Secure Audit in Support of an Adrenal Cancer Registry

Anthony Stell, Vedant Chauhan and Richard Sinnott
School of Computing and Information Systems, University of Melbourne, Melbourne, Victoria, Australia

Keywords: Adrenal Cancer Registry, Clinical Process Audit, Blockchain.

Abstract: This paper describes the use of blockchain technology to ensure the integrity of data logs for a clinical registry, providing a technological means for a secure audit of that registry. The characteristics of a secure audit – tamper-resistance, verifiability, searchability and privacy – are described in their application to this registry, then an evaluation is performed detailing the use of blockchain to achieve these audit goals. The clinical registry tested – supporting ENSAT (the European Network for the Study of Adrenal Tumors) – is a production repository of clinical, phenotypic and genetic information about patients with adrenal cancer, a rare but often serious condition that affects approximately 1 in 600,000 of the world population. The registry is implemented using a standard n-tier web application, with a MySQL database back-end, and Java/JSP business logic and user interface. The information contributing to the full audit of data and usage of the registry, is captured in the application log-files, which are stored in two "mirrored" formats: ASCII text files compiled through the Java log4j project and in a MongoDB NoSQL database. Following a discussion of the relevant supporting features, the fully implemented solution – a private blockchain known as "ensatChain" – is evaluated for overall security, using the Microsoft "STRIDE" threat model.

1 INTRODUCTION

Due to the increasing sophistication and influence of digital technologies, online disease registries are a growing source of clinical and research endeavour. Many are established as web-accessible applications, built upon cloud infrastructures, for instance with web servers, databases and operating systems running on virtual machines. Often capturing phenotypic and genotypic information, such resources allow information on patients and their conditions to be shared by clinicians with wider research communities outside of a given clinical setting, and also potentially with patients and their advisory groups.

Such registries are particularly useful when the condition is serious but rare. In recent decades, research into the diagnosis and treatment of rare diseases has started to attract larger-scale international funding such as the European Union FP7 (FP7, 2019) and Horizon 2020 (Horizon 2020, 2019) programs. One of the main goals of such initiatives has been to overcome the lack of data which impedes the development of knowledge about the conditions, and their causes and potential treatments. Due to the nature of these rare conditions, multi-centre, international studies have nearly always been necessary to move the clinical science forward, and this is one of the primary drivers of the utility of online registries.

Such registries are now widespread, and as is common with maturing generations of technologies, the focus of their development is now more often on the quality of the data they contain. Key aspects to ensuring high quality of data include understanding the mechanisms that provide assurances about the provenance and security of that data. A fundamental feature therefore is the establishment of an effective audit of the registry.

The ENSAT (European Network for the Study of Adrenal Tumours) (ENSAT, 2019) registry is a production repository of data and information about adrenal cancer in Europe and requires just such tools and methods for audit. The registry itself was designed and released in 2009, has gone through three version upgrades, and is currently operational with over 400 clinical users, and over 16,000 unique patient records (a very high number for a condition as rare as adrenal cancer). The registry collects phenotypic, genetic and biomaterial information, with almost 100,000 distinct clinical annotations. It is housed at the University of Melbourne in Australia, is used for multiple full-phase clinical trials and studies, and has global data interconnections, in particular to

Stell, A., Chauhan, V. and Sinnott, R.
Secure Audit in Support of an Adrenal Cancer Registry.
DOI: 10.5220/0009160402370248
In *Proceedings of the 13th International Joint Conference on Biomedical Engineering Systems and Technologies (BIOSTEC 2020) - Volume 5: HEALTHINF*, pages 237-248
ISBN: 978-989-758-398-8; ISSN: 2184-4305

the American Asian Australasian Adrenal Alliance (Else, 2019).

In this project, we propose a technique using blockchain technology (Nakamoto, 2008) to ensure the required aspects of audit meta-data in log files generated by a clinical registry using both ASCII text files (captured using the Apache log4j (Apache log4j, 2012) project) and the same log information stored in a NoSQL database (MongoDB (MongoDB, 2019)).

The rest of this paper is structured with a discussion on the clinical and technical motivation, related literature, the implementation of this solution applied to the ENSAT adrenal cancer registry, an evaluation of the solution security when applied to the Microsoft STRIDE threat model (Potter, 2009), and a final discussion on the strengths and weaknesses of the solution.

2 MOTIVATION

Two considerations motivated this work: the increasing need for close and accurate audit of digital assets, especially in the clinical sphere, and the emergence of "one-way" trust establishment technologies such as blockchain.

2.1 Clinical Audit Requirements

In the clinical and healthcare industry, there are many requirements to maintain compliance with regulations and organizational policies. At a technical ("low") level, a successful audit requires the comprehensive documentation of all transactions within a process. In a clinical context, transactions can be defined as a sequence of activities that have a source and destination. Building upon this history of transactions, an organisation/process should have a complete audit log (or "trail") detailing the start and completion of every single transaction in chronological order.

