Optimized and Dynamic KYC System Based on Blockchain Technology

José Parra-Moyano

University of Zurich

Tryggvi Thoroddsen

University of Zurich

Omri Ross

University of Copenhagen

Abstract: Systems that use blockchain technology to improve the know-your-customer (KYC) process have only been proposed at a conceptual level and all share certain attributes that make their adoption by financial institutions (FIs) very difficult. We propose and program a blockchain-based system that reduces and shares out among the financial institutions that work with a customer the costs of the KYC process and also makes it possible for FIs to dynamically update information related to customers and disseminates this information among participating FIs. Additionally, our system addresses some of the attributes that hinder the adoption of previously proposed solutions by FIs. The result is a programmed, stand-alone solution that can be implemented by FIs to reduce the cost of the KYC process without requiring any central instance to store the customer's data, and in which FIs share the initial costs of the KYC process as well as the running costs of keeping the information about customers up to date. Our system increases the levels of security and regulatory compliance in the KYC process and significantly reduces the cost of that process for all parties involved.

Keywords: Blockchain, KYC, AML, Distributed Ledger Technology

1 Introduction

The know-your-customer (KYC) process that financial institutions (FIs) are obliged to follow whenever they establish a financial relationship with a new customer represents a significant financial burden for FIs but creates no productive added value. The KYC process is made up of a series of routine tasks that, when carried out, are meant to verify the lawfulness of a potential customer's activities. Every FI needs to follow the KYC process before even starting to work with a new customer. The cost of KYC is rising. Thompson Reuters (2017) estimates that on average large financial institutions with turnovers in excess of USD 10 billion increased their annual spending related to KYC obligations from USD 142 million to USD 150 million during 2016. The same report contains the prediction that spending on KYC-related tasks would increase by 11 percent over the 12 months following its publication. According to Thompson Reuters (2017), corporate customers work on average with 11 FIs, which implies that this—costly—KYC process is repeated on average eleven times for each corporate customer. The average time an FI takes to "onboard" a corporate customer is 26 working days.

The increasingly widespread use of blockchain technology has led to the development of new systems that are meant to improve the efficiency of the KYC process and to enable cooperation among FIs. If achieved, both these goals will lower the costs of KYC. Parra-Moyano and Ross (2017) were the first to suggest that the KYC process should be conducted only by the first FI that wishes to work with a given customer, and that the result of conducting the process (proof of the lawfulness of that customer's activities and of the customer's "validation") should be shared in an anonymized and secure form with all FIs that subsequently wish to establish a financial relationship with that customer. The system proposed by Parra-Moyano and Ross (2017) also includes a structure that distributes, proportionately, those KYC costs initially borne by the first FI in order to work with a given customer among all the FIs that subsequently work with that customer, including that first FI. While their proposals constitute an innovative approach to reducing the costs of KYC, they involve certain inherent inefficiencies that make their implementation in a corporate environment difficult. These include the need for a trusted third party (TTP) to store the customers' data and carry out the financial compensations between FIs and that no updates or changes in the information status of a customer are possible. Parra-Moyano and Ross (2017) also only develop their proposed system on the conceptual level.

Our aims with the present paper are to review the work carried out by Parra-Moyano and Ross (2017) as well as a range of the other blockchain-based systems thus far proposed as ways of improving the KYC process and to tackle the open issues that make the implementation of these systems in the corporate environment difficult. This enables us to suggest a system that can be realistically implemented in the financial sector and to develop a stand-alone proof of concept (PoC) of the system that can be used as a foundation from which corporations and regulators will be able to explore and—eventually—conduct the implementation of blockchain-based KYC solutions. In Section 2 we briefly describe the current KYC process and the requirements it must fulfill. In Section 3 we provide a brief introduction to blockchain technology and innovations in the architecture of distributed databases, focusing on the attributes of these two technologies that make the system that we propose possible. In Section 4 we analyze those systems that claim to improve—by using blockchain technology—KYC as it currently stands. In Section 5 we describe how we used design science research (DSR) to refine these previously proposed systems and to derive our PoC. In Section 6 we describe, from a non-technical perspective, the refined system that we propose, and in Section 7 we present the code that yields the PoC. In Section 8 we conclude.

2 Current KYC system

In recent years, the KYC due diligence process has evolved from a simple formality into a thorough process supervised by national institutions. The Financial Action Task Force (FATF)—an intergovernmental body established to combat money laundering and the funding of terrorism—sets the international standard for KYC. That standard is outlined in the FATF Recommendations (The Financial Action Task Force, 2012-2017), a document that was first published in 2012 and was updated in November 2017. We can paraphrase the FATF Recommendations' minimum requirements for FIs conducting the KYC process as follows:

1) Identify the customer and verify that customer's identity using reliable, independent source documents, data, or information.

2) Identify the "beneficial owner", verify the beneficial owner's identity, and understand the ownership and control structure of the customer.

3) Understand and obtain information on the purpose and intended nature of the business relationship.

4) Conduct ongoing due diligence on the business relationship throughout the course of the relationship to ensure that the transactions being conducted are consistent with the FI's knowledge of the customer.

The first three of these requirements must be met by each FI before it establishes a financial relationship with a new customer. Thus, if one customer works (or intends to work) simultaneously with *n* FIs, the KYC process for that customer will be repeated *n* times. Although each FI is responsible for its own KYC process and must conduct due diligence independently of other FIs, a core portion of KYC due diligence—namely, points 1, 2, and 4 in the above list—is a routine process that is carried out in parallel by all FIs that work (or intend to work) with the same customer. Thus, costly tasks are carried out repeatedly and in parallel whenever a customer works with two or more FIs. Figure 1, which we derive from Parra-Moyano and Ross (2018), schematically describes the current scenario.

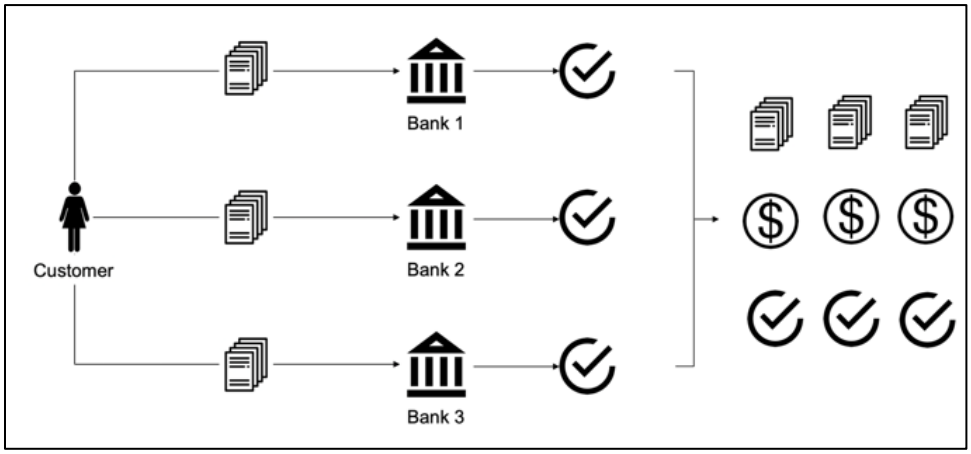

Figure 1. Current KYC Scenario (adapted from Parra-Moyano & Ross, 2017)

3 Blockchain and distributed database architecture

This section briefly introduces the two technologies that we use as basis for the solution that we propose in Section 5—namely, blockchain technology, which was introduced by

Nakamoto (2008), and the private distribution of data across distributed databases, introduced by Siegenthaler and Birman (2009a and 2009b).

3.1 Blockchain technology

Blockchain technology offers a global, distributed transactional database in which nodes are linked to one another by a peer-to-peer (P2P) communication network with an own layer of protocol messages for node communication. Users of a blockchain can reference one another using their respective public keys and can use their private keys to cryptographically sign messages and transactions (Glaser, 2017). Although blockchain technology has gained notoriety primarily due to the advent of crypto-currencies, such as Bitcoin, which was introduced by Nakamoto (2008), researchers and practitioners are applying it in a number of ways to improve existing information systems (IS) and make them more efficient.

One of the ways in which blockchain technology contributes to the improvement of IS is that it makes the execution of "smart contracts" by any node that has access to the blockchain possible. Smart contracts are computer protocols that facilitate, verify, or enforce predefined clauses whenever a given set of conditions is met (Parra-Moyano & Ross, 2017). Once a smart contract is triggered, it can carry out automatized, predefined actions. Currently, the three largest distributed ledger platforms that facilitate smart contracts are Ethereum, from the Ethereum Foundation; Hyperledger, from IBM; and Corda from R3. Because, in a blockchain, a copy of the ledger is distributed to each node, there is no need for a TTP to act as a notary with regard to processes that involve participating nodes that do not trust one another.

The validity of the information stored on a blockchain's ledgers is ensured by the network's nodes with the help of a secure hash algorithm (SHA). Blockchain technology uses an SHA to translate the contents of a block into a cryptographic fingerprint referred to as a "hash". An SHA can also be used to generate from a digital document a unique "fingerprint" of that document, such that this fingerprint cannot be replicated unless it is generated from the exact same document. This ensures that all of a blockchain's participants can easily verify the authenticity of any document previously hashed simply by hashing it again and comparing the hash they generate to the hash that was previously generated using the authentic document. Further, the hash does not reveal any information about the contents of a document, just as analyzing a human fingerprint can help one to prove the identity of an individual but fails to reveal—for example—the features of that individual's face. In a distributed ledger with multiple nodes, the information recorded by the network is stored sequentially in a list of records that is divided into blocks and distributed to all nodes on the network. The information in each individual block is then used by the system's protocol to generate a secure hash that identifies that specific block. Each subsequent block records the hash of the previous block such that all blocks are chained together sequentially making it impossible to change information in one block without changing all previous blocks. If one node alters the information on its ledger and tries to interact with the network using what is, thus, "false" information, the hash will no longer match the ledger distributed to the other nodes on the network and the transactions that this node attempts to conduct will not be accepted by these other nodes. The process of verifying transactions and ensuring that blocks have not been altered is carried out by the nodes of the network.

Blockchains are, most commonly, either "public" or "permissioned". A permissioned blockchain limits the number of nodes that can access it or that can approve the hashes that are to be saved on the ledger. A public blockchain, meanwhile, has an unlimited number of nodes and is accessible to all.

In order for it to be maintained, a blockchain requires a protocol that defines the roles and rules that apply on it. The many protocols that can be implemented in blockchain technology include proof of work (PoW), proof of stake (PoS), and proof of authority (PoA). PoW incentivizes the participating nodes to spend computational power (work) and write new blocks. The fact that spending computational power is costly means that rewriting the blockchain is expensive; this secures the blockchain against fraudulent attacks, which indeed need to rewrite the blockchain in order to be successful. To compensate "miners" (nodes that verify transactions in all kinds of mineable blockchains), the protocol provides a reward in the form of crypto-currency to the first miner that writes a valid block. The PoS protocol relies on a smart contract that holds deposits—of a crypto-currency—made by nodes that wish to act as miners. The node that supplies the largest amount of crypto-currency is assigned the authority to mine by the blockchain's owner. Once a node has been granted this authority it no longer needs to rely on computational power to be allowed to mine. The PoA protocol defines that pre-authorized nodes act as miners and add blocks to the blockchain. Instead of using hash power to write valid blocks—as is the case with PoW—or providing funds in order to be granted the right to mine, PoA nodes are able to add blocks to the blockchain at any time.

3.2 Private information sharing across distributed databases

Siegenthaler and Birman (2009a and 2009b) introduce a database architecture that allows the electronic sharing of privacy-sensitive data across distinct nodes and respects very high privacy standards. This architecture was constructed to allow hospitals—holders of patients' medical data—to share patient data with each other to improve patients' treatment and respects three privacy principles that ensure that only the necessary information about patients is shared and that hospitals making and receiving queries do not learn anything about one another that can reveal information about the patients that is sensitive or is irrelevant to a patient's treatment. Specifically, the database architecture that they introduce allows entities to store different pieces of data such that the following three principles are respected:

1) Data privacy. Queriers learn only the answer to their query, not any of the data used to compute that answer.

2) Query privacy. The data owner does not learn the particulars of the query, only that a query was performed against a particular patient's information.

3) Anonymous communication. Neither queriers nor data owners know who the opposite party is.

For this architecture to work, the database schemas of the hospitals are irrelevant; it is sufficient for the data producers to provide a read-only API to the members of the network. This system allows data to be shared between nodes in a secure and encrypted manner.

This system does, however, require a TTP to manage both access and the right to perform queries against other nodes' databases.

4 Previously proposed KYC systems based on blockchain technology

Since the KYC process—whether observed from a national or an international perspective—is characterized by many duplicated tasks carried out by agents that do not trust one another, it seems that blockchain technology may have significant promise when it comes to reducing inefficiencies and costs and to offering a more efficient structure under which to conduct KYC. It comes as little surprise then that a number of bodies and organizations have suggested various approaches to using distributed ledger technology (DLT) to improve KYC systems.

Parra-Moyano and Ross (2017) propose a blockchain-based system that improves the efficiency and reduces the cost of the KYC customer-onboarding process. Their system is meant to be run by a national regulator, which provides and maintains the system's physical and operational structure. In this system, the KYC process is carried out only once, by the first FI that is approached by a customer. When that customer approaches another FI with the aim of establishing a financial relationship, this second FI can see—by consulting the blockchain—that the KYC process has already been carried out (in this case by the first FI) and can thus focus solely on certain, limited aspects of KYC (namely, understanding the customer's activities) and does not need to perform routine, mechanical document verification. Further, the system includes a mechanism that distributes the costs of conducting the KYC process proportionally among all participating FIs that work with a given customer. While this is the most comprehensive work on blockchain-based KYC systems that the literature currently contains, it possesses a series of characteristics that hinder its implementation in the corporate environment and therefore needs to be improved upon if it is to become truly useful for FIs and regulators. The first aspect that needs to be improved is the fact that in Parra-Moyano and Ross' (2017) solution the TTP must periodically check that the FIs in the system have paid the proportion of the cost that they are meant to have paid and have not simply used the system to verify that the documents presented to them by any given customer have previously been validated. The second aspect that requires improvement is the fact that in the system proposed by Parra-Moyano and Ross (2017) the status of a customer cannot be updated by an FI in a decentralized way; rather, the KYC onboarding process is only conducted once for each customer and by one FI only, and the system does not envisage the potential need for periodic updates with regard to a customer. Thus, if a customer is validated by the first FI but its status later needs to be changed due to—for example—irregularities in its activities, information with regard to the customer's new status cannot be disseminated using the system to all those FIs that work with that customer. A third aspect that is susceptible to improvement is that the proposed system is unable to make dynamic compensations between participating FIs. Such a dynamic compensation system is required in order to allow for financial compensations among the FIs participating in the system over time, specifically whenever an FI needs to update a customer's KYC status or history. A fourth aspect that could be improved concerns the storage of customers' documents. Parra-Moyano and Ross (2017) propose a complex database architecture in which customers need to store these data privately and circulate them among the FIs with which they want to work. Such a structure is costly, and it becomes clear—when one compares the customer journey that emerges

from this structure with the existing customer journey—that the self-storage aspect would be a disadvantage.

The R3 Project, run by a consortium of banks, conducts applied research on blockchain applications. R3 runs Corda, an open-source distributed ledger platform designed to record, manage, and automate legal agreements between businesses. The Corda network is made up of nodes, where each node represents a JVM run-time environment hosting Corda services and executing applications, or "CorDapps". CorDapps are participant applications that execute contract code and communicate using a flow framework to achieve consensus over some given business activity. While there already exist CorDapps for asset trading (IRS Demo and Trader Demo), as well as for portfolio valuation (see SIMM and Portfolio Demo—also known as the Initial Margin Agreement Demo), there does not yet exist a functioning CorDapp for KYC/AML. Rutter (2018) describes the benefits of decentralizing the KYC process and provides a conceptual description and comparison of two different decentralized scenarios run on Corda—namely, the "Self-Sovereign Model" and the "Bank Sharing Model". In the self-sovereign model corporate customers create and manage their own identities and relevant documentation, granting permission to multiple participants to access this data whenever they require it. In such a system, the relationship remains one of customer to bank, with the rights and responsibilities of each laid out in a contract and the bank not necessarily storing any of the customer's data. Instead, the customer permits the release of their data to each individual bank.

The use of blockchain technology in the development of digital identity has also proved promising. Blockchain technology can be used to register and store the credentials and ID-related information of users and can act as a TTP, verifying users' identities (Shocard, 2017; and Civic, 2017). These types of digital identity blockchain solutions may have important implications for the improvement of the KYC process.

Britton (2016) briefly analyzes the potential uses of blockchain technology in the context of KYC/anti-money laundering (AML) measures. While he considers that the KYC framework is probably one of the most suitable for the application of blockchain technology, he also addresses the difficulties of realistically establishing such a system. Specifically, he states that while there is immense potential in the application of blockchain technology for KYC, there are certain challenges that need to be addressed if one is to create a viable proposition that can be adopted by the industry. He pays special attention to the network effect that must be present in a valid blockchain-based KYC solution and claims that it could only result from collaboration among market participants working toward a mutually beneficial solution that would enable them all to focus on the customer.

The Hong Kong Monetary Authority (2016) has studied the potential benefits of blockchain technology for the financial sector. The authors of the study conclude that the technology offers the potential for banks to share identity information in an effective and secure manner, such that digitized customer records and documents could be shared among banks using a blockchain-based platform. The Authority specifically states that such an arrangement would offer a number of benefits. First, customers would no longer need to repeat the same processes and submit the same personal information to different banks for KYC purposes; second, the costs incurred due to and the resources needed for the identity-verification process would both decrease because the information in question would be readily accessible and shared via the blockchain; third, the checking of customers' history

could be carried out efficiently by participating banks because customers' information would be available in the blockchain; and fourth, a better customer experience would result. The authors of the study also state that existing KYC requirements and customer-authentication processes are manually intensive and require significant resources from FIs that seek to be compliant. The authors do not, however, propose a specific design for such a system.

5 Research methodology: design science research The aim of design science research is to extend the boundaries of human and organizational capabilities by creating new and innovative artifacts (Hevner, 2004).

Since in this paper we aim to propose an optimized, blockchain-technology-based KYC artifact that enhances the capabilities of FIs, design science research is an appropriate method for our purpose. Well-founded design science research, according to Hevner (2004), follows seven guidelines and results in a technology-based solution that solves a relevant business problem. These guidelines are depicted in Table 1.

Guideline	Description
1. Design as Artifact	Design science research must produce a viable artifact in the form of a construct, a model, a method, or an instantiation.
2. Problem Relevance	The objective of design science research is to develop technology-based solutions to important and relevant business problems.
3. Design Evaluation	The utility, quality, and efficacy of a design artifact must be rigorously demonstrated via well-executed evaluation methods.
4. Research Contributions	Effective design science research must provide clear and verifiable contributions in the areas of the design artifact, design foundations, and/or design methodologies.
5. Research Rigor	Design science research relies upon the application of rigorous methods in both the construction and the evaluation of the design artifact.
6. Design as a Search Process	The search for an effective artifact requires using available means to reach desired ends while satisfying laws in the problem environment.
7. Communication of Research	Design science research must be presented effectively both to technology-oriented and to management-oriented audiences.

Table 1: Guidelines for DSR (adapted from Hevner, 2004)

We rigorously follow these seven guidelines in order to propose an optimized KYC system. Because in this paper we propose a viable, technology-based artifact in the form of an instantiation that solves the problem of the high cost of KYC we follow guidelines one and two. We demonstrate the utility, quality, and efficacy of the artifact by following a well-executed process—that proposed by Peffers, Tuunanen, Rothenberger, and Chatterjee (2007), which is similar to the procedure followed by Parra-Moyano and Ross (2017)—and thus follow guideline three. Specifically, the process is divided into five steps, as illustrated in Figure 2: identifying the problem, defining the objectives, designing and refining the artifact, demonstrating the artifact, and evaluating the artifact. We recursively repeated the last three steps of the process to carry out a vigorous evaluation of the artifact's design.

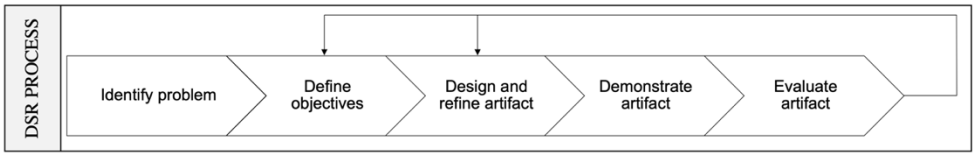

Figure 2. DSR Process (adapted from Peffers et al., 2007)

In following guideline four, we provide not only a blockchain-technology-based KYC solution that solves the open issues of all the previously published blockchain-technology-based KYC solutions, but also a programmed, functioning artifact that can serve as the foundation upon which other researchers and practitioners may develop the artifact further. Programing such an artifact has never been done before. To ensure our adherence to guidelines five and six we collaborated for seven months with blockchain and KYC experts from Origo, Iceland's leading systems integrator and managed service provider. Origo's experts not only systematically evaluated the progress made by our research, they also contributed—with valuable insights—to the design and the coding of the artifact. We communicate our results in a manner that is in line with prevailing academic style and language, but one that also renders the proposed solution accessible to practitioners, as is dictated by guideline seven.

Previous research has identified the problem of the increase in KYC costs for FIs, as we have explained in the previous section. But while researching those systems that have been proposed explicitly to improve the KYC process by applying blockchain technology we failed to identify even one single academic publication that addressed both the theoretical conception of a blockchain-based KYC solution and its technical development in the form of a PoC. Further, during our research we identified a series of inefficiencies in the previously proposed conceptual solutions. We embodied all these problems in the definition of the following research question: "*Can we conceptually develop a blockchain-based solution that improves the KYC process for financial institutions and that solves the open issues of previously suggested systems, