authorization to perform queries and transactions on the network. The local MongoDB database stores user authentication credentials and identifying information. The database links user credentials to the corresponding Hyperledger Fabric username, which in turn links the user to their certificate and keypair that establishes their identity on the blockchain network. This certificate and keypair is stored on the client application's filesystem. Upon successful authentication, the user receives a JSON Web Token (JWT) signed by the client application. This JWT serves as a key that provides authenticated users access to protected endpoints on the client application. All endpoints that perform transactions and queries on a user's behalf are protected; thus, in order to gain access to blockchain functionality, a user must successfully authenticate and receive a JWT from the NodeJS client.

5.3.2 Role-restricted functionality

Some endpoints may only be utilized by users with a certain role, ensuring that only users with particular permissions are able to perform sensitive actions on the system. For example, some endpoints are restricted to only admins (see Section 5.4). Endpoints that make transaction requests to put creation, deletion, and modification events on the blockchain are restricted to only clinicians and admins.

AuditChain would need to support a richer identity management strategy than the current implementation to realize a properly secured client application for a real healthcare ecosystem. However, this simplified model of role-based access provides an example of how access control mechanisms could be implemented on the client application to meaningfully restrict user access to chaincode functionality.

Potentially many frontend applications at each participating health care organization would need access to the blockchain ledger and to chaincode functionality. Implementing the client application as a server with defined endpoints allows these potentially many frontend apps to have one common access point to the audit log chaincode, a common authentication mechanism, and a common method for recording audit log data, addressing the issue described in Section 2.3.4. The NodeJS client consolidates all communication with the blockchain into one application, rather than communication being fragmented across multiple frontend applications.

5.4 Identity Management

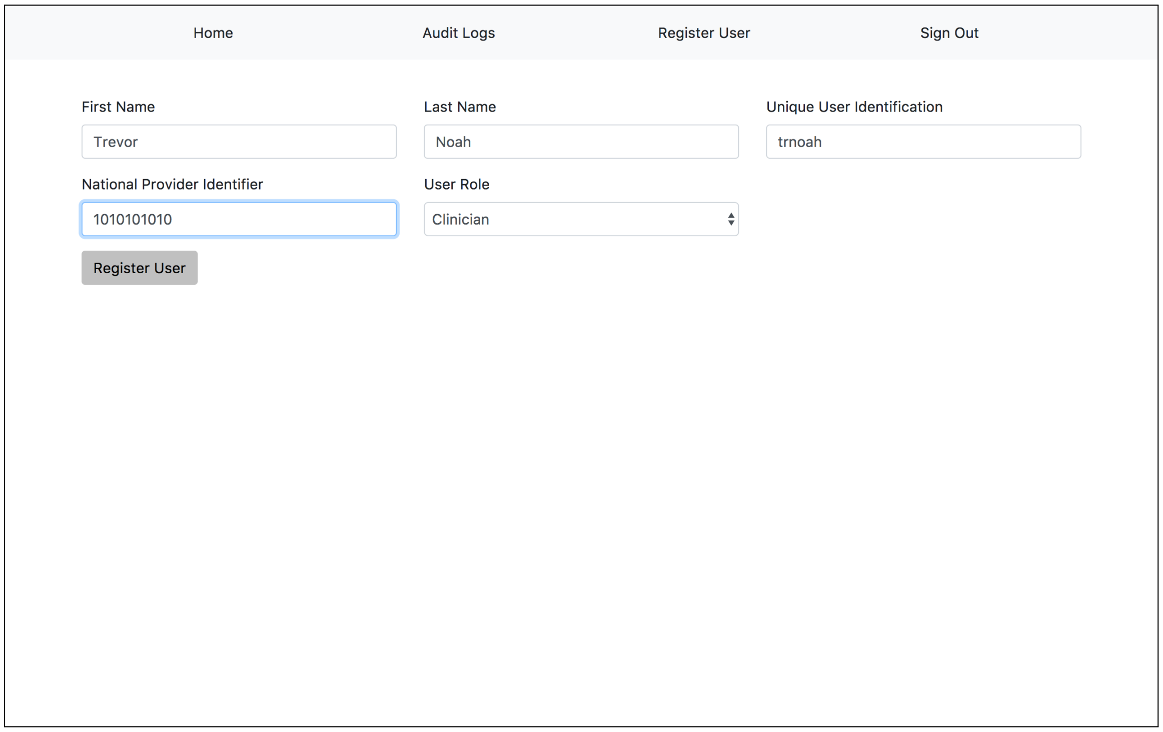
Identity management in AuditChain links end-users to their cryptographic identities on-chain in order to maintain user accountability. Each end-user owns a certificate and keypair that links them to their actions on the network. However, a certificate and keypair do not constitute a user-friendly authentication mechanism, particularly when they may be logging on from multiple machines that they do not own. To bridge this usability gap, AuditChain establishes a system by which a username and password link users to their cryptographic identities on-chain.

AuditChain accomplishes authentication and on-chain identity provenance through a two-part strategy. Users authenticate to the client application using credentials (a username and password) that they choose when they enroll in the system. Upon successful authentication, the client application signs every transaction and query that the authenticated user performs with the certificate and keypair associated with that user's identity on the blockchain. (Associations between authentication credentials and certificate/keypair are stored in the MongoDB database as explained in Section 5.3.1.) The user's keypair resides on the machine running the Node client application that communicates with the organization's peer node. This two-part strategy ensures that each cryptographic user identity in Hyperledger Fabric maps directly to a human end user.

User identities are created through a two-step registration and enrollment process.

5.4.1 User Registration

The client makes a call to the certificate authority to register a new cryptographic identity. This request is signed by the network's bootstrap admin user (which is itself a cryptographic identity comprised of a keypair and certificate stored on the client), and consists of an enrollment ID, affiliation, and user role. Upon successful completion of the request, the certificate authority returns an enrollment secret. The enrollment ID and secret serve as a temporary username and password for the new user in the local database, and the enrollment ID is linked to the new user in the database.