In common with the use-cases of audit generally, there are four well-defined characteristics of a successful audit: tamper-resistance, verifiability, searchability and privacy (Waters, 2004). To make these abstract concepts more concrete, consider the following two examples of inconsistent data:

1) An administrative error, where a clinical organization have modified a patient's details, only discovered in post-hoc analysis. This incident could be handled more swiftly and accurately if there was an easily-accessible audit trail.
2) A clinician deliberately modifies a piece of patient information without authorization, such as the chemotherapy dosage of the patient. An audit check of information only would not pick up such an edit, but if a tamper-proof solution was engaged, the inconsistency of provenance would be highlighted, helping to identify modifications after-the-fact.

In both of these examples tamper-resistance and verifiability are clear requirements. Possibly less obviously, searchability (the ability to find and identify the issues) and privacy (the ability to protect a patient's details from outside/undue influence) are also key to ensuring both issues are fixed correctly.

Applying these considerations to an online disease registry, the question then becomes how technology can be applied to establish these four characteristics, ensuring that the information in permanent (e.g. in a database or logfiles) or volatile (e.g. rendered through business logic) memory are transparent, consistent and secure (i.e. have not been tampered with).

Blockchain technology (Nakamoto, 2008) is known for its ability to provide tamper-resistance, and therefore allows a metric of trust to be defined. This would appear to be an appropriate solution to establish the tamper-resistance and verifiability requirements described above. Within blockchain, a chain of blocks is created and if one of them has been tampered with, then the chain is inconsistent, and the details of these inconsistencies are readily available as notifications of potential modifications. In this project, we propose the use of MongoDB to provide searchability and PKI-style authentication and authorization methods to implement privacy.

2.2 Trust-establishment Technology

Developed to maturity in the late 00's and given public prominence by its use in cryptocurrencies such as Bitcoin (Bitcoin, 2019), blockchain is a decentralized distributed ledger which has a key characteristic of immutability, allowing the integrity of the entities within it to be asserted (or not). Conceptually it is described as a chain of blocks, with attributes made from the previous hash, the root hash, a timestamp, a nonce (seeding "number once"), and data. Each subsequent hash is derived from the root hash through the hyper-repetitive process of "mining".

A technique that underlies the properties of Blockchain are Merkle Trees – tree data structures where the child nodes produce a hash of each non-leaf node. As idempotent information is retained

throughout the hash information within the nodes (i.e. information about the tree root can be inferred from isolated node information), Merkle Trees enable scaling with reliable data integrity.

Transactions – identified as critical to both clinical audit processes and to the development of a ledger technology – are atomic events that are authenticated, validated, and tracked. Security of transactions is established using cryptographic techniques and their integrity are determined using consensus algorithms, which is the process of decision-making by a network (one of the major unique strengths of blockchain and the source of the "mining" process). Examples of consensus algorithms determining a transactions authenticity are the use of Proof of Work or Proof of Stake (101Blockchains, 2018).

Access to a blockchain determines its access "type" (or visibility), which has ramifications for its effectiveness. A public blockchain (with no access restrictions) provides high integrity, but completely open transparency to its content. A private blockchain (with access control strictly enforced) conversely provides high privacy – only a small group of selected individuals can view the content – but low integrity as verification is similarly restricted to that small group of individuals. A combination of these two access types ("consortium") attempts to find a reasonable compromise between the two.

Whilst blockchain still has underlying challenges, such as a heavy reliance on one-way cryptographic protocols, the similarity to the requirements and structure of a typical audit, clinical or otherwise, is apparent. With the emergence of this technology, the opportunities for verifiability of transactions, the immutability of records and transparency in the entire audit process, are many.

3 RELATED LITERATURE

Studying the related literature in this space broadly falls into two categories: the history of trust-establishment technologies leading to the development of blockchain, and current solutions providing secure audit.

3.1 Historically-relevant Technology

Traditionally, data protection in logs were maintained by writing once and reading multiple times on an optical drive (Bellare and Yee, 1997). In the early days of the Internet, the (reasonable) assumption was that a single computer was not likely to be compromised. If an attack happened then the local log data could be sent over the network to a remote logging site. Replicating the log data over multiple locations was the only solution if any site was compromised, providing a disaster recovery strategy, but poor privacy of potentially sensitive organisational data.

(Schneier and Kelsey, 1999) made a significant improvement to this problem by creating log files, which were in an encrypted format. The proposal was to share a secret authentication key from an untrusted machine, with a trusted machine. To add another log entry required computing an encryption key to create encrypted log entries. These encrypted log entries would make it difficult for the attacker to decrypt the file without the correct key. However, under this system, if the network contained an insecure machine, then it could be compromised by an attacker by creating new log entries without changing other existing entries (i.e. an undetected insertion attack).

Snodgrass et al. (Snodgrass, Yao, and Collberg, 2004) focused on the integrity of audit logs, using SHA1 signatures on database management systems. Their system contained a separate audit log validator which was set up to provide diagnostic information if the system was compromised. This generated files very similar to a ledger maintaining the overall state of the system, a clear precursor to blockchain.

(Waters et al., 2004) proposed a system similar to that created by Snodgrass et al, but used additional hash functions to maintain the integrity of the database system, and to aid in searching the encrypted log. Two complementary techniques based on symmetric and asymmetric key encryption were used, allowing the shared secret challenge of symmetric encryption to be solved. The practical advantage of this approach is the ability to search encrypted data results over many applications. However, the technique results in a significant overhead of identity-based encryption leading to performance limitations (in common with many public-key cryptography solutions).

