Witnesses for the Doctor in the Loop

Peter Kieseberg[1,2,3](✉), Johannes Schantl[2,3], Peter Frühwirt[1], Edgar Weippl[1], and Andreas Holzinger[2,3]

[1] SBA Research, Favoritenstr. 16, A-1040 Vienna, Austria
pkieseberg@sba-research.org
[2] Research Unit HCI-KDD, Institute for Medical Informatics, Statistics and Documentation, Medical University Graz, Auenbruggerplatz 2, 8036 Graz, Austria
[3] CBmed - Center for Biomarker Research in Medicine, Stiftingtalstrasse 5, 8010 Graz, Austria

Abstract. The "doctor in the loop" is a new paradigm in information driven medicine, picturing the doctor as authority inside a loop supplying an expert system with information on actual patients, treatment results and possible additional (side-)effects, as well as general information in order to enhance data driven medical science, as well as giving back treatment advice to the doctor himself. While this approach offers several positive aspects related to P4 medicine (personal, predictive, preventive and participatory), it also relies heavily on the authenticity of the data and increases the reliance on the security of databases, as well as on the correctness of machine learning algorithms. In this paper we propose a solution in order to protect the doctor in the loop against responsibility derived from manipulated data, thus enabling this new paradigm to gain acceptance in the medical community.

Keywords: P4 medicine · Fingerprinting · Data driven science

1 Introduction and Motivation

While the concept of the "doctor in the loop" seems to be a logical consequence of the application of machine learning technologies and derived knowledge into medical science, one major problem arises: The doctor in question is forced to trust the results derived from algorithms based on the authenticity of stored data to a large extent, while being seen as the primary responsible party during information provisioning, as well as during treatment, i.e. the doctor retains responsibility or, in case he/she is involved in the selection of the source data, even gains more, while loosing control over the process. With the technology available to tackle large amounts of complicated data in real time through Big-Data techniques, results derived from such processes may even become more uncontrollable. This opens up the problem of acceptance of the "doctor in the loop" approach by medical personal: The question is the trustworthiness of the underlying data and execution chains, especially considering manipulation, e.g. in the aftermath of a wrong treatment. Thus, in order to mitigate this risk for the overall concept, manipulations in the underlying database need to be detected, as well as control over the information entered by the doctor needs to be safeguarded against subsequent manipulation. This also includes the manipulation-secure logging of execution chains of enrichment and analytics algorithms and workflows. The contribution of this work can be summarized as follows:

- We provide a model of the "doctor in the loop concept" including an abstract architecture of its entities with respect to security.
- Attack scenarios and attacker models against this approach are devised.
- Based on these models, strategies for mitigation are defined.

2 Background and Related Work

The problem of securing infrastructures relying on human behaviour has been discussed throughout the last decade and more, being on of the very fundamental problems for computer security [1]. The problem is often related to the issues of awareness [2] or missing usability in security [3], as well as other subtopics, also including the sharing of data between different entities [4]. This is also often related to the issues of providing health related information to other clinicians [5] or to automated systems [6].

2.1 Chained Witnesses

The term "chained witnesses" was coined in [7], where the authors propose a technique for securing internal mechanisms of databases against manipulation. The main advantage of this approach over the multitude of approaches described in the literature was resilience against an attacker model that included the database administrator as possible adversary. While this is discussable in most real-life systems where the database administrator is seen as a trusted entity, this is especially interesting in the "doctor in the loop" concept.

The main principle of this approach lies in appending a so-called *witness* for each transaction that is issued against the database to the internal logging mechanisms: The database storing the information is considered as untrusted, furthermore, even file system administrator rights are assumed for the attacker. Let D_i be the i^{th} data record written to the database at time t_i. Furthermore, we assume that $\mathcal{H}$ is a cryptographically secure one-way hash function, $\mathcal{T}$ is a trusted third party and $\mathcal{R}$ is a secure pseudo random number generator (PRNG) and r_i is the result of its i^{th} iteration. The witness for transaction D_i is calculated as

$$w_i = \mathcal{H}(w_{i-1}||D_i||t_i||r_i) = \mathcal{H}(w_{i-1}||D_i||t_i||\mathcal{R}(r_{i-1}))$$

with $||$ denoting string concatenation. The tuple (t_i, w_i) is then called the *signature* of the record D_i. In order to start the hash-chain, an initialization phase is required: A trusted third party $\mathcal{T}$ selects a random number s as seed for the PRNG and thus generates r_0 by using the PRNG on s. Furthermore, the initial witness is defined as $w_0 := \mathcal{H}(r_0)$.

Due to the definition of the witnesses as chained hashes, any changes in older data sets lead to cascading changes in all subsequent witnesses (Figure 1 shows the chaining). For the verification, the data of the protected internal logging mechanisms is executed against an old trusted backup under the premise of $\mathcal{T}$ and compared to the investigated database instance. In [7] the authors propose several mechanisms for achieving this kind of manipulation security in real-life environments, especially targeting internal database mechanisms for providing rollbacks (so-called *transaction logs*[1]). Furthermore, the database management system (DBMS) must be modified in a way to provide the calculation of the respective witness as an atomic action, invisible to the administrator, i.e. the mechanism for writing the transaction log needs to be modified directly in order to fetch the random numbers r_i and calculate the witness immediately, without leaking r_i to the administrator. As shown in [7] the implementation of such a process can be done for MySQL, furthermore, the authors pointed out solutions for closed source DBMSs based on the database replication logs.

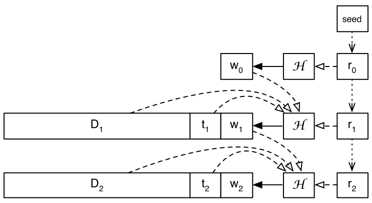

Fig. 1. Chained Witnesses ([7]).

2.2 The Doctor in the Loop

The concept of the "doctor in the loop" is an extension of the increasingly frequent use of knowledge discovery for the enhancement of medical treatments together with the "human in the loop" concept: The expert knowledge of the doctor is incorporated into "intelligent" systems (e.g. using interactive machine learning) and enriched with additional information and expert know-how. Using machine learning algorithms, medical knowledge and optimal treatments are identified. This knowledge is then fed back to the doctor to assist him/her (see Figure 2).

While general techniques regarding data driven research have their own problems with respect to privacy protection (see e.g. [8]), an additional major problem for the doctor in the loop lies in guaranteeing the trustworthiness of the data provided by other entities and by analysis workflows. Furthermore, the data provided by the doctor needs to be secured against subsequent manipulation in the case of a cover-up, either by the system, or by the doctor himself. In this work we will solely focus on this problem and leave the problems of privacy protection and data leakage discovery to the literature [4,9].

[1] It must be noted that the term "logs" is slightly misleading, since these are not human readable log files, but internal mechanisms for ensuring transaction safety.

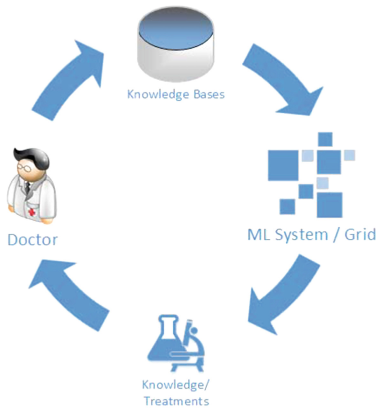

Fig. 2. The doctor in the loop.

3 The Approach

The approach outlined in this section is based on the generic concept of the "doctor in the loop" as described in 2.2. In order to motivate the chaining approach, we will define the entities and their relations, including the chaining mechanism.

3.1 Entities and Relations

For our analysis, we define a more specific model for the doctor in the loop. Figure 3 gives an overview on the components:

– The **Doctor**, who is the main expert in the cycle, collects data from patients, including their reactions to individual treatments and eventual other effects. Furthermore, he/she provides additional knowledge from his/her experience and sanity-checks results. All data he/she provides to the system is sent to the Knowledge Base, which also provides him/her with the relevant feedback.
– The **Knowledge Base** provides the store for the data and all results of workflows and external resources, as well as the only means for communication between the doctor and the other entities. This entity is the primary target for our chained witnesses approach, since all data that is transferred between the relevant entities for the "doctor in the loop" approach utilize it. The knowledge base may also host stored procedures for the analysis of the data, i.e. parts of the ML-grid are implemented as stored procedures inside the knowledge base.
– The **Grid** serves as a generic model for a machine learning / reasoning structure that takes input data and returns results using analytics algorithms. The grid may be implemented as external mediation tool, as well as in the form of internal stored procedures inside the knowledge base. In our concept, the exact definition of the grid will be kept on an abstract level, since securing will be done on the side of the underlying database of the knowledge base.

- **Interfaces** from other entities to the knowledge base are logged by the underlying DBMS. This includes all transactions changing data or structures in the database, as well as the change and invocation of stored procedures that may implement part of the grid.
- The entity **Medical Research** denotes external knowledge bases that serve as external data input to the grid, or to the knowledge base.
- **ML Research** provides the grid with new algorithms for the analysis of the data stored in the knowledge base.

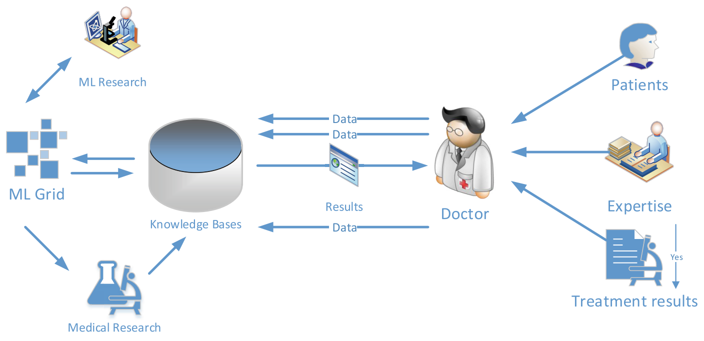

Fig. 3. Entities and Relations.

3.2 Interaction and Chaining

For the abstract approach we only consider a general scenario where a generic data receiving decision maker $\mathfrak{M}$ (e.g. the doctor) sends data to a generic data store $\mathfrak{S}$ (e.g. the knowledge base). Furthermore, an entity $\mathfrak{P}$, the data provider, operates on the same database and delivers a result to $\mathfrak{S}$. $\mathfrak{M}$ takes a result (e.g. a treatment) based on the results and returns additional information, especially on the reaction of the patient and other (side-)effects. Furthermore, $\mathfrak{M}$ controls the results stored in $\mathfrak{S}$ with respect to sanity-checks based on his background knowledge and issues respective corrections to $\mathfrak{S}$ that are subsequently used by $\mathfrak{P}$. From a security point of view this especially implies that the exact order of the transactions with respect to the knowledge base is of vital importance in order to guarantee authenticity.

Data Provider: The model of $\mathfrak{P}$ is selected to be as generic as possible and covers all single data providing entities except the decision maker. This especially includes all parts of the grid, as well as additional data sources with respect to 3.1. Due to the assumption that $\mathfrak{P}$ might be some proprietary entity, incorporating additional mechanisms for controlling the decision provider(s) is not reasonable. Furthermore, $\mathfrak{P}$ might in reality consist of several different entities (internal stored procedures and external workflow engines), i.e. $\mathfrak{M}$ might provide data to and receive information from several different $\mathfrak{P}_i, i \in \mathbb{N}$ data providers. Thus, the $\mathfrak{P}$ only needs to fulfill the following prerequisites:

1. All results are written to $\mathfrak{S}$, there is no additional side channel to $\mathfrak{M}$, i.e. $\mathfrak{M}$ and $\mathfrak{P}$ are independent.
2. Everything sent to $\mathfrak{S}$ by $\mathfrak{P}$ is signed using state of the art cryptographic technologies and is therefore assumed to be unforgeable.