First Name: Trevor

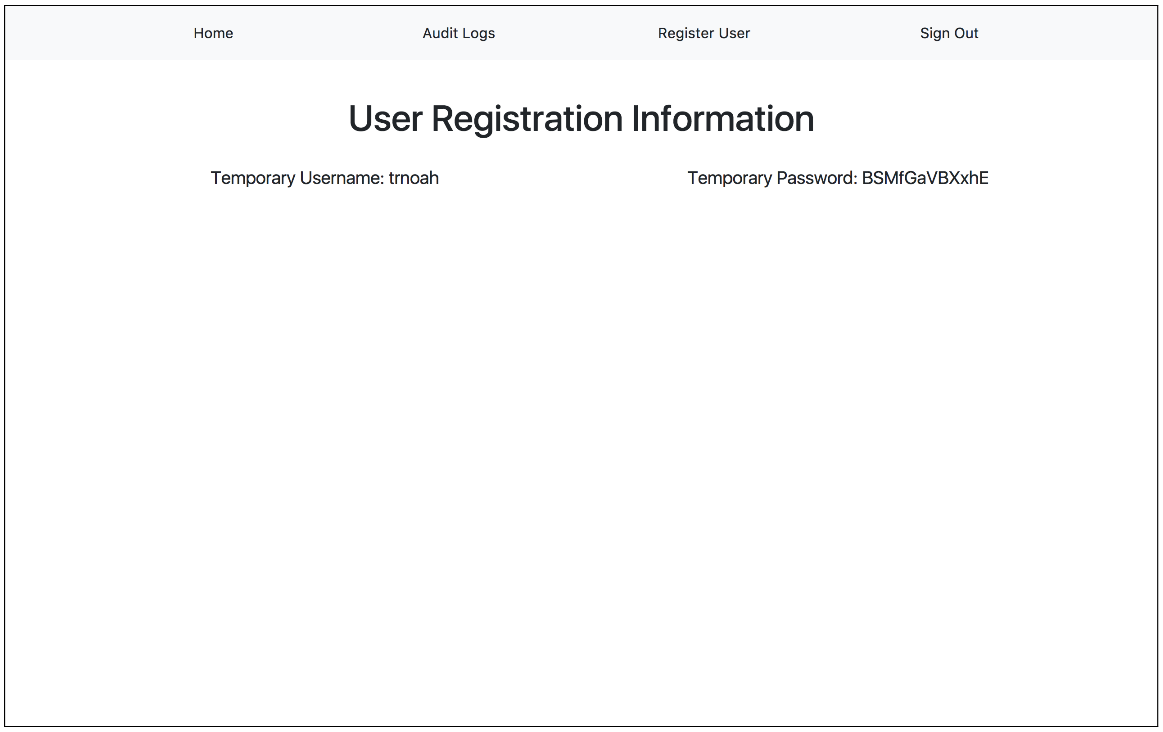
Last Name: Noah

Unique User Identification: trnoah

National Provider Identifier: 1010101010

User Role: Clinician

Figure 5.3: The interface that an administrator uses to register a new user on AuditChain



User Registration Information

Temporary Username: trnoah

Temporary Password: BSMfGaVBXxhE

Figure 5.4: HLF enrollment ID (”temporary username”) and HLF enrollment secret (”temporary password”) that administrator gives to new user to enroll with

The NodeJS client only allows authenticated admin users to register new user identities. Once the certificate authority returns the enrollment secret for a particular enrollment ID, the admin user is responsible for communicating the enrollment ID and secret to the new end user, who will use them to create new local authentication credentials and to enroll as a user on the blockchain network to activate their certificate and keypair.

5.4.2 User Enrollment

Once the new end-user receives their enrollment ID and secret from the admin who performed the registration, she can use these credentials to authenticate to the client and enroll in the blockchain network via her organization's certificate authority. Upon successful enrollment in the blockchain network, the Node client stores her certificate and keypair in their proper directories. She is also prompted to create a new username and password, and the MongoDB database is updated accordingly.

Through this identity management system, each end user and that user's credentials are linked to their cryptographic identity on the blockchain. The user does not need to know about or manage their certificate and keypair, as these are stored on and administered by the HLF client. Authentication and identity are handled through a familiar account creation and login flow. All interactions that an authenticated user has with the blockchain network are linked to her cryptographic identity from the time of authentication; users are unable to access the blockchain network without authenticating to the client application.

AuditChain thus manages identity with via two connected mechanisms. The first mechanism is local authentication: end-users prove their identity with a username and password. They use this username and password to gain access to the second piece of their AuditChain identity, which is the certificate and keypair that they sign requests to the blockchain with.

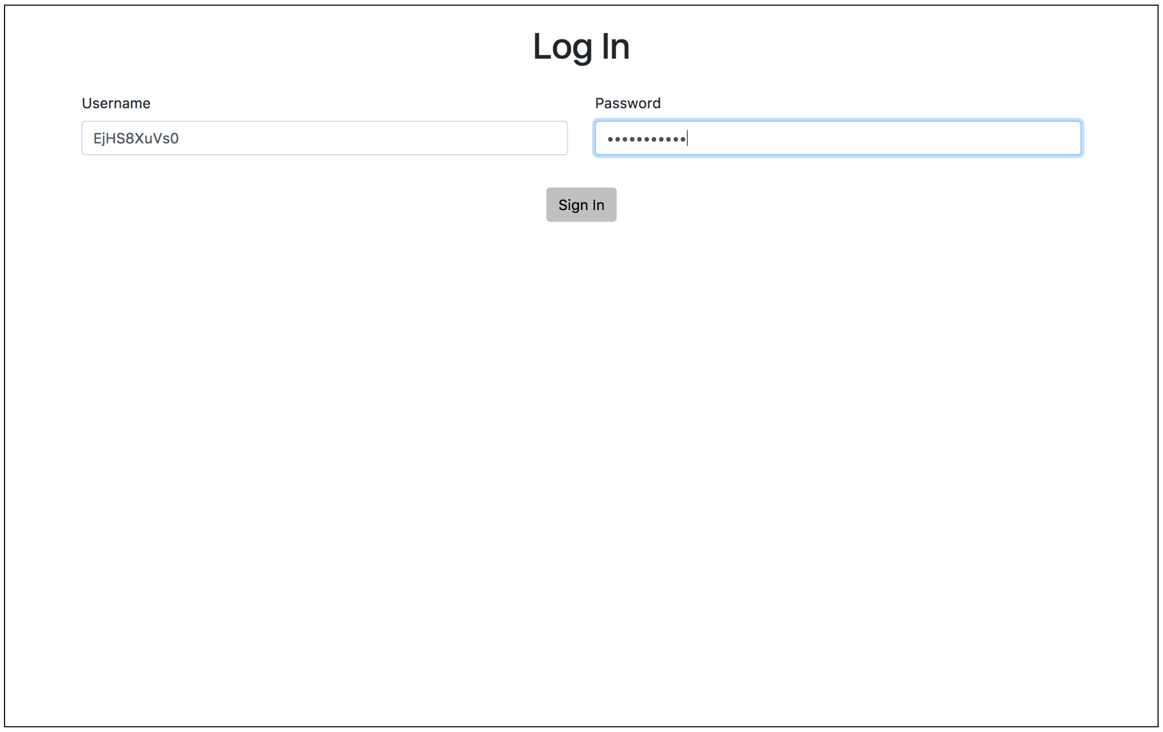
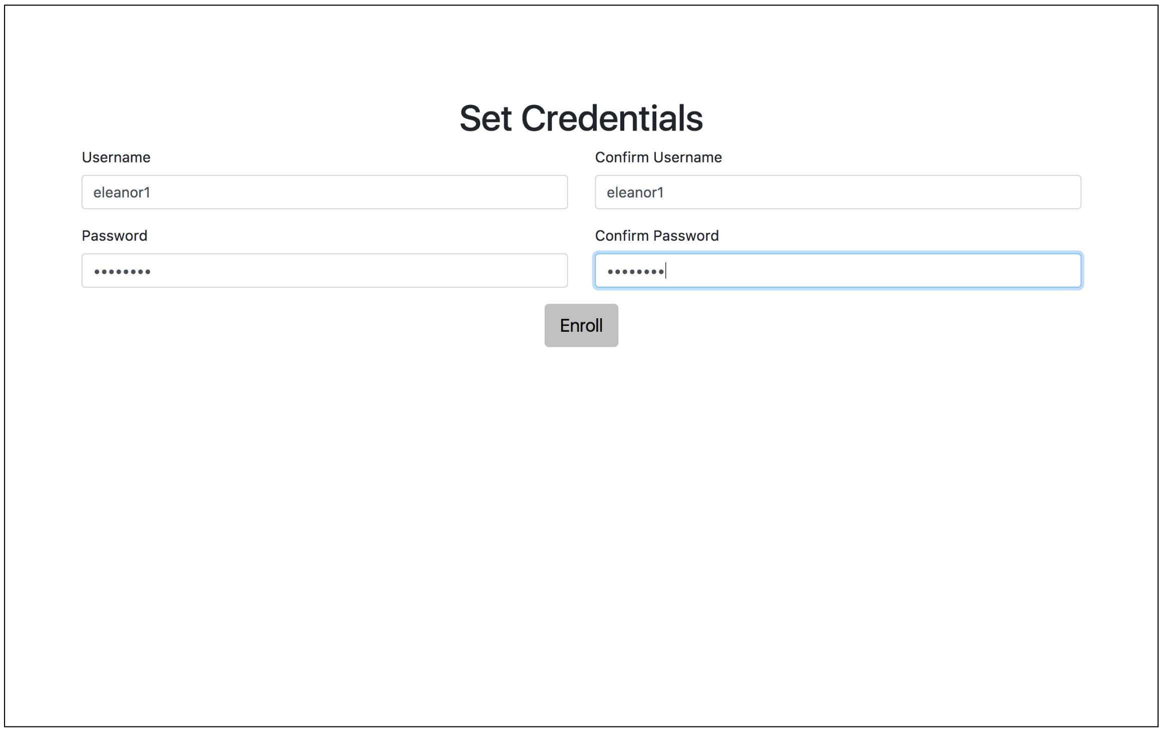


Figure 5.5: When a user logs into AuditChain's interface for the first time, they provide the HLF enrollment ID and enrollment secret given to them by the administrator who registered them


Set Credentials

Username	Confirm Username
eleanor1	eleanor1
Password	Confirm Password
••••••••	••••••••

Figure 5.6: The first time a new user logs in, she provides a new username and password of her choosing

Chapter 6

Results

AuditChain leverages the features and capabilities of Hyperledger Fabric's permissioned blockchain system to improve upon existing audit log software. As outlined in Section 2.3, many current audit logging systems have issues that permissioned blockchain technology is well-suited to resolve. The following section recaps situations in which current systems tend to fall short, explains how AuditChain offers improved functionality in each of these situations, provides example usage scenarios demonstrating improvements, and finally analyzes the limitations of AuditChain's current implementation in addressing each situation.

6.1 Producing Patient Metadata

6.1.1 Current Problem

Full patient metadata, including audit log data, can be expensive, difficult, and time-consuming to assemble. In legal contexts requiring the production of patient metadata, experts are often required to produce and analyze this data.

6.1.2 AuditChain solution

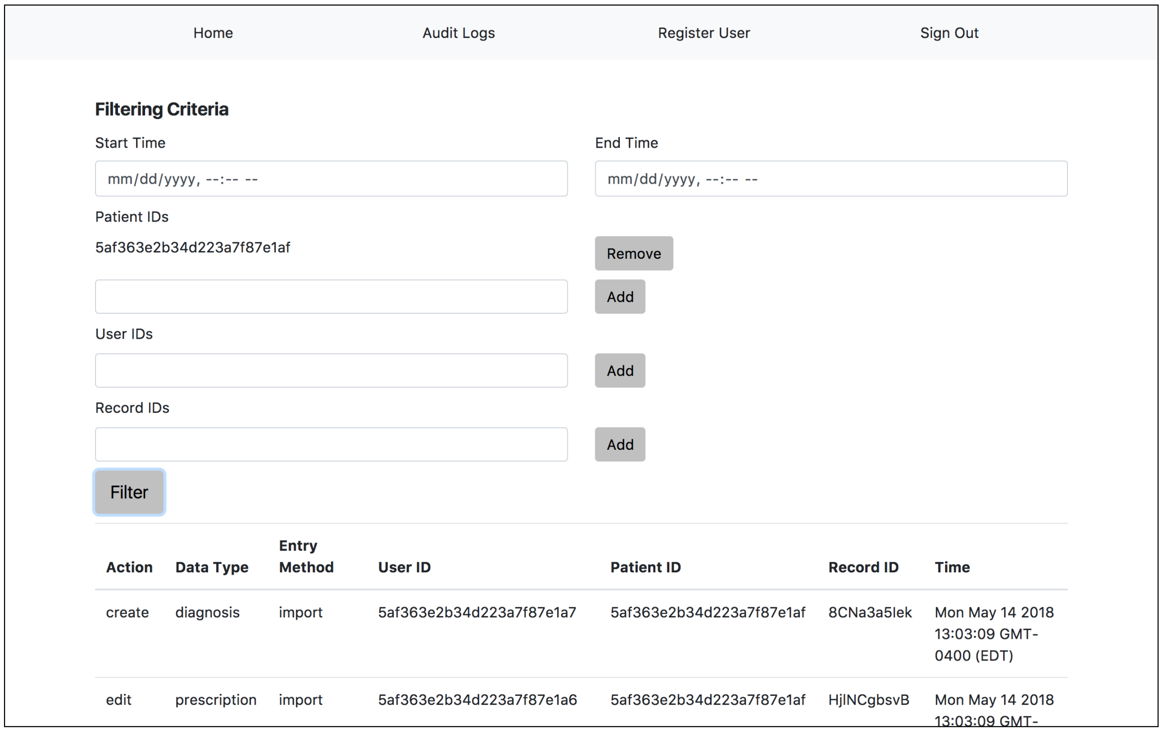
AuditChain simplifies the collection of audit log data, using the blockchain to consolidate audit log data both within and across provider organizations and to store this data in a single commonly understood and agreed-upon format in the blockchain ledger. Each participating organization holds a complete copy of the blockchain ledger on its peer node or nodes, which holds a complete audit trail for each patient within the provider organizations participating in the network.

Suppose Hospital A was directed to produce full patient metadata for Patient B in a case of malpractice that Patient B brought against Doctor C. An administrator, *admin*, could log into her AuditChain account to collect all metadata for Patient B. Then, the administrator would simply have to input Patient B's ID in the AuditChain desktop app to obtain all audit logs for Patient B's record (see Figures 6.1 and 6.2).

When *admin* makes the filtering request through the AuditChain UI, the Node backend makes a call to AuditChain's chaincode deployed to the Hyperledger blockchain network that Hospital A participates in. The chaincode indexes records by patient ID, allowing for easy retrieval from the ledger of the peer that the Node backend makes its request to. As a result, *admin* gets the most up-to-date, consensus-verified record of audit log metadata for Patient B to hand over to the requesting authority.

6.1.3 Limitations

Currently, the AuditChain UI requires users to construct filtering requests with patient IDs, which is not an intuitive or user-friendly method of data retrieval. Future iterations of AuditChain would allow users to search for patients in the system by name, birthday, SSN,



Filtering Criteria

Start Time: mm/dd/yyyy, --:-- --

End Time: mm/dd/yyyy, --:-- --

Patient IDs

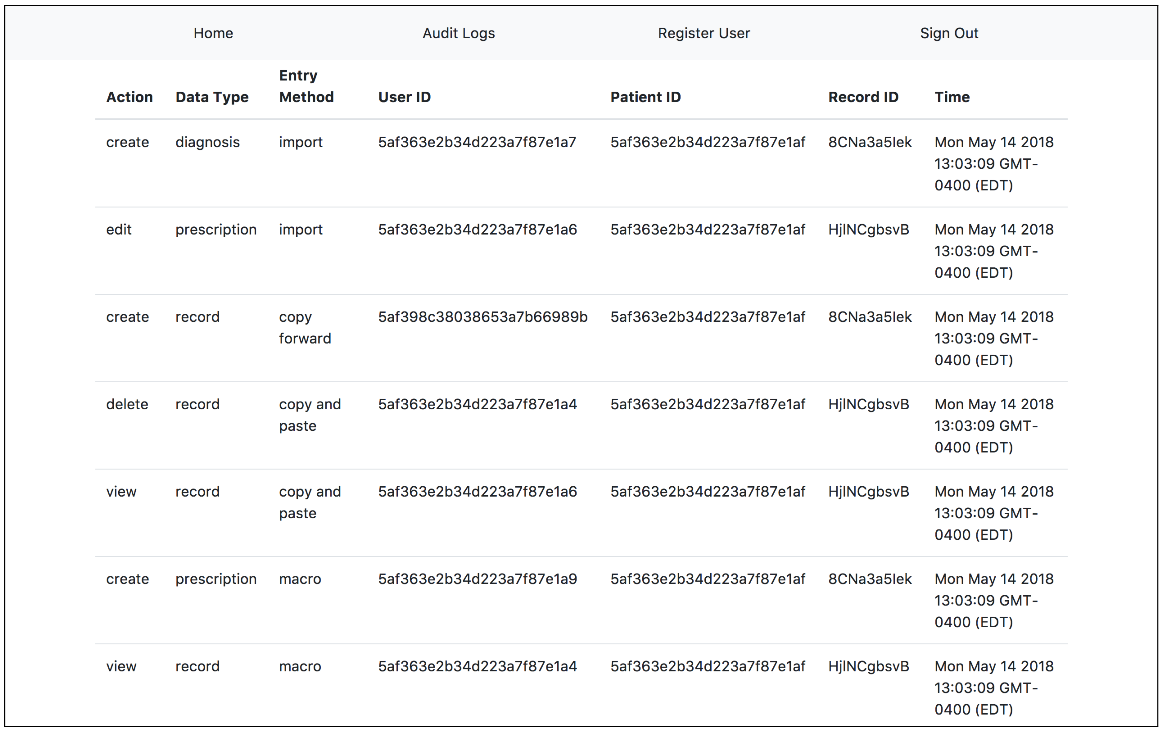
5af363e2b34d223a7f87e1af

User IDs

Record IDs

Action	Data Type	Entry Method	User ID	Patient ID	Record ID	Time
create	diagnosis	import	5af363e2b34d223a7f87e1a7	5af363e2b34d223a7f87e1af	8CNa3a5lek	Mon May 14 2018 13:03:09 GMT-0400 (EDT)
edit	prescription	import	5af363e2b34d223a7f87e1a6	5af363e2b34d223a7f87e1af	HjlNCgbsvB	Mon May 14 2018 13:03:09 GMT-