3.2 Current Solutions

The closest contemporary solution attempting to achieve similar audit security as proposed here, is a commercial application known as LogSentinel (White Paper - SentinelTrails Documentation, 2019). It provides a secure audit log service that is *"simple to integrate and guarantees the integrity of all your audit data"*. The system addresses the challenges of being tamper-proof and searchable with a solution that uses a hash chain mechanism (similar to, but

distinct from, blockchain) to audit log entries. As of the current date of publication, the application runs as an alpha (non-production) version on an Ethereum blockchain. The most significant limitation of this appears to be the confidentiality of the data, which is open to all as it is implemented on a public blockchain. This was a defining reason why this solution was not chosen to implement an audit on the ENSAT registry.

Whilst the technologies reviewed here provide aspects of the functionality required, none are comprehensive. There also exist other techniques such as remote logging and replication, but these have been deemed outside the current scope of this project.

4 REQUIREMENTS AND SOLUTION

Each of the approaches listed so far in this section provide additional concepts to the notion of auditing log files. In nearly all cases, the applications are improving the security, but also have to compromise in terms of performance or the confidentiality of data. It has been identified, therefore, that a solution is required to assure all four characteristics of secure audit, mentioned previously: tamper-resistance, verifiability, searchability and privacy.

4.1 Tamper-resistance

The primary nature of tamper-resistance is that there should be a guarantee that only a specified user with appropriate privileges can create and alter valid log entries. If a system is compromised (e.g. a bad actor gains root access to a machine) it is very difficult to prevent modifications of a log, either through editing, deletion or modifying the mechanism for future outputs. Therefore, a mechanism for preventing such modifications must be inherently resistant to even high-privilege system access. An ideal extra function would be the ability to "checkpoint" the log creator's status periodically - either through copies of the actual log data or through another function (e.g. a signature) of it's log data to another owner, to ensure credibility via a trusted third party.

4.2 Verifiability

There must be a mechanism to definitively verify the integrity of a log, for a secure audit. Audit logs can either be verified publicly - i.e. validated by anyone with properly-authenticated public information, such as a public key of the log systems server, or an authenticated hash of all existing audit entries. Or, they can be verified in a private manner, using a trusted verifier, such as a designated party holding one or more secrets, such as a message authentication code (MAC) key.

4.3 Searchability

Given that the data in an audit log may be sensitive, it can be encrypted or secured privately for authorized parties. However, for secure - but functional - audit, legitimate search access should be allowed to an appropriate subset of all audit log entries (e.g., all valid entries that match the "Smith" keyword). Therefore, capability delegation is essential to allow an investigator to search and view a narrow scope of log entries. For instance, if Alice Smith wanted to investigate all records related to her, the audit escrow agent could give her the ability to search for all records that match the "Smith" keyword, but not give her anything more. The alternative is to provide searchability for authorized peers to access the system and data.

As an aside, with the development of the European General Data Protection Regulation (GDPR) legislation in 2017, an interesting paradox has developed in this regard: how to make information immutable and resistant to modification, whilst simultaneously maintaining the “right to be forgotten” of data subjects (van Humbeeck, 2017). Research into this issue will likely be the subject of the next iterations of this work.

4.4 Privacy

A final consideration is the type of data contained in an audit log-file. Generally, this is not considered when discussing functional audits, but is critical to the provision of a secure audit, as the content type potentially relates to privacy/confidentiality of data, especially in a clinical context. To provide this, each application should provide access to data to authorized users only and data should not be available to outside entities.

5 SYSTEM ARCHITECTURE

Given the discussion above, the solution presented in this paper has been designed to provide tamper-proof and verifiable integrity of a set of log data (either in file or database format), which is also searchable, whilst maintaining appropriate levels of privacy.

The base technology for this solution is the use of a private blockchain with a secure block and transaction structure, and a "proof-of-work" consensus algorithm. This private blockchain has been called "ensatChain" and has been developed keeping the four fundamental aspects of secure audit in mind. The ENSAT registry environment is described first, followed by the detail of ensatChain, which covers three of the secure audit aspects: tamper-resistance, verifiability and privacy. The searchability aspect is described across two contexts: the searchability of the ensatChain ledger, and the searchability of the log information content itself (in both Apache log4j and MongoDB NoSQL format). Figure 1 outlines a schematic of the log information output and its translation into a blockchain.

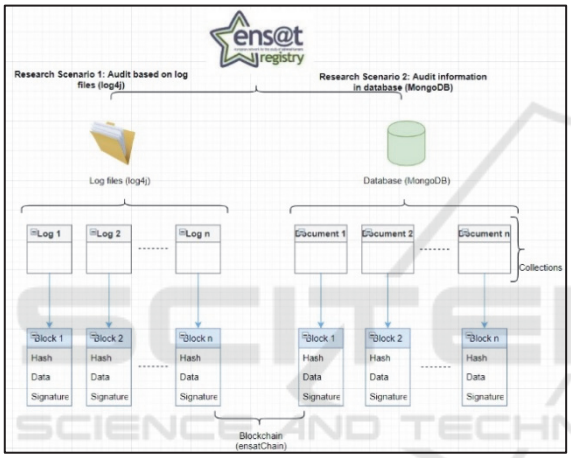

Figure 1: Hierarchical architecture of log information and their translation into blocks, using both Apache log4j and MongoDB formats.