Especially requirement two seems to be strong, still this is standard in many current communication protocols.

Data Store: The data store possesses an internal table structure for storing all collected data, invoked enrichment algorithms, as well as the received data, protected with the chained witnesses approach: For each entry in the transaction log D_i, the respective signature (t_i, w_i) is stored (see [7]). It must be kept in mind that the only connection between two entries D_i and D_j lies in their timely succession, i.e. all changes in all tables are stored in the same transaction mechanism, ordered by the time of entering t_i. In the **setup phase**, the initialization is done by a trusted third party $\mathfrak{T}$ (see below). We furthermore assume that the data store is run independently from the underlying physical server, i.e. $\mathfrak{S}$ possess administrator privileges over all tables, as well as full access to the file system for enrichment and processing of incoming and outgoing data, as well as for restructuring the database layout (tables, views ...), including full control over log settings. Still, it does not possess root privileges on the underlying machine, which is run by $\mathfrak{T}$ or another trusted entity. In addition, the data store frequently sends a backup to $\mathfrak{T}$, which is validated as shown below. The newly validated database image iteratively serves as the new base point for the next validation cycle.

Decision Maker: The decision maker $\mathfrak{M}$ is independent from the data store, i.e. it does not have any control over $\mathfrak{S}$. Furthermore, it is also independent from all data providers (see there). In this approach we assume that the decision maker is honest in general (see data insertion).

Trusted Third Party: The trusted third party $\mathfrak{T}$ controls and manages the random values needed in the chained witnesses approach for the data store. During the setup phase, a new random seed s is selected and the first random value r_0 is generated using the cryptographically secure pseudo random number generator (PRNG). Furthermore, the first witness $w_0 = \mathfrak{H}(r_0)$ is sent to $\mathfrak{M}$. Additionally, $\mathfrak{T}$ can be the entity responsible for running the physical server for $\mathfrak{S}$, including root privileges. While $\mathfrak{T}$ is thus in a very powerful position, $\mathfrak{T}$ must be independent from all other entities, especially from all data providing parties, thus possessing no interest in data manipulation. Furthermore, interaction between $\mathfrak{S}$ and $\mathfrak{T}$ is limited to the setup phase and during the verification of authenticity.

Network Providers: The network provider is responsible for enabling the communication between the data provider and the decision maker. We assume that all traffic is protected by end-to-end encryption against eavesdropping, other attacks by a malicious network provider, e.g. denial of service, are not inside the scope of this paper. This also holds true for the underlying public key infrastructure that is needed in order to facilitate the encrypted communication.

Data Insertion: The decision maker is modeled to receive data from outside machine based systems, especially by the patients during personal consultation. As outlined later in the attacker model 4.1, we assume that the decision maker is in principle honest, i.e. at the time of consultation, no harm towards the patient is intended from his/her side. This also means that the data entered into the database is correct at the time of insertion. All data received by the patients is immediately stored to $\mathfrak{S}$.

Verification of Authenticity: In the verification step, $\mathfrak{T}$ extracts the internal transaction logs (this is possible using a method provided in [10]) and uses a trusted backup as starting point for consecutive execution of the log entries, thus verifying the witness for each transaction by using the secret initialisation vector s and the PRNG. The first encountered invalid witness provides the position of a manipulation of the log. Furthermore, the result of the verification is compared bit-wise to the current database, thus being able to uncover changes done directly in the underlying file system.