Figure 6.1: Filtering by patient



Action	Data Type	Entry Method	User ID	Patient ID	Record ID	Time
create	diagnosis	import	5af363e2b34d223a7f87e1a7	5af363e2b34d223a7f87e1af	8CNa3a5lek	Mon May 14 2018 13:03:09 GMT-0400 (EDT)
edit	prescription	import	5af363e2b34d223a7f87e1a6	5af363e2b34d223a7f87e1af	HjlNCgbsvB	Mon May 14 2018 13:03:09 GMT-0400 (EDT)
create	record	copy forward	5af398c38038653a7b66989b	5af363e2b34d223a7f87e1af	8CNa3a5lek	Mon May 14 2018 13:03:09 GMT-0400 (EDT)
delete	record	copy and paste	5af363e2b34d223a7f87e1a4	5af363e2b34d223a7f87e1af	HjlNCgbsvB	Mon May 14 2018 13:03:09 GMT-0400 (EDT)
view	record	copy and paste	5af363e2b34d223a7f87e1a6	5af363e2b34d223a7f87e1af	HjlNCgbsvB	Mon May 14 2018 13:03:09 GMT-0400 (EDT)
create	prescription	macro	5af363e2b34d223a7f87e1a9	5af363e2b34d223a7f87e1af	8CNa3a5lek	Mon May 14 2018 13:03:09 GMT-0400 (EDT)
view	record	macro	5af363e2b34d223a7f87e1a4	5af363e2b34d223a7f87e1af	HjlNCgbsvB	Mon May 14 2018 13:03:09 GMT-0400 (EDT)

Figure 6.2: Results of filtering: data from one patient displayed in table format

address, and other human-readable identifying information.

Second, audit log data returned from a query should be exportable in a standard file format to allow a user such as *admin* to easily share the data with the requesting authority. Currently, the information is only available in the user interface; to convert the data into a shareable format, the user would have to utilize error-prone and potentially insecure copying and pasting.

Though resolving these limitations would improve AuditChain's functionality and usability, doing so exceeds the scope of this project. Neither improvement directly involves the core aim of the project, which is to leverage the permissioned blockchain to improve audit logging capabilities.

6.2 Interoperability

6.2.1 Current Problem

The healthcare sector has not yet adopted an interoperable audit log standard.

6.2.2 AuditChain solution

AuditChain uses chaincode to define a structural standard for audit log data, and makes this standard interoprable by engaging multiple organizations on the same blockchain network. The blockchain consensus mechanism ensures organizations must follow a flexible, agreed-upon consensus protocol when changing the structure or content of audit logs defined by chaincode. Therefore, all participating providers record and access audit log data in the same way, and can only change how data is recorded and accessed by reaching consensus via the protocol defined on the network.

In order for a hospital to be able to participate in the network, the EHR vendor that the hospital uses must first integrate the AuditChain API into its software. (This step is a limitation that reduces the ease with which organizations can join an AuditChain network.) Once the EHR provider has integrated the AuditChain API, the hospital can join the blockchain network as a new participant organization, register its EHR users on the network, and begin recording data on the ledger in the interoperable format defined in the audit log chaincode.

6.2.3 Limitations

AuditChain's major limitation is its naive implementation of audit log structure. Organizations such as HL7 and IETF have proposed audit log standards that are more complex and comprehensive than the simple structure of AuditChain's records. However, a full implementation of one of these standards is outside the scope of this thesis. Having demonstrated that permissioned blockchain technology can be used to impose an audit log standard across multiple organizations, a later iteration of AuditChain could choose the most widely agreed-upon audit log standard and implement it through a new or upgraded version of the audit log chaincode.

A second limitation of AuditChain's current implementation is a lack of a generalized method for linking the authentication credentials and mechanism that EHR users employ within an organization to the cryptographic identities that AuditChain end-users hold on the blockchain network. Forming this link would be necessary functionality to implement for any organization joining the AuditChain network and participating in the interoperable audit log standard. AuditChain demonstrates one way to link local authentication credentials with on-chain cryptographic identities; future iterations should expand upon this baseline demonstration to integrate with ore authentication systems (such as SAML, OAuth2, and other industry standards).

6.3 Audit Log Content

6.3.1 Current Problem

Audit logs do not reliably contain all required, recommended, or useful data.

6.3.2 AuditChain solution

AuditChain ensures that audit log records follow legal requirements and additional recommendations through Hyperledger Fabric's chaincode and flexible consensus protocol (defined by the network's endorsement policy). The audit log enforces audit log structure and content that fulfills HIPAA Meaningful Use requirements, and additionally incorporates further recommended best practices. If new requirements or recommendations arise, provider organizations participating in the AuditChain network could operate via the network's endorsement policy to collectively agree to update and re-deploy a new version of the audit log chaincode to the network that reflects the updated requirements and recommendations.

Suppose that Hospitals A, B, and C participate in a network that runs AuditChain to record audit log data, and that the consensus protocol on the network requires any two of Hospitals A, B, and C to endorse a chaincode upgrade. If HIPAA Meaningful Use is updated with new requirements, any of the organizations could propose an updated version of AuditLog's chaincode that reflects these changes. Suppose Hospital A proposes such an upgrade and installs the new code on its peer, and Hospital B agrees to the upgrade and installs the updated chaincode on its peer as well. Then, either Hospital A or Hospital B could upgrade the chaincode on the network because enough organizations have agreed to the update. Specific processes for updating the audit log chaincode to reflect changing requirements and recommendations would differ based on the consensus protocols in a particular network.