5.1 Registry Environment

ENSAT (European Network for the Study of Adrenal Tumours) is an online adrenal cancer registry detailing associated phenotypic, genetic and bio-sample information. Though pseudonymously de-identified, it nevertheless holds sensitive information about patients and clinical trials/studies. It is implemented as an online web application, holding data in a MySQL database, controlled through business logic and user interfaces coded in Java and JSP (Java Server Page).

The registry log information is generated through outputs from the application to Apache log4j files, which are parsed and input into a MongoDB (NoSQL) database. Notifications about the status of the registry, the log-files, and the disaster recovery functions (nightly database dumps, zipped, encrypted and archived off-site) are sent to a small group of privileged administrators.

The web container itself is Apache Tomcat (v8) running on Ubuntu OS (v16), on a dedicated virtual machine, hosted at the University of Melbourne, through the NeCTAR cloud infrastructure program (National eResearch Collaboration Tools and Resources). The log information is stored on a virtual machine with similar underlying infrastructure, and the MongoDB database primarily accessed through a Robo3T viewer (Robomongo, 2019).

5.2 ensatChain

ensatChain is a Java-based blockchain application, utilising an AMD Radeon RX 500 Series processor for mining.

5.2.1 Blocks, Hashes and Cryptographic Protocols (Tamper-resistance)

Each block (whether an "actual" block or "virtual" – discussed further in section 5.3) in the ensatChain has its hash, data, time-stamp, nonce (number once), and the hash of the previous block as outlined in figure 2. This means that each block contains a hash of the previous block as well as its own hash, which is mathematically derived from that prior hash.

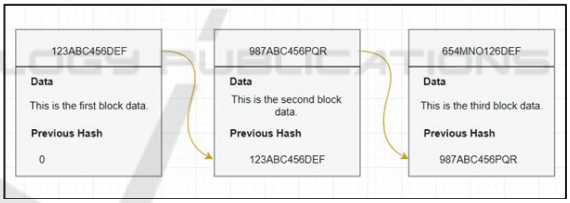

Figure 2: Chained block structure.

Therefore, if the last block's data is modified then this block's hash will also vary from what is expected and this will, in turn, affect all of the hashes of the subsequent blocks. This calculation helps to decide whether the block's signature has been altered, and therefore whether the ledger has been tampered with.

A strong cryptographic algorithm is desirable to generate a reliable hash, so ensatChain utilizes the SHA3-256 algorithm. SHA3 (Secure Hash Algorithm 3) practices sponge construction to make the randomness generation flexible. Sponge construction uses random permutation to "absorb" (or input) any size of data and to "squeeze" (or output) any volume of data whilst acting as a pseudo-random function with regard to all previous inputs (Wikipedia, 2019d).

It is a more modern, and notably stronger protocol, particularly when protecting against "length

extension" attacks (where the size of an encrypted value is used for an attack), than previous successors (e.g. SHA1 and SHA2), and in terms of performance, can run at similar speeds with the same accuracy, to SHA2-256 and SHA2-512, by cutting the "pre-image resistance" (a known encrypted value) in half. For any size of string input, SHA-3 generates a 256-bit output size. This predictable size of output makes the algorithm ideal for calculating the hash for the block using the previous hash, time-stamp, data, and nonce.

5.2.2 Proof of Work (Verifiability)

Proof of Work (PoW) is a consensus algorithm that supports the flat architecture of the network of nodes that run ensatChain. Each node is equivalent to others with no elected leaders and promotes direct communication between nodes on the network.

"Mining" is the process by which proofs for block addition to the blockchain are generated, with each participant in the mining process called a "miner". PoW expects miners to add a block to the blockchain by solving complicated mathematical puzzles. These puzzles require massive computational power — for instance, hash functions or integer factorization, or finding the output to other challenges without knowing the input. The complexity of the puzzle is very difficult to guess and usually depends on the maximum number of users, minimum power and total load on the network. A new block is created and added once a miner solves the puzzle, and the transaction is confirmed by the other network nodes.

Due to this high barrier of complexity, this consensus process imbues trust and reliability to the network by producing a portion of the information that is hard to break by external malefactors. The difficulty level of the PoW determines the overall chain's level of difficulty, and hence is a useful measure of strength of resistance to external attack.

The reason that PoW was chosen for ensatChain (a public blockchain feature applied to a private one) is to maintain the decentralized nature of ensatChain (i.e. as mentioned above, none of the nodes are "leaders") compared to consensus algorithms more typically found in private blockchains, such as Proof of Authority (PoA) (101Blockchains, 2018), which tend towards a more centralised nature.

A major drawback of PoW is that it suffers from slow throughput if the number of nodes is large and there is a correspondingly high energy consumption for the mining process. For future developments of ensatChain, other approaches may be explored such as Proof of Stake (PoS) or Practical Byzantine Fault Tolerance (pBFT) (101Blockchains, 2018). Figure 3 (at the end of the paper) gives a pictorial summary of the strengths, weaknesses, and other features of different consensus algorithms (101Blockchains, 2018).

