Enabling trust in healthcare data exchange with a federated blockchain-based architecture

Mirko Koscina
mirko.koscina@ens.fr
DIENS, École normale supérieure, CNRS, PSL University
Paris, France

David Manset
david.manset@g2s-group.com
Almerys
Clermont-Ferrand, France

Claudia Negri
claudia.negri@almerys.com
Almerys
Clermont-Ferrand, France

Octavio Perez Kempner
operezkempner@di.ens.fr
DIENS, École normale supérieure, CNRS, PSL University
Paris, France

ABSTRACT

We propose a new healthcare data exchange platform for research centers, hospitals and healthcare institutions. Our model is based on a federated blockchain network that interconnect the healthcare institutions and orchestrate the data life cycle from the data publication to the data consumption. The blockchain is responsabible to keep the traceability of the whole process and we use a specially designed smart contract to control the data sharing process. Moreover, we provide the means to enforce GDPR and thus achieve a GDPR compliant model.

CCS CONCEPTS

• **Computer systems organization** → **Distributed architectures**; • **General and reference** → *Design*; • **Software and its engineering** → *Peer-to-peer architectures*; • **Theory of computation** → Cryptographic protocols.

KEYWORDS

health data, data exchange, blockchain, smart contracts, GDPR

ACM Reference Format:
Mirko Koscina, David Manset, Claudia Negri, and Octavio Perez Kempner. 2019. Enabling trust in healthcare data exchange with a federated blockchain-based architecture. In *IEEE/WIC/ACM International Conference on Web Intelligence (WI '19 Companion), October 14–17, 2019, Thessaloniki, Greece.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3358695.3360897

1 INTRODUCTION

In the health industry it is of utmost importance to constantly research about different topics. In this sense, the development of new techniques and treatments on diseases and epidemics strongly relies on gathering data from multiple sources and populations.

However, gathering data for research purposes can be an exhausting and daunting job for researchers. Obtaining the correct data and of good quality has high monetary costs and it is time consuming. Moreover, data gathering usually must also overcome ethical and methodological challenges, which further complicate the scenario.

It is crucial then to provide the means to lower the costs and fasten the process in a secure way in order to allow data exchange between research institutions and medical organizations. However, data exchange brings a wider range of challenges to the table.

From a socio-technical perspective, current data protection regulations and standards are pushing forward into more strict approaches to data management. Regulations such as the General Data Protection Regulation (GDPR) and the Health Insurance Portability and Accountability Act of 1996 (HIPAA) increases the responsibilities and accountability on data controllers and processors when processing health data. For instance, under Article 9 of the GDPR, "data concerning health" is defined as a special category, and thus must be further protected than normal personal identifiable information [1] which establish further responsibilities for the processing of health related data. Thus, exchanging data between actors cannot be tackled with a straightforward approach.

On the other hand, not having a system that is properly designed for data management (particularly for data exchange), can have huge business consequences. The average cost of a data breach per capita in the healthcare sector is of 408 USD [2]. Given that data in movement has a higher risk of being breached, this is indeed an important number to take into consideration. In addition, patients and data subjects have certain expectations about the management of their health data. Thus, in case of a data breaches, organisations that have suffered from them, can also see their reputation and trust affected.

As a consequence of the above situation, gathering consents from the patients to use their personal data has become more difficult. To overcome this problem, healthcare institutions need to improve their internal management processes and in particular those related to the data and consent management.

All in all, we have designed a model that can help in this direction and that would allow healthcare institutions to have access to data in a faster way, whilst also sharing the data in a secure and trusted manner. For that, our design seeks to make faster the data exchange process between the actors without loosing crucial properties like traceability and accountability. By improving the whole process, our design contributes to a better data governance within the healthcare

WI '19 Companion, October 14–17, 2019, Thessaloniki, Greece

ACM ISBN 978-1-4503-6988-6/19/10…$15.00
https://doi.org/10.1145/3358695.3360897

data exchange. The purpose of this work is to explain our model, its design and architectural decisions emphasizing on how they contribute to achieve such goals.

2 RELATED WORK

In [5] authors propose a blockchain model that ensures GDPR compliance and provide the means to carry out sensitive data sharing between participants of a network. This work generalizes work from [5] by clearly defining a compatible architecture and detailing its main components. As a result we obtain a specification that has been instantiated in the Horizon 2020 project MyHealthMyData[3] but also that is of it own interest of study.