6.3.3 Limitations

AuditChain currently offers no user-friendly way to propose upgraded chaincode or to perform an upgrade. System administrators would need to use Hyperledger Fabric's command-line to install updated chaincode on peers and to perform an upgrade on the network. Secondly, AuditChain's current implementation does not handle backwards compatibility for audit log data recorded with old versions of chaincode. This old audit log data must be kept to comply with federal requirements. Therefore, to fully support chaincode upgrades, AuditChain would need to handle backwards compatibility with old chaincode and audit log content/structure. Backwards compatibility may be achieved by writing the necessary functionality into new versions of chaincode.

6.4 Multiple EHR Applications

6.4.1 Current Problem

Health IT ecosystems often contain multiple applications from different vendors that each record audit data in different ways. Differing audit log structures across applications can complicate the collection and analysis of audit log data for an EHR comprised of multiple applications.

6.4.2 AuditChain solution

AuditChain uses the blockchain to consolidate data in one ledger, as well as chaincode on the blockchain to standardize the data that is stored on this single ledger. Not only is all audit data from one organization stored on one blockchain, but all audit data from all participating organizations is stored on one blockchain. Each application uses the same API to record audit log data, which them becomes accessible in a single location via AuditChain's interface.

Returning to the usage scenario outlined in Section 6.1.2, suppose that Patient B's care spanned multiple applications within Hospital A. Under current audit log systems, metadata would likely be stored in different places and in different formats for each application, requiring manual cross-referencing to construct a comprehensive view of Patient B's metadata. AuditChain simplifies analysis for the authority requesting the metadata by standardizing audit log content and structure across each application in Hospital B, so that records from different applications can be more easily understood in relation to one another.

6.4.3 Limitations

Currently, AuditChain is simply a proof-of-concept implementation, and has not taken into account the multiple types of audit log data that may have to be recorded or how audit log data may differ across applications. For example, audit log data from radiology software likely differs from audit log data from an application to record patient encounters. Proposed standards such as HL7's FHIR protocol and IETF account for differences across applications; future iterations of AuditChain would use proposed standards to inform on-chain audit log structure. However, this detailed analysis of audit log data in different applications is beyond the scope of this thesis.

Chapter 7

Discussion and Future Work

AuditChain uses permissioned blockchain technology to address the problems laid out in Section 2.3 with audit logs in current EHR systems. In this thesis, I evaluate the potential for permissioned blockchain technology to address these problems by highlighting the key features of AuditChain that use the blockchain to resolve these issues (see Chapter 6). Specifically, AuditChain uses the Hyperledger Fabric permissioned blockchain to simplify the production of patient metadata, to enforce the adoption of an interoperable audit log standard by participating organizations, to ensure that participating health care providers record all required, useful, and recommended data in audit logs, and to simplify the cross-referencing of audit logs across multiple EHR applications and among participant provider organizations.

However, AuditChain is currently a proof-of-concept implementation that does not take the complexities of a real-world EHR ecosystem into account. To become a more useful and realistic healthcare tool, AuditChain should incorporate the requirements and needs of provider organizations, clinicians, and existing software systems in to the way that it handles and records audit log data. Several next steps are necessary in order to allow AuditChain to integrate with current software and provide utility to health care organizations.

7.1 Incorporating real-world requirements

In order for AuditLog to be useful in a real healthcare setting, it must take real-world data requirements into account. Health Level Seven (HL7) international has proposed an interoperable audit log standard in the Fast Healthcare Interoperability Resources (FHIR), AuditEvent resource. Currently, the FHIR framework is not widely adopted by the healthcare community, but has shown promise as a lightweight and fairly easily implementable framework to build interoperable healthcare technology [14]. AuditChain would benefit from implementing the FHIR framework, as doing so would follow a growing trend towards adopting the framework.

Evaluating the different sources that audit log data comes from and developing a comprehensive mapping of these resources onto the FHIR data model would further increase AuditChain's viability as a component of an EHR system. For example, AuditChain would need to define how audit log events from radiology software map onto the FHIR standard differently from audit log events from pathology lab software. Defining this comprehensive mapping onto FHIR's lightweight and interoperable framework would allow AuditChain to be evaluated in real-world healthcare contexts and with real audit log data.

7.2 Integration into existing systems

AuditChain's audit log creation functionality currently operates as a standalone API that must be directly integrated into the code of existing software systems. A next step in evaluating AuditChain's efficacy and feasibility would be to integrate the API into an existing open-source EHR platform such as OpenEMR [27]. Integrating AuditChain's API into existing EHR software would allow AuditChain to be evaluated in a real-world context while demonstrating the feasibility of incorporating AuditChain into a pre-existing software system.

7.3 User testing for interface

Health care professionals and security officers should test AuditChain's interface to appraise the way it handles retrieving and displaying audit logs. Specifically, these test users should evaluate what audit log data should be displayed to whom, how this data should be displayed, how the data should be shareable or exportable, and how the interface should allow users to filter and interact with the data. For example, should patients have access to their audit log data at all, and if so, should this access be limited? Should doctors be able to see audit log data, and should the same role-based access restrictions be applied to their audit log viewing capabilities as in the rest of the system? What filters should be applicable to the data, and how should these filters be implemented on the interface? Professionals who interact with EHR software on a daily basis and who frequently handle audit log data would provide valuable insight into ways to maximize the usability and utility of AuditChain's user interface.

7.4 Authentication integration and support

As data security is a vital aspect of EHR systems, AuditChain's authentication functionality should be expanded to integrate with industry standard authentication systems. For example, AuditChain should integrate with authorization frameworks that health care organizations may use such as OAuth2, SAML, two-factor and multi-factor authentication. Integrating authentication mechanisms that health care organizations already employ would allow users to authenticate to AuditChain with the credentials they use in the rest of the EHR system, preventing the need to create separate login credentials for AuditChain.

One complexity of the current local signin approach and of the integration approach proposed above is that the same end-user working at two or more participating provider organizations would have a different on-chain cryptographic identity at each organization. To resolve this, future work could go beyond currently accepted authentication standards to propose a blockchain-based approach to authentication. Storing and checking blockchain authentication credentials on the blockchain would tie end-user identity more tightly to on-chain cryptographic identity and allow users to retain the same on-chain identity across multiple organizations.