4 Evaluation

4.1 Attacker Models and Attack Vectors

In this section we give a description of the attacker models and attack vectors with respect to the assets of the "doctor in the loop" approach.

Data Provider and Decision Maker: Both entities could have the interest of manipulating data on the data store in case of cover-ups, e.g. manipulating previously delivered incorrect data. The main attack vector of the decision maker lies in updating data on the data store, either provided by itself, or result (treatment) data from the data provider. The data provider possesses the same attack vectors, in addition, he/she might try to manipulate and/or re-execute stored procedures that operate on the data in order to cover up wrong results. In order to keep the concept as simple and strong as possible, we assume that there is no dedicated secure application controlling access to and from the database by the entities, i.e. the entities write their changes directly into the data store. This is especially important in order to be secure against SQL-injections or related attacks by default.

Data Store: The data store itself is an important entity in the overall concept, since it serves as the central data exchange platform and is thus vital for providing trust into the "doctor in the loop" concept. The database administrator controls all access to the database, including the possibility to undo logs, as well as change arbitrary data and structures. Furthermore, not only the database itself, but also the underlying file system, can be of interest for an attacker: As outlined in related work [10], file carving techniques can be used in order to retrieve or manipulate data by directly accessing the database files on the file system. In this evaluation we thus concentrate on these two fundamental attack vectors:

- The **Database Administrator (DBA)** possesses administrator privileges on the database itself, including the ability to change logging routines and user rights, as well as read access to the underlying file system.
- The **File System Administrator (FSA)** can modify arbitrary files on the server, including the files belonging to the database, as well as the OS (system) logs. He has no access to the database query interface though.

Neither of the two attackers possesses root privileges on the respective database server.

4.2 Security Evaluation

In this Section we will analyze the respective assets that could be targeted by the attackers modelled in the previous section.

Manipulation through the database (All except FSA): Both, the data provider, as well as the decision maker could be interested in reissuing incorrectly entered data. In case they act with their own privileges, i.e. as data provider or decision maker, every modification of data is stored in the internal logs, together with the respective timestamp of the change, making it easily detectable. In case the attacker possesses administrator privileges on the DBMS (DBA), the internal log mechanisms are under the full control of the attacker, except for the chaining: Since the attacker still does not possess root privileges on the server, it is impossible for him/her to read the value r_i from the RAM, which is then used in the generation of the witness with $\mathfrak{H}$. Since $\mathfrak{H}$ is a cryptographic hash function, when given $h := \mathfrak{H}(\mathfrak{h}')$, h' cannot be deduced from h.

Targeting stored procedures (DBA): The database administrator can execute and modify every stored procedure on any stored data set. Still, in case executions change any data in any table on the whole database, the changes are logged in order to retain transaction safety.

Manipulation through database files (FSA): The file system administrator could bypass all logging mechanism by manipulating data directly in the underlying database files. This includes the transaction log and other rollback mechanisms, which have to be invoked by the DBMS. Using the witnesses these changes remain detectable, since the resulting database in the verification step will be different from the one currently on the server. Still, the attacker could insert data via the file system and remove it right before the validation, making this manipulation undetectable. As a countermeasure, the validation process should be done frequently at random times. Furthermore, we propose to use the chaining witnesses approach with respect to special logs containing checksums of the database files.

Manipulation of the DBMS: The attacker could remove the chaining witnesses from the source code of the DBMS and install a recompiled version. While this is possible, this action would require root privileges on the server. Furthermore, modifications on the binary could be easily detected via frequent comparison of checksums of the respective code to the originally issued version.

Modification of the transaction mechanism (DBA): The authenticity of the information in the transaction mechanism/log is protected by the chained witnesses approach, so every manipulation can be detected under the given attacker model and the manipulated record can be identified. This could only be circumvented by deleting the whole log, which itself is an highly obvious manipulation pointing to the database administrator.

Combined attackers: In the above examples we split the attacker between the DBA and the FSA, still, the resilience of the approach is retained even in case the attacker possesses both privileges. This can be directly inferred from this section, since the chaining is done on DBMS level, without the involvement of either, the DBA or the FSA.

4.3 Limitations

The limitations of the proposed approach can mainly be attributed to limitations of the original chained witnesses approach, especially regarding the lifetime of the internal transaction logs and problems related to an attacker possessing root privileges. Furthermore, the approach only works with DBMSs that actually provide transaction safety and thus provide the respective mechanisms.

More specific to the architecture provided in this paper, the main limitation lies in the assumption of independence of the different entities, which in reality may not be guaranteed due to the setup of the overall environment (e.g. a hospital running a "doctor in the loop" approach might control the doctor, the database and parts of the grid, as well as $\mathfrak{T}$).

5 Conclusion and Future Outlook

In this work we provided an approach for protecting decision relevant data in a generic "doctor in the lop" setup against manipulation targeting the underlying database. This is especially needed in order to increase trust in the "doctor in the loop" concept for both sides, the involved medical personal, as well as external partners and research labs providing results based on the data. The work is based on the chained witnesses approach outlined in [7]. Future work is especially needed in the area of usability in order to effectively incorporate the architecture into the daily routines without introducing even more overhead for the medical personal, thus enabling the "doctor in the loop" to use the benefits of machine supported medicine. Future work from our side includes the development of a prototype implementation in order to test the effects of introducing this concept into real-life environments.

Acknowledgments. Parts of this work have been carried out with the K1 COMET Competence Centers SBA Research and CBmed, which are funded by the Federal Ministry of Transport, Innovation and Technology (BMVIT); the Federal Ministry of Science, Research and Economy (BMWFW); Land Steiermark (Department 12, Business and Innovation); the Styrian Business Promotion Agency (SFG); and the Vienna Business Agency. The COMET program is executed by the FFG.

References

1. Smith, S.W.: Humans in the loop: Human-computer interaction and security. IEEE Security & Privacy, **1**, 75–79 (2003)
2. Lupiana, D.: Development of a framework to leverage knowledge management systems to improve security awareness (2008)
3. Clark, S., Goodspeed, T., Metzger, P., Wasserman, Z., Xu, K., Blaze, M.: Why (special agent) johnny (still) can't encrypt: A security analysis of the apco project 25 two-way radio system. In: USENIX Security Symposium, Citeseer (2011)
4. Kieseberg, P., Hobel, H., Schrittwieser, S., Weippl, E., Holzinger, A.: Protecting Anonymity in Data-Driven Biomedical Science. In: Holzinger, A., Jurisica, I. (eds.) Interactive Knowledge Discovery and Data Mining in Biomedical Informatics. LNCS, vol. 8401, pp. 301–316. Springer, Heidelberg (2014)
5. Randeree, E.: Secure health knowledge: Balancing security, privacy and access. In: AMCIS 2005 Proceedings, 287 (2005)
6. Warkentin, M., Johnston, A., Adams, A.: User interaction with healthcare information systems: Do healthcare professionals want to comply with hipaa? AMCIS 2006 Proceedings, 326 (2006)
7. Frühwirt, P., Kieseberg, P., Krombholz, K., Weippl, E.: Towards a forensic-aware database solution: Using a secured database replication protocol and transaction management for digital investigations. Digital Investigation **11**, 336–348 (2014)
8. Hobel, H., Schrittwieser, S., Kieseberg, P., Weippl, E.: (Anonymity and pseudonymity in data-driven science)
9. Heurix, J., Zimmermann, P., Neubauer, T., Fenz, S.: A taxonomy for privacy enhancing technologies. Computers & Security (2015)
10. Fruehwirt, P., Kieseberg, P., Schrittwieser, S., Huber, M., Weippl, E.: Innodb database forensics: Reconstructing data manipulation queries from redo logs. In: The Fifth International Workshop on Digital Forensics (WSDF) (2012)