Finally, the actual individual validation process in ensatChain is similar to most blockchain solutions. ensatChain compares the hashes by checking through all blocks in the chain. If a peer wants to verify that, say block 101, has not been tampered with, then the peer will compute the appropriate hash, check it with the hash stored on block 101 on ensatChain and repeat the same process for the previous block's hash (also stored in block 101). If all comparisons result in

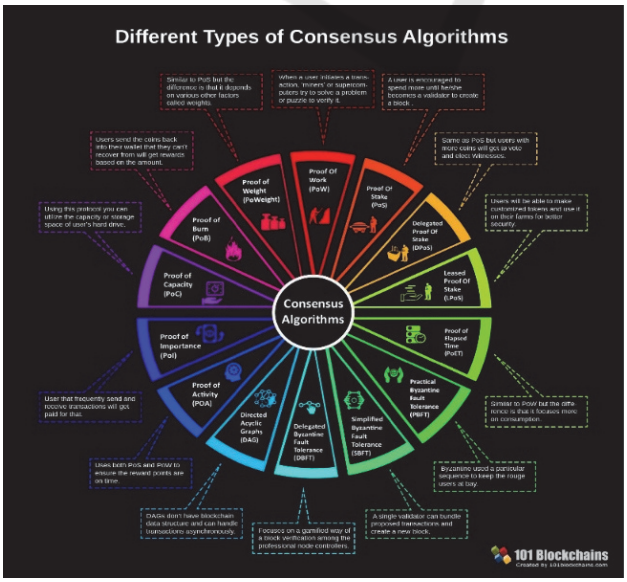

Figure 3: Variants of consensus algorithms (101Blockchains, 2018).

equality, then the chain is valid and unbroken. However, if the comparison is false, then the ledger has been altered.

5.2.3 Structure and Access (Privacy)

ensatChain is a hybrid development using features from both private blockchains (permissioned access control) and public blockchains (a consensus algorithm to establish integrity).

The private blockchain feature is the role-based application access which controls the privileges assigned to authorized individuals, providing limits of privacy to the network and data. This is obviously desirable in an environment such as ENSAT, where confidential clinical information should be handled with clear privacy bounds.

Additional benefits are that it is faster and cheaper compared to public blockchains because of the lower number of nodes on the network, which in turn requires significantly less energy and resources to reach a consensus.

5.2.4 Transaction Structure and Implementation (Searchability)

A transaction in ensatChain is a collection of meta-data log information. In common with most public-key cryptographic systems, for signing and verifying the transaction, each user must have a public and (secret) private key. The public key acts as the recipient's address while the private key signs the transaction as a method of authentication. The public key is utilized to verify that the signature in the transaction is authentic, and that the information has not been tampered with. ensatChain uses an ECDSA elliptic curve algorithm (Wikipedia, 2019a) to generate these key-pairs.

To facilitate the secure audit, every transaction has the following structure:

- The sender's address
- The receiver's public key (as address)
- The log meta-data to be stored on the ledger
- A reference to the previous transactions, i.e. outputs.
- The sender's private key signing the transaction with an ECDSA signature

ensatChain uses MongoDB as a distributed database for storage of the current state of the ledger (not to be confused with the use of MongoDB to store the actual log-file data content). This implementation provides decentralization and transparency within the (private) network.

5.3 Performing the Log Audit

The ENSAT registry generates a vast amount of log information reflecting the application-level usage and user behaviour. It also makes connections to the lower levels of infrastructure, such as web container, database, and even in some cases, the underlying operating system. This allows the log to form an audit trail that can aid with diagnosing system or application issues, disaster recovery, and to monitor user behaviour. As described previously, there is a large motivation to make this audit trail secure and to develop a deep understanding of the activities within a complex system.

Currently the ENSAT registry is set up to generate and store log information in two ways: 1) ASCII text files generated by the Apache log4j project, and 2) a searchable MongoDB NoSQL database. The content of each mirrors the other. It should be noted that the reason for the two formats is due to production migration of one format (Apache log4j) to the other (MongoDB), which has been a necessary feature of system maintenance, but has also provided an opportunity to compare the efficacy of each format against the other.

Conceptually, the log4j method lends itself to the notion of a "chain of blocks" more naturally, as distinct files are generated on a daily basis. Therefore, the insertion of a hash into the already-delimited blocks (the individual files) is somewhat intuitive.

In the MongoDB implementation a further step of outlining what constitutes a block is required. In this case the delimiter is the date, as this then matches the natural delimiters of the log4j method. The storage implementation for MongoDB is a collection of documents, bound together using standard NoSQL methods, and a natural side-effect of database storage is the improvement in efficiency of scalability and storage space.

5.3.1 Hash Generation

Vast amounts of data are contained in a single day's log information, therefore an efficient way of generating and storing a hash is to convert it into a segment of a hash chain, which provides a stripped-down version of a blockchain, but with the same properties of immutability (Wikipedia, 2019b). These segments are then stored in ensatChain, rather than the entirety of the day's log information itself. This hash is generated in the same way as the hash of a block (see section 5.2.1) using the SHA3-256 cryptographic algorithm on the same set of data outlined in section 5.2.4.

Once the hash of a day's log file is generated, it is added to the log file itself with an ECDSA digital signature of the created owner. This creates a segment of a hash chain and a full chain is created by appending the corresponding log files to each segment.