To the best of our knowledge no other blockchain-based model for similar purposes can ensure GDPR compliance as our model can. For instance, Medicalchain [4] which uses blockchain technology to store health records does not provide information on weather or not their solution is GDPR compliant. In fact, the only reference to the GDPR in their whitepaper, when referring to data structures that are being used, states that: "These are subject to change depending upon different regulations and requirements in order to make the Medicalchain platform HIPPA and GDPR compliant".

3 ON THE APPLICABILITY OF A FEDERATED BLOCKCHAIN

In this section we will argue on the applicability of using a federated blockchain as the backbone of our model to enable trust in healthcare data exchange. We will first introduce the main ideas behind a blockchain.

A blockchain is a type of data structure which is replicated, shared, and synchronised between different participants that rely on a peer-to-peer network to connect with each other.

Roughly speaking, as a data structure, it is an append-only ledger organised as a chain of blocks where any modification in a block compels the regeneration of the following blocks in the chain. This is due to the fact that blocks in the chain link to its predecessor by a hash pointer. In addition, blockchains usually work with blocks timestamps to make it more difficult for an adversary to modify the chain. By following this design, the modification or deletion of any block can be extremely difficult to achieve.

It is replicated because participants hold their own local copy and it is synchronized because participants rely on a consensus mechanism to govern the management, updates and operations on the blockchain. As a result, parties can agree to a specific state of the blockchain as the valid one in a distributed manner (with no single entity in control).

With the above in mind, the following properties should be ensured.

- **Persistance [8]:** Once an honest party reports a transaction "deep enough"[1] in the ledger, then all other honest players will report it indefinitely whenever they are asked, and at exactly the same position in the ledger. This property is usually referred as the immutability of the blockchain.
- **Liveness [8]:** Transactions from honest parties will be included in the ledger of honest parties. This property roughly states that the blockchain will be able to process transactions comming from honest users.

It is worth to clarify what an honest party means in this context. To do so let us introduce a distinctive feature of a blockchain regarding the management of participants in its network.

- **Permissionless:** Usually the most well-known type of blockchains, where anyone can join and participate. The most famous examples are *Bitcoin* and *Ethereum*. In these blockchains no assumptions should be made at all regarding the honest behaviour of the parties. This means that some parties may arbitrarily decide not to follow the protocol in different ways to take advantage. In this scenarios, mechanisms to reward honest behaviour need to be triggered as a way to discourage malicious users from deviating from the protocol. This comes with a cost that can be reflected as a considerable overhead to the system.
- **Permissioned:** Participants in these blockchains can be fully identifiable and thus access can be granted or denied to them. Also, since the participants can be identified, permissioned blockchains can make use of different consensus protocols that cannot be used in the permisionless setting and benefit from it.

As the reader may find different terminology in the literature such as public or private blockchains, we would like to clarify a relevant aspect related to it. Public blockchains usually refer to permissionless blockchains whereas private may not be the analogous (permissioned). We consider private blockchains as those that are manage privately within a given organization and thus prefer the term federated to talk about permissioned blockchains that are not under the control of a single entity.

Thus, when we talk about federated blockchains, we refer to the fact that the permissioned blockchain in question is not governed by a single entity but rather by a federation or consortium of organizations. These organizations, in agreement, define the policies for its access control layer which will define which permissions are granted to which users with respect to the blockchain.

It is also worth to mention that the fact that participants in a permissioned blockchain can be fully identified does not guarantee that all of them will behave honestly. It says, at best, that there will be less incentives for them in deviating from the protocol (as they can be caught if doing so).

In some scenarios where parties that are known to each other need to cooperate or work together, but do not fully trust each other, a federated blockchain can help. Moreover, when it is the case of a cooperation that need to follow a business logic that cannot be afforded by the parties to be public, a federated blockchain may be more suitable.

The above scenario can be applied to sensible data exchange if we think in a consortium or federation of organizations that are willing to cooperate with each other but at the same time do not trust or cant afford the cost to trust in a single entity.

As an example we can mention projects like Hyperledger Fabric[9] which provide different tools within the Hyperledger framework to build and deploy enterprise blockchain solutions for similar

[1] If a transaction **tx** is commited in the block $n + k$ that is contained in a blockchain of $n + k + t$ blocks, t can be seen as the deep of the transaction **tx** with respect to the sate of the blockchain.

cases. Moreover Hyperledger Fabric works on the basis of plugable-consensus algorithms that can be plug in and out accordingly to the needs of the solution.

4 OVERALL ARCHITECTURE GOALS

In section III we presented the applicability of a federated blockchain in allowing participants of a network to agree and enforce a shared business logic to rule the helthcare data exchange process. In this section we will specify the main goals of our model taking the federated blockchain as a starting point to define such goals.

4.1 Process automation

As stated in section I a key aspect to be addressed is how to fasten the process of the data exchange in a secure way. Smart contracts can be helpful when it comes to process automation as we will discuss below.

First, let us recall that the idea of smart contracts is not new. They were originally proposed by Szabo in 1996 [7] to allow two or more parties to agree on a subject by delegating the management and enforcement of a contract to an application. Nonetheless, it was not until the rise of blockchain and decentralized applications that smart contract development flurished.

In 2015 Ethereum achieved for the first time the execution of Turing-complete code using a blockchain to that end. This in turns, made possible the original idea behind smart-contracts in terms of "autonomy" and gave rise to what we call today decentralized applications.

In a brief, the main idea behind smart contracts is to program rules and business logic so that they can be later on enforced independently of the participants interest. So once a smart-contract is deployed it will run and take actions solely based on the code that has been written to it. Ideally, under this paradigm, participants in the network would invoke a given smart contract by providing it with an input and retrieve the output but would not be able to interfere in the process. That is to say, a party would not be able to convince other party to accept an output which is different from the one defined by the smart contract logic given a fixed input. It is worth to mention that this posses challenges when dealing with non-deterministic computations, we refer the reader to [11] to further look into the perils of such a challenge.

With the above in mind, smart contracts result very appealing when it comes to defining different business logic among participants that may have a conflict of interest or that may try to take advantage over other participants who also rely on the same logic.

Accordingly to the Hyperledger fabric documentation, "smart contracts are not only a key mechanism for encapsulating information and keeping it simple across the network, they can also be written to allow participants to execute certain aspects of transactions automatically." [9]

Since the smart contract will "play by the rules by default", participants can trust their execution and thus process automation can be trusted by all the parties. This is what drives our first goal, to define in a secure and trustful manner a process automation to handle the data exchange process between parties. We will explain in the next sections how this can be achieved by using a smart contract.

4.2 Data traceability

We refer to data traceability as the process to know what type of data has been in the system, when and who used it, and under which conditions/purposes.

This idea of data traceability helps to improve the model of data governance. Overall, we can understand data governance as the data management and IT strategy by which organizations establish rules, policies, models and management of the data [12] [14]. More specifically, NIST has adopted the following definition for data governance:

"A set of processes that ensures that data assets are formally managed throughout the enterprise. A data governance model establishes authority and management and decision making parameters related to the data produced or managed by the enterprise" [14] [15].

To enable a proper data governance model, it is important to control data throughout its lifecycle. This implies, among others things, that data owners should know where the data is, who has used it, when and under which conditions. This item is particularly important for data exchange or sharing, as organisations should record to whom data has been shared with, for compliance with regulations, enable data subject's rights and security measures. Thus, having a good set of policies, rules, framework and systems that allows data owners to trace data, strength the data governance.

To accomplish this tasks, we introduce the following concepts:

- **Proof of existence**: It should be clear for any given data to be exchanged what is the exact content and under which conditions it can be shared. For healthcare data the relation between the data and the consent given to its usage need to be defined at the very early stage of the process. Thus it is important to be able to prove the existence of such relation at any given time. This in turns helps to know what type of data has been in the system.
- **Proof of matching**: To better provide the means to the when and who used which data, we introduce the notion of proof of matching to capture which and when the data has been used and by who. A proof of matching then implies that the party sharing the data can prove no only when that data has been used but also by whom.

4.3 Decentralization

One of biggest challenges of data governance in a centralized way, is that data management is carried out by a single entity. Therefore, there is a risk that records can be tampered or altered, either in a malicious or unintended way. This entails that data subjects must rely on the organisation's policies and trust that they are being enforced. A direct consequence for this model is that costs tend to be high due to all the work that needs to be done by the entity in charge of the data management.

Also data controllers may be reluctant to delegate the hole process of data exchange to a single entity even though they could afford the cost of it.

Decentralization in this scenario allows every party to keep the control of their own data and to participate in the decision making process of data exchange. Moreover, by distributing the responsibility of the process among parties there will be less chances for

the information to be tampered as long as most of the participants remain honest.

4.4 Auditability and GDPR compliance

Auditability is a key component in data security which in turns can help to achieve GDPR compliance. Given the traceability feature of the system, it is possible to examine and systematically review the data management. Even more, given the design of the network, it is possible to achieve the desired granularity in the retrievable process, with the possibility of knowing specifically where each data set has gone.

Due to the immutability principle of the blockchain, it is possible to audit the data trails and data management with assurance that the database has not been tampered.

One of the interesting aspects about this design, is that the "right to access" [13] enabled by the GDPR could be accomplish by the model. If a given user wants to carry out its own right of knowing where its data has been and to whom it was shared with, a positive answer should be delivered to him. This is what we seek by setting this goal, to take the immutability and decentralized properties of the system to ensure that every node in the network can be able to trace its own data. Moreover, some other rights such as the right to be forgotten should be also subject to exercise in order to be fully GDPR compliant.

4.5 Enable trust

One major concern when designing such a system is to provide a secure way to transfer and exchange data between the parties. In this sense, in order to comply with data protection regulations and possible malicious actors, it is crucial to preserve confidentiality and integrity of the data. The model should also assure that entities are authenticated and part of the network to clearly identify every action.

The aforementioned properties of the system are important for data exchange for three main issues: data protection, regulatory compliance and to enable trust. The first two, data protection and regulatory compliance, are tightly related as we seen before when referring to the GDPR.

Furthermore, it is also important for any organisation handling personal data, to have trust with their data subjects. Given that health data is considered a sensitive (or special category), special attention must be given to security when data is in movement. If a data breach occurs, as it has been noted before, reputational harm can be done to the organisation. Thus, our goal in this sense is to prevent as much as possible data leakages. Which is also another reason for enhancing data traceability.

5 OUR MODEL

Even though the fedareted blockchain is a central component in our model, it is certainly not the only one. In this section we will dive into each of the components explaining their main functionalities and how these contribute towards achieving the goals that we previously defined.

Let us first start by saying that under this model different organizations agree on participating together in the same network. As an example one can imagine a network composed by two research centers and one hospital.

Following this example, every organization will have their own users and every user will be recognized as a valid one by other organizations. This means that organizations have control on their own users and credentials issued to them are recognized by any other member of the network.

It is important to highlight that every organization runs at least one node in the blockchain network and thus the governance of the federated blockchain is distributed among the organizations.

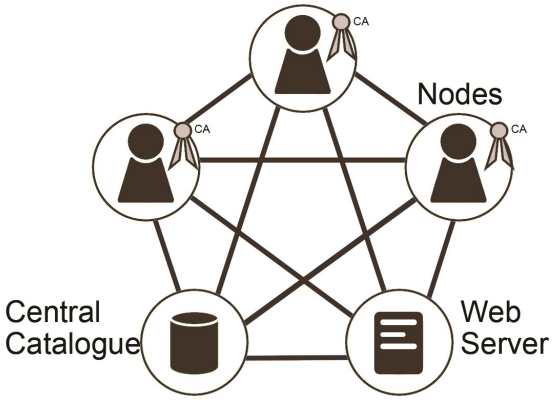

Figure 1: Model representation

5.1 Central web server

We define a central web server as the users endpoint to communicate and interact with the hole system. A user belonging to one of the organizations will use this web server to carry out all of its requests and interaction with the network.

It is worth noticing that introducing a component of centralization in this architecture does not prevent the goals we previously defined related to the decentralization since the web server will be an access point but will not hold any sensitive data (other than information related to users authentication).

The idea of the central web server is to facilitate the interaction with the system and with the central catalogue (which will be introduced next).

Since we have a federated blockchain network, authentication needs to be handled differently. We propose a single sign-on approach where users authenticate with the web server specifying username, password and organization so that the web server can validate these credentials with the corresponding organization. This in turns means that if one of the organizations experiences problems, only users from that organization would not be able to use the system.

5.2 Catalogues

In order to create a data request a user needs to know which data is available in the system so he can ask for it. To manage the data available in the system we make use of local catalogues held at every data controller and one central catalogue.

Local catalogues index the data that will be available to the network from the data sources. This indexation process includes a normalization of the data (the generation of metadata from the actual data) and the registration of the metadata in the central catalogue.

The central catalogue is just an aggregated version of the available data which enables users in the system to browse for data in a consolidated way. We rely on metadata to describe the available data in order to avoid direct links on the actual data. We should stress that is also desirable to require data controllers to perform at least one level of anonymization to the data before sharing it.

This way, when a user browse the metadata from the central catalogue it will know which type of data is available but not who is providing it.

5.3 Certificate authorities

From the authentication point of view, every organization has its own Certificate Authority with an API (CA API) and a second API (Authentication API) that is integrated with the web server to handle the authentication with the users. This Authentication API is consumed by the web server to carry on the single sing-on process and in turns will consume the CA API to perform the right mapping between user's credentials and associated public keys. This model can be easily deployed with Hyperledger Fabric for example.

By managing certificate authorities we can authenticate any action performed in the system. As for the blockchain, every CA will generate certificates for its users and for the TLS communication between the nodes. Also, properties such as non-repudiation and data integrity can be easily provided as well.

5.4 Network nodes

We define a network node by the following components:

- A blockchain peer, eventually with more than one peer running in the same node for performance purposes.
- A driver (backend application) which implements the main business logic. It process requests from the central web server and communicates with the local peers to execute and process transactions.
- A local mapping database that is used to keep track of the references that every data item has in the blockchain. It links a data item with every data request in which it was used.
- A certificate authority (CA) responsible of issuing certificates within the organization.
- An authentication API which communicates internally with the CA and externally with the central web server. The idea behind this is that the authentication API connects to the CA to generate a user and store it locally. It then provide the means to the central web server to authenticate users following the single sign-on approach.

Two operation modes for the driver are defined, with data sources and without data sources. Organizations can process and request data but not necessarily every organization on the network will process data. This means that some organizations may will only ask for data but never share data. This is the case of pharmaceuticals for example. In this case, as the organization will not be sharing data, the driver does not need to communicate with a local catalogue or data sources adapters. Depending on the operation mode of the driver a network node may have other services running or not.

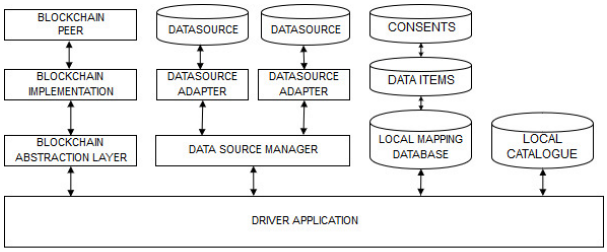

Figure 2: Logic abstraction of the driver

6 DATA EXCHANGE FLOW

Our defined goal on process automation introduced the idea of delegating the administration of the data exchange flow to a smart contract. In this section we list the steps that we defined for the data exchange flow and how they interact with the logic established by a smart contract that we call ExchangeFlow.

As a smart contract, ExchangeFlow provides the following methods that can be invoked and manage different assets.

- createDataRequest: allows the user to define a new asset in the system and record information on the blockchain regarding a new data request.
- query: allows the user to query on the assets which exists on the system.
- registerData: allows the user to define a new asset in the system and record information on the blockchain regarding new registration of data.
- registerResponse: allows the user to define a new asset which contains a response for a data request that was previously defined in the system. This assets are later on retrieved by the data requestor to process the request.
- updateDataRequest: allows the user to update a data request asset in order to reflect wheather or not all the responses have been sent. This facilitates the processing of closing the request and delivering the results to the end user.

In order to better explain the details of the step by step process, let us first stress the following related to the actual way in which the exchange is performed. A session key is generated every time that data items are shared. This session key is used to encrypt those data items and so it need to be shared with the data requester. To share the session key, a public key (which we introduce in step 2) is be used to encrypt it so that only the data requester is able to obtain it. The data requester will download the data items (which are encrypted) and will have to decrypt them with the corresponding session key.

We detail below the step by step process which consists of the following steps: 1) Data registration, 2) Data request, 3) Data fetch and 4) Data response.

6.1 Data registration

The driver is used to publish data to the network. This process, called data registration, consists of several steps that we will describe in detail.

Recall that a driver has its own local catalogue and data sources adapters. The adapters are used to fetch the data from the data sources. As data can be heterogeneous and thus come from different sources we define such adapters to achieve a modular design. Every datasource adapter implements the same interface. Thus, adding a new datasource boils down to implementing the datasource adapter interface for that datasource.

Different datasource adapters are handle altogether from a datasource manager which communicates back and forth with the driver application.

When a datasource recieves new data, its datasource adapter communicates with the datasource manager which in turns communicates with the local catalogue to index the data.

Once data is indexed it needs to be registered so it is available to the network.

To register the data, the driver invokes ExchangeFlow.registerData which in turns creates an asset attesting the registration of the data in the blockchain. This works as a **proof of existence** and allows the driver to later prove that it had available data with a given consent by the time of the registration.

Note that in order to register data the consent needs to be specified but most importantly the consent will need to be checked before sharing data. In this sense, the consent management can be seen as the process of checking that these two steps are being done correctly (i.e a consent is defined and properly checked for every data item).

In order to register the data the local mapping database is used. The content of this database are pairs (key,value) where the key is a unique reference for every data item and value is a set of references from that data item to the blockchain. In the value field, a tuple (blockchainTransactionId, offset, consent, hash,hmac) is saved every time a data item is registered with a consent. Below we explain the differents fields for the tuples in value:

- blockchainTransactionId: indicates the transaction in the blockchain where the data item has been registered.
- offset: is an integer assigned during the invoke of ExchangeFlow.registerData. A global counter is used to assign a unique value to every data item.
- hash: is the cryptographic hash of the data item to make sure that the data has not changed.
- hmac: is a key-hash message authentication code of the data item hash, its unique id and the consent. This allows the driver to prove data integrity and authentication for the data item and its consent.

It is important to recall that such mapping is entirely in the driver of the data controller and thus only that party, which is the intended one, would be able to respond in case of an audit. For the rest of the parties the information recorded on the blockchain will be useless without the references at the local mapping.

6.2 Data request

We now describe the data request process. A user logged into the system can use the central web server to browse data available from the central catalogue. At this point the user knows which type of data is available for request but does not know who has such data. Also, the user knows which types of consents are compatible for the available data as this was previously specified during the data registration process.

Once that the user defined the type of data that he will request, the next step is to communicate such decision to the network. At this point, the user specified how he is going to use the data and for which purposes. Both need to compatible with the consents defined by the data registrants, otherwise they will not carry out the exchange.

With the above information the central web server communicates to the driver which belongs to the user organization and forwards the request.

The driver process the request by invoking ExchangeFlow.dataRequest specifying the following information:

- dataRequestId: a unique id assigned globally to every dataRequest when before creating the asset.
- dataRequestInfo: general information which includes the purpose of the request as well as the requested consent.
- publicKey: a public key is created per data request to allow all the entities to privately communicate with the requester when sending their responses.
- status: a field indicating the status of the data request that will be updated once that all the responses have been collected. This is field is also used to notify the user in case of problems arising from the data exchange process.

Finally, a corresponding transaction reflecting these changes is added to the blockchain. This will allow all the networks nodes to be notified on a new data request when reading the updates from the blockchain.

6.3 Data fetch

Every driver upon notification of a data request checks internally if it has available data to exchange. As the data request specifies the conditions on the data usage and the required consent, every driver can check and decide weather or not share the data.

If the answer is positive the data needs to be fetched from the data sources and a data package needs to be build from the available data. We call this process data fetch.

6.4 Data response

Data responses can be positive or negative. In order to response an invoke to ExchangeFlow.registerResponse has to be made specifing the follow fields to create the corresponding asset:

- dataRequestId: the id of the data request it is responding to.
- result: indicates if it is a positive answer or negative answer. If it is negative the rest of the fields can be ignored.
- info: indicates general information on the response. For example, the hashes of the data items to be shared so they can be checked afterwards.

- privateField: a session key to decrypt the data items being exchanged is encrypted with the publicKey obtained from the data request and sent in this field.
- downloadLink: a link to download the data items.
- bitmapHash: the hash of the bitmap for the resultset (set of data items to deliver). A data request would normally use less data items than the ones that where registered by the driver. The driver calculates and stores a compressed bitmap from the offset of the data items involved in the request response. Thus, bitmaps are used to establish the link between data items that have been used by a data request for every data request. As a **proof of maching** the driver will save the bitmap hash. This allows it later to prove that a given set of data items have been used in a given data request. It is worth to notice that since the bitmaps are stored locally on each driver only the driver who stores the bitmap can know where check where the data items have been used.

It is important to notice that the actual data exchange is carried off-chain and no direct information on the actual data is saved on the blockchain.

The data requestor will read the responses from the blockchain and process them one by one downloading the data sets and decrypting them with the corresponding session key. Finally it will invoke ExchangeFlow.updateDataRequest to end the data request.

Information related to the whole process can be later checked and retrieved by invoking ExchangeFlow.query on the corresponding assets.

7 CONCLUSIONS AND FURTHER WORK

In this paper we presented a novel approach to empower organizations in the decision making process of exchanging healthcare data while avoiding the need to rely on a third party. As said before, this can be expensive from the monetary point of view and also from the point of view of data breaches. Under this model, every institution is responsible of its own data and the actions it takes with it and thus can apply its own policies. Moreover they can easily trace to whom they deliver the data and under which conditions. From the data requester perspective, they are forced to declare in advance such conditions so that the rules of the exchange remain transparent for the involved parties and auditing the process becomes easier.

Process automation in this sense makes it also easier to gather data in a faster way which in turns can boost its usage in fields like health data analysis and mining.

Furthermore, to the best of our knowledge, our model is the first GDPR compliant blockchain-based solution in the health data exchange domain.

As future work, integration with novel approaches to enhance privacy (e.g developing techniques in privacy preserving smart contracts, secure multi-party computation and zero-knowledge) would allow the extension of this model to different domains.

ACKNOWLEDGEMENTS

This project is funded by the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No. 732907.

REFERENCES

[1] European Parliament and Council EU repealing Directive 95/46/Ec of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the L 119/1.
[2] Ponemon Institute (2018) 2018 Cost of a Data Breach Study:Global Overview. Available at https://www.ibm.com/security/data-breach (Accessed July 2019)
[3] http://www.myhealthmydata.eu/ (Accessed July 2019)
[4] Medicalchain Whitepaper v2.1. Available at https://medicalchain.com/Medicalchain-Whitepaper-EN.pdf (Accessed July 2019)
[5] Bayle, A. Koscina, M. Manset, M. Perez-Kempner, O. (2018) When Blockchain Meets the Right to Be Forgotten: Technology versus Law, *Healthcare Industry WI* pp.788-792.
[6] Nakamoto, S. *Bitcoin: A peer-to-peer electronic cash system*, 2008.
[7] Szabo N. (1996) Smart Contracts: Building Block for Digital Markets. Avaialable at http://www.fon.hum.uva.nl/rob/Courses/InformationInSpeech/CDROM/Literature/LOTwinterschool2006/szabo.best.vwh.net/smart_contracts_2.html (Accessed: July 2019)
[8] Garay J., Kiayias A. and Leonardos N. (2015) The Bitcoin Backbone Protocol: Analysis and Applications. Advances in Cryptology *EUROCRYPT 34th Annual International Conference on the Theory and Applications of Cryptographic Techniques* pp.281-310.
[9] Hyperledger (2019). Introduction. Available at https://hyperledger-fabric.readthedocs.io/en/release-1.4/blockchain.html (Accessed July 2019)
[10] Pramod J. Sadalage Martin Fowler. NoSQL Distilled: A brief Guide to the Emerging World of Polyglot Persistence. Addison-Wesley, 2013.
[11] Androulaki E, et. al. Hyperledger fabric: a distributed operating system for permissioned blockchains. In Proceedings of the Thirteenth EuroSys Conference (EuroSys '18). 2018. ACM, New York, NY, USA, Article 30, 15 pages.
[12] IBM (no date) Data Governance. Available at https://www.ibm.com/analytics/data-governance (Accessed July 2019)
[13] Regulation (EU) 2016/679. Available at https://eur-lex.europa.eu/legal-content/EN/TXT/PDF/?uri=CELEX:32016R0679 (Accessed July 2019)
[14] Committe on National Security System, (2015) Committee on National Security Systems (CNSS) Glossary - CNSSI No. 4009
[15] National Institute of Standards and Technology NIST (no date) Glossary - Data governance. Available at https://csrc.nist.gov/glossary/term/data-governance (Accessed July 2019)