7.5 Evaluation at scale

Pursuing the modifications outlined above would enable AuditChain to be deployed and evaluated in a real health care organization. Several performance metrics should be considered during evaluation in a health care environment. First, latency should be measured: how long the blockchain system takes to handle transactions at scale, with multiple requests occurring at short intervals. Second, accuracy should be evaluated. If multiple users simultaneously interact with the same patient record, do some of these audit log transactions become invalidated during the ordering process and are therefore not recorded on the ledger? If some transactions are dropped, how should these be handled to ensure the data is not lost? Third, system load and storage requirements should be evaluated. Replication of ledger data across all peers in AuditChain's network of peers causes high data redundancy. Healthcare organizations already remark on the large storage overhead required for audit log data [6], a problem that would be exacerbated by this data redundancy. Modifications for efficiency, such as data hashing, may be demonstrated to be necessary to reduce storage requirements.

7.6 Possible drawbacks of blockchain

Though blockchain offers several mechanisms to mitigate the issues with audit logs that I identify in this thesis, the technology also presents some limitations and drawbacks when applied to the audit log problem. I will discuss two of these potential issues here: data storage overhead and audit log data privacy in the ledger.

As mentioned in Section 7.5, some hospitals point out that storing audit log data requires considerable space [6]. This is true in an EHR system that only records audit log data from a single hospital. Under AuditChain's architecture, all participating organizations would store all audit log data from each organization in addition to their own, effectively multiplying storage requirements by the number of organizations participating in the network. Storing all audit logs from all organizations on each peer may therefore prove untenable, potentially reducing the applicability of permissioned blockchain technology to the audit log use case.

The second potential issue relates to audit log privacy among organizations. Complexities would arise if a participating health care organization did not want its audit log data on another organization's peer node due to PHI privacy concerns. Furthermore, even if all participant organizations agreed to store their audit log data on other organizations' peer nodes, the audit log chaincode would need to implement carefully constructed access controls to ensure compliance with HIPAA privacy rules.

AuditChain would need to be evaluated in a health care network to determine whether these two concerns are valid. I argue that the potential benefits of permissioned blockchain technology sufficiently overshadow these concerns to make a compelling case for testing AuditChain in a health care environment.

7.7 Summary

A considerable amount of future work is thus possible for AuditChain. The current approach is limited in practicality and feasibility, as discussed in Sections 6.1.3, 6.2.3, and 6.3.3, 6.4.3. Taking the next steps outlined above would allow AuditChain to be deployed, tested, and evaluated in a real-world health care context to more fully determine its viability as a solution to current problems with EHR audit logs.

Chapter 8

Conclusion

Audit logs perform a wide range of functions in health care organizations. They help protect patient privacy by allowing security officers to monitor user activity and PHI access in EHR systems. Larger-scale analysis of audit log data can inform updates to access control policies that better reflect real-world workflows in health care organizations. In legal cases, audit trails can be used as evidence to prove or disprove malpractice allegations. Current research is developing data mining techniques that automate the detection of anomalous behavior in EHRs and that model interactions among EHR users and general workflows in health care organizations.

Each of these use cases requires well-constructed and reliable audit log data that is recorded in a common, understandable format. However, EHRs do not always record all necessary or useful information in audit logs; furthermore, the lack of an interoperable standard complicates the analysis of audit logs both within and across organizations, limiting the extent to which health care organizations can implement these use cases.

AuditChain uses Hyperledger Fabric's permissioned blockchain system to fill the gap currently preventing health care organizations from extracting full utility from audit log data. AuditChain uses the blockchain to consolidate the audit trail in a single ledger that is shared among all applications within a health care organization and among all organizations participating in the network. This consolidation simplifies the retrieval and analysis of audit log data. Furthermore, AuditChain's audit log chaincode imposes a standard structure on audit log data, facilitating the interoperability of audit logs among all health care organizations participating in the network. The interoperable structure that this chaincode imposes upon audit logs ensures that all necessary and useful data is recorded with each log entry. AuditChain's specific focus on using the blockchain to improve audit logging departs from previous literature on blockchain in the health care sector, and demonstrates the useful features of blockchain that may improve upon the current standard.

Several future improvements are necessary in order to make AuditChain a viable component of a live health care ecosystem. Audit log structure in chaincode should take the requirements of different EHR applications into account; furthermore, health care professionals should test and verify AuditChain's usability when operating at scale. Evaluation at scale should also address storage requirement and data privacy concerns that arise as a result of putting audit log data on the blockchain's distributed ledger. However, despite the need for future work to realize utility as a health care technology, AuditChain does demonstrate the ways in which permissioned blockchain technology may improve the usability of audit logs and allow health care organizations to fully capitalize on the capabilities they offer. Future work should seek to turn AuditChain into a usable system in order to evaluate its performance in a real health care environment.

Acknowledgments

Thanks to my advisor, Sean Smith, who provided direction and guidance while allowing me to define and pursue a topic that interested me. Thanks to Andrew Gettinger and Ross Koeppel, whose insight into the world of health care and health care technology proved invaluable in writing this thesis. Thanks to Elena Horton, my blockchain thesis buddy and general partner-in-crime. And, finally, thanks to my family - to Lloyd and Lyle for doing work alongside me at our kitchen table at home (even though we were all on break), and to my parents for encouraging me the whole way through.

Bibliography

[1] Tareq Ahram, Arman Sargolzaei, Saman Sargolzaei, Jeff Daniels, and Ben Amaba. Blockchain Technology Innovations. *2017 IEEE Technology & Engineering Management Conference*, pages 137–141, June 2017.

[2] Elli Androulaki, Artem Barger, Vita Bortnikov, Christian Cachin, Konstantinos Christidis, Angelo De Caro, David Enyeart, Christopher Ferris, Gennady Laventman, Yacov Manevich, Srinivasan Muralidharan, Chet Murthy, Binh Nguyen, Manish Sethi, Gari Singh, Keith Smith, Alessandro Sorniotti, Chrysoula Stathakopoulou, Marko Vukolić, Sharon Weed Cocco, and Jason Yellick. Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, pages 30:1–30:15, New York, NY, USA, 2018. ACM.

[3] Asaph Azaria, Ariel Ekblaw, Thiago Vieira, and Andrew Lippman. MedRec: Using Blockchain for Medical Data Access and Permission Management. *2nd International Conference on Open and Big Data*, 2016.

[4] Christian Cachin. Architecture of the Hyperledger Blockchain Fabric. Technical report, IBM, July 2016.

[5] Konstantinos Christidis and Michael Devetsikiotis. Blockchains and Smart Contracts for the Internet of Things. *IEEE Access*, 4:2292–2303, 2016.

[6] Ricardo Cruz-Correia, Isabel Boldt, Luis Lapao, Ctia Santos-Pereira, Pedro Rodrigues Pereira, Ana Margarida Ferreira, and Alberto Freitas. Poor Quality of Hospital Information Systems Audit Trails. *BMC Medical Informatics and Decision Making*, 13(84), 2013.

[7] Economist. The Great Chain of Being Sure About Things. *Economist*, October 2015.

[8] Andrew Gettinger. Conversation about Issues with Modern EHR Technology. private communication, December 2017.

[9] Hyperledger. Architecture Explained, 2017.

[10] Hyperledger. Hyperledger Fabric Model, 2017.

[11] Hyperledger. Ledger, 2017.

[12] Matthew P Keris. A Pandora's Box: The EMR's Audit Trail. *COUNTERPOINT*, February 2017.

[13] Jason King, Ben Smith, and Laurie Williams. Audit Mechanisms in Electronic Health Record Systems: Protected Health Information May Remain Vulnerable to Undetected Misuse. *International Journal of Computational Models and Algorithms in Medicine (IJCMAM)*, 3(2):23–42, 2012.

[14] Georgios C Lamprinakos, Aziz S Mousas, Andreas P Kapsalis, Dimitra I Kaklamani, Iakovos S Venieris, Anastasis D Boufis, Panagiotis D Karmiris, and Spyros G Mant-

zouratos. Using FHIR to Develop a Healthcare Mobile Application. In *Wireless Mobile Communication and Healthcare (Mobihealth), 2014 EAI 4th International Conference on*, pages 132–135. IEEE, 2014.

[15] Daniel R. Levinson. Not All Recommended Safeguards Have Been Implemented in Hospital EHR Technology, December 2013.

[16] Bradley Malin, Steve Nyemba, and John Paulett. Learning Relational Policies from Electronic Health Record Access Logs. *Journal of Biomedical Informatics*, 44(2):333 – 342, 2011.

[17] Chrissa McFarlane, Michael Beer, Jesse Brown, and Nelson Prendergast. Patientory: A Healthcare Peer-to-Peer EMR Storage Network v1.1. Technical report, Patientory, May 2017.

[18] M. Mettler. Blockchain Technology in Healthcare: The Revolution Starts Here. In *2016 IEEE 18th International Conference on e-Health Networking, Applications and Services (Healthcom)*, pages 1–3, Sept 2016.

[19] Max Mitchell. Patient Should Have Access to 'Audit Trail' Data, Judge Rules. Online, May 2017.

[20] Satoshi Nakomoto. Bitcoin: A Peer-to-Peer Electronic Cash System. Technical report, Bitcoin, 2008.

[21] U.S. Department of Health and Human Services. Audit Protocol - Updated April 2016, April 2016.

[22] Office of the National Coordinator for Health Information Technology. Test Procedure for §170.314(d)(3) Audit Report(s), December 2012.

[23] Office of the National Coordinator for Health Information Technology. Test Procedure for §170.314(d)(2) Auditable Events and Tamper-Resistance, March 2014.

[24] Office of the National Coordinator for Health Information Technology. Hospital Progress to Meaningful Use by Size, Type, and Urban/Rural Location. online, August 2017.

[25] Office of the National Coordinator for Health Information Technology. Office-Based Health Care Professionals Participating in the CMS EHR Incentive Programs. online, August 2017.

[26] The Office of the National Coordinator for Health Information Technology. Meaningful Use: Meaningful Use and the Shift to the Merit-Based Incentive Payment System. online, September 2017.

[27] OpenEMR. OpenEMR. online, 2018.

[28] Daniel Palmer. Blockchain Startup to Secure 1 Million e-Health Records in Estonia, March 2016.

[29] Lillian Røstad and Ole Edsberg. A Study of Access Control Requirements for Healthcare Systems Based on Audit Trails from Access Logs. *Annual Computer Security Applications Conference*, 2006.

[30] IBM Global Business Services Public Sector Team. Blockchain: The Chain of Trust and its Potential to Transform Healthcare - Our Point of View. In *"Use of Blockchain in Health IT and Health-related Research" Ideation Challenge*, August 2016.

[31] Feng Tian. An Agri-Food Supply Chain Traceability System for China Based on RFID & Blockchain Technology. In *Service Systems and Service Management (ICSSSM), 2016 13th International Conference on*, pages 1–6. IEEE, 2016.

[32] Helma van der Linden, Dipak Kalra, Arie Hasman, and Jan Talmon. Inter-Organizational Future Proof EHR Systems: A Review of the Security and Privacy Related Issues. *International Journal of Medical Informatics*, 78:141–160, 2009.

[33] Tom Walsh. Privacy and Security Audits of Electronic Health Information (2014 Update), 2014.

[34] Justin M. Weis and Paul C. Levy. Copy, Paste, and Cloned Notes in Electronic Health Records. *CHEST*, 145(3):632–638, March 2014.

[35] Gavin Wood. Ethereum: A Secure Decentralized Generalized Transaction Ledger. Technical report, Ethereum, August 2017.

[36] X. Xu, I. Weber, M. Staples, L. Zhu, J. Bosch, L. Bass, C. Pautasso, and P. Rimba. A Taxonomy of Blockchain-Based Systems for Architecture Design. In *2017 IEEE International Conference on Software Architecture (ICSA)*, pages 243–252, April 2017.

[37] He Zhang, Sanjay Mehotra, David Liebovitz, Carl Gunter, and Bradley Malin. Mining Deviations from Patient Care Pathways via Electronic Medical Record System Audits. *ACM Transactions on Management Information Systems*, 4(4):17:1 – 17:20, 2013.

Appendices

Appendix A

Github Repository

The code and documentation for AuditChain can be found at the following link: https://github.com/jessie-anderson/audit-chain