For the Apache Log4j project, these considerations of generated hashes are represented using standard ASCII text. In MongoDB the data records are stored as binary representations of JSON objects, known as BSON (JSON and BSON 2019). Therefore, the hashes are generated, used to create hash chains as above, but then are converted into BSON format, and attached to the document collection within the database.

5.4 Threat Models (STRIDE)

To perform a basic evaluation of the responsiveness and resilience of this system, a threat model was used to attempt to obtain definitive metrics of the security of the overall system, as well as considering the four characteristics identified. In general, a threat is a potential or actual adverse situation that can be accidental, such as a storage device failure, or malicious such as a Denial of Service (DoS) attack. Threat modelling improves security by clearly defining vulnerabilities and describing countermeasures to prevent or mitigate the impact of threats to the system.

The central aim of threat modelling is to allow an informed decision to be made about the commitment of resources and effort in order to keep a system safe, to whatever level of risk is deemed acceptable. This variable changes as technology, and the associated threat landscape, evolves. Therefore, threat modelling can be considered an iterative process of determining a security profile for every asset within an organisation.

Developed at Microsoft, the STRIDE threat model (Potter, 2009) is used here to assist in that process of understanding the vulnerabilities of the ensatChain system with regard to the overall security, but also with particular focus on the four characteristics of a secure audit.

The component vulnerabilities of the STRIDE model are defined as follows:

- Identity Spoofing: this is the act of illicitly accessing and using another entity's authentication information. For example, obtaining the username and password of doctor to modify chemotherapy dosage data.
- Data Tampering: this is the unauthorized alteration of information. For instance, the modification of data by an unauthorized party, whether in storage in a database, or in flow across a network.
- Repudiation: this vulnerability occurs when parties deny executing an action, and the alleging other party does not have a piece of compelling evidence otherwise. Non-repudiation is a technique to counteract such repudiation threats. For instance, if a doctor updates a patient's medicine in the system, with his signed digital credentials, then bad consequences occur and the doctor attempts to (falsely) repudiate the action, an administrator can show the doctor's timestamp digital signature as evidence (i.e. non-repudiation).
- Disclosure of Information: this is a vulnerability where protected information is exposed to unauthorized parties.
- Denial of Service (DoS): this vulnerability is where valid users are denied access to a service. For instance, temporarily disabling access to a web server, or a dedicated attack on a server to disrupt its function availability to a large user base.
- Privilege Elevation: this vulnerability is where an unprivileged entity obtains access to a restricted system.

The overall system is evaluated against this specific list of vulnerabilities, with the relevance to the four characteristics of secure audit highlighted.

6 EVALUATION

The evaluation relates to the four characteristics of secure audit and to the overall security of the system as evaluated through the STRIDE threat model. This section outlines the conclusions reached about each.

6.1 Secure Audit Feature Evaluation

Evaluating the four characteristics of secure audit, the following conclusions were drawn:

Privacy: ensatChain correctly hides protected information from external parties and exclusively provides data access to authorized users. The mechanism for this is a password file combined with identity key. However, complete public key-pair access would be a more secure and hence desirable solution.

Tamper-resistance: integrity is maintained in the log information by comparing the hashes computed and stored in ensatChain against those in the log4j files or MongoDB document collections.

Searchability: this aspect depends on the storage mechanism of the log information. In log4j the searchability was very limited (string matching within bulk text files). However, in either the ledger chain or the MongoDB log storage, searchability is facilitated through the document query tools, and is much more flexible and powerful.
Verifiability: by sustaining the transaction history in the ensatChain, all the entries are accessible to authorized users, and can be validated individually and retrospectively through the consensus algorithm driven by all the network nodes.

6.2 STRIDE Threat Model

Evaluating the overall system security against the STRIDE model, the following conclusions were drawn:

6.2.1 Spoofing

The ENSAT registry log information is stored on a central server, with replicated backups on separate servers and archived copies off-site. ensatChain is implemented to operate at each server (on both log4j and MongoDB formats), each being designated as a "peer node". Each peer node has the same view as the others, every new transaction modifies the state of the system overall and is replicated at each node using MongoDB as the distributed blockchain database.

User authorization is maintained using a protected password file, and associated identity key for all authorized peers (users), though as mentioned in 6.1, a full public key-pair mechanism would be preferable. Only these peers will be able to access the network. If these measures are breached, then a further layer of protection is that the network must still confirm any modifications by this breached node as valid.

ensatChain currently does not provide any device authorization. Any device would be able to connect to the distributed ledger as long as they produce the correct credentials. There is also currently an inability to distinguish clearly between remote and local commands issued over the peer network.

6.2.2 Tampering of Data

ensatChain introduces immutability through the blockchain mechanisms described in section 5. Immutability in this sense indicates that an entity can still modify the records, but that these modifications will be tracked and notified to the appropriate administrative authorities. Related to this, is that the validation process of ensatChain allows the integrity of log information in the registry to be maintained.

To protect information over the network (c.f. at rest in storage) secure channel protocols (TLS, SSH, etc.) allow message integrity to be ensured. In this context, the "message" is the transfer of log information from the central ENSAT server to the distributed nodes, and the publishing of block contents to the rest of the ensatChain ledger. The message packets are encrypted with the digital signature of the owner and can only be decrypted by authorized parties.

6.2.3 Repudiation

A transaction in ensatChain is digitally signed and time-stamped to a particular date and time, with the same signature applied to the content of the log files (in both log4j and MongoDB formats). The use of the time-stamp information in hash generation allows the property of non-repudiation to be enforced. If a user logs into the ensatChain network, their activities are tracked, and all access to all information is also tracked. If any deviation or discrepancy is noticed, an alert to the administrator is issued.

6.2.4 Information Disclosure

ensatChain is a permissioned blockchain with strictly enforced access control. Similar to that described in section 6.2.1, the enforcement of this is through an identity key and password file, which in turn enforces privacy/confidentiality to authorized entities only. These authentication barriers are implemented "in depth" at many layers - for instance, even if an attacker manages to access the entire the ensatChain network, the binary data of the log files is encrypted, and the secure channels used on network traffic ensure privacy on those exchanged messages.

6.2.5 Denial of Service (DoS)

These attacks are amongst the most commonly encountered on blockchain networks. The disruption of service provision and the creation of an unresponsive environment are the most significant effects of such an attack (i.e. service availability).

As ensatChain is a distributed blockchain network, it is similarly susceptible to such attacks. The decentralised nature does mitigate somewhat against a DoS attack, as remaining uncompromised nodes could continue to operate whilst others are down, and no central hub acts as a single point of failure. Similarly, the use of a Proof of Work algorithm would mitigate

against a DoS attack as it requires an attacker to exhaust a larger amount of resource than otherwise.

However, as ensatChain is a private blockchain, the number of peers will be limited, and this would also amplify the effects of a DoS attack.

6.2.6 Elevation of Privilege

Similar to "insider attacks", this is one of the most challenging vulnerabilities to protect against. A well-secured application can be protected with multiple internal security layers like virtual private networks, firewalls, etc. and still an insider attack could penetrate all these layers of defence. The authentication and authorization mechanisms provided by ensatChain are unlikely to be enough to withstand such an attack. The only guaranteed defence in this case is the tamper-resistance property of blockchain to track modifications, though this would be unlikely to be a sufficient full defence (as it can only act retrospectively).

Table 1 gives an overview of the threats evaluated with this threat model.

Table 1: The STRIDE evaluation of ensatChain.

Type of threat	Security property	Evaluation
Spoofing	Device authz	No
	User authz	Yes
Tampering	Data integrity	Yes
	Message integrity	Yes
Repudiation	Non-repudiation	Yes
Information disclosure	Message integrity	Yes
	Message confidentiality	Yes
Denial of service	Availability	Yes (with caveats)
Privilege elevation	Authorization	No

According to the table, "No" means the technique is still susceptible to the threat, while "Yes" means a technique can be used to counter or mitigate the attack. The use of binary "Yes"/"No" responses gives a broad sense of the security for simplicity, but within those responses are further grades of subtlety (e.g. there are mitigations against DoS attacks, but their effectiveness depends on the scale of the attack).

7 DISCUSSION

The system described in this paper attempts to provide the mechanism for a secure audit of an online clinical registry. This chapter discusses the relative strengths and weaknesses of the system developed to date, both in terms of general security and design and implementation choices.

7.1 Strengths

The main strength of this project is that the implementation of ensatChain is believed to achieve the four significant characteristics of a secure audit:

– Tamper-resistance: with the integration of ensatChain into the ENSAT registry, it is now possible for administrators to be notified about unauthorized modifications of the log information in the system.
– Privacy: with the use of private blockchain features in ensatChain, such as authorized access control, the benefits of other blockchain features such as tamper-resistance are realised, whilst maintaining the privacy of system and user information.
– Searchability: the use of MongoDB to support both the log information and the ensatChain ledger, allows for searchable stored log and hash data.
– Verifiability: the use of blockchain consensus algorithms, allows the ensatChain transactions to be verified by the network of users and nodes.

From a security perspective, the system strengths include protection against immediate threats such as user authorization, data integrity, message integrity, confidentiality, non-repudiation, and availability.

In the context of performance, strengths include the use of the AMD Radeon RX 500 series Graphical Processing Unit (GPU), which has a better cost to profitability ratio than the nearest alternative, which is the Antminer S9 Hydro. Also the overall performance (accuracy at set speed) is high, due to the low number of nodes in the ensatChain network (currently five), requiring a small amount of time to process and mine. However, as the system scales, block generation time would increase non-linearly and this would become a weakness.

7.2 Weaknesses

The main weaknesses of this project include the use of the Proof of Work (PoW) consensus algorithm which is often inefficient and can consume a high amount of energy and resources. The functionality of the system is not currently hindered by this

inefficiency due to the low number of nodes in the network. However, as mentioned in section 7.1, as the system scales this will likely become an issue. The solution in this case will be to use a faster and more efficient consensus algorithm, such as Proof of Stake (PoS) or Practical Byzantine Fault Tolerance (pBFT).

Related to this is the fact that PoW provides a weaker difficulty level of hash generation, therefore making the mining process less resilient to external attack. Choosing a different consensus algorithm, such as pBFT, would also have the effect of increasing this difficulty level.

The ensatChain implementation overcomes most threats as outlined in the STRIDE model. However, threats like device authorization and elevation of privilege are still potential vulnerabilities.

It is also vulnerable, as are most closed systems, to internal or insider attacks, which are the most dangerous yet difficult to protect against. The most ensatChain can do to counter such a threat is to assert authorization challenges using a "defence in depth" modality, but these are mitigations, not complete prevention.

Similarly, there are a family of blockchain-specific attacks (e.g. 51% attack (51% Attack, 2014), or Sybil attacks (Wikipedia, 2019c)) which are difficult to protect against. ensatChain suffers from these vulnerabilities as well. Developing further mitigations against these attacks either within ensatChain or at a higher application level will be the focus of future work.

8 CONCLUSIONS

This paper has presented a description of an implemented technology that attempts to provide secure audit of the meta-data generated by an adrenal cancer registry, where "secure" is defined by the four characteristics of tamper-resistance, verifiability, searchability and privacy. The potential applications of such audit security are many and are particularly relevant to the clinical and health domains, where vast amounts of sensitive and important data are stored and transmitted on a daily basis.

The use of blockchain technology has allowed an exploration of the benefits of public features (e.g. a Proof-of-Work consensus algorithm) versus private features (e.g. permissioned access control) to attempt to gain the benefits of both. Additionally, the application was implemented on both log files generated as ASCII text, and those stored in a NoSQL database.

The overall system was tested against the Microsoft STRIDE threat model and was found to perform well on many of the aspects of security and secure audit previously discussed but still had overall security weaknesses when addressing device authorization or privilege elevation vulnerabilities.

The implementation is now operating in production on the ENSAT adrenal cancer registry and with some improvements, will be applied to other clinical registries developed by the Melbourne eResearch Group (MeG) in the near future.

ACKNOWLEDGEMENTS

The work leading to this solution has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement n° 259735 and from the European Science Foundation (ESF).

REFERENCES

ENSAT (2019), http://www.ensat.org/ [Online; accessed 19-May-2019]

FP7 (2019), https://ec.europa.eu/research/fp7/index_en.cfm [Online (archived); accessed 22-Nov-2019]

Horizon 2020 (2019), https://ec.europa.eu/programmes/horizon2020/en [Online; accessed 22-Nov-2019]

Nakamoto, Satoshi et al. (2008), *"Bitcoin: A peer-to-peer electronic cash system"*, In: Working Paper

Else, Tobias (2019), *"Providing Evidence: Step by Step – a letter from associate editor, Tobias Else, MD"*, In: Hormones and Cancer, Feb 2019, vol 1:10, pp 1-2

Apache Log4j (2012), http://logging.apache.org/log4j/1.2/manual.html, [Online; accessed 19-May-2019], [Page updated; 13-May-2012]

MongoDB (2019), https://www.mongodb.com/, [Online; accessed 19-May-2019]

Potter, Bruce (2009), *"Microsoft SDL Threat Modelling Tool"*, In: Network Security, vol 1:2009, pp 15-18

Waters, Brent R et al. (2004), *"Building an Encrypted and Searchable Audit Log.",* In: NDSS. Vol. 4, pp. 5–6

Bitcoin (2019), *"What's the Difference Between Bitcoin (BTC) and Ethereum (ETH)?"*, https://www.cryptocompare.com/coins/guides/what-s-the-difference-betweenbitcoin-btc-and-ethereum-eth/ [Online; accessed 20-May-2019]

Bellare, Mihir and Bennet Yee (1997), *"Forward integrity for secure audit logs"*, Technical report, Computer Science and Engineering Department, University of California at San Diego

Schneier, Bruce and John Kelsey (1999), *"Secure audit logs to support computer forensics",* In: ACM Transactions on Information and System Security (TISSEC) 2.2, pp. 159–176

Snodgrass, Richard T, Shilong Stanley Yao, and Christian Collberg (2004), *"Tamper detection in audit logs"*, In: Proceedings of the Thirtieth international conference on Very large data bases-Volume 30. VLDB Endowment, pp. 504–515

White Paper - SentinelTrails Documentation (2019), https://docs.logsentinel.com/en/latest/whitepaper.html, [Online; accessed 20-May-2019]

Van Humbeeck, Andries (2017), *"The Blockchain- GDPR Paradox"*, https://medium.com/wearetheledger/the-blockchain-gdpr-paradox-fc51e663d047

Robomongo (2019), https://robomongo.org/ [Online; accessed 22-Nov-2019]

101blockchains.com (2018), *"Different Types of Consensus Algorithms Infographic*", [Online; accessed 20-May-2019] [Online; accessed 22-Nov-2019]

(2019a), Elliptic-curve cryptography — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Elliptic curve_cryptography&oldid=895042492,

[Online; accessed 20-May-2019]2019b), Hash-chain – Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/wiki/Hash_chain, [Online; accessed 22-Nov-2019]

JSON and BSON (2019), https://www.mongodb.com/json-and-bson, [Online; accessed 20-May-2019]

51% attack (2014), https://en.bitcoinwiki.org/wiki/51%25_attack [Online; accessed 19-May-2019]

(2019c), Sybil attack — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Sybil_attack&oldid=893108975, [Online; accessed 20-May-2019]

(2019d), SHA3 – Wikipedia, The Free Encyclopedia,

https://en.wikipedia.org/wiki/SHA-3 [Online; accessed 22-Nov-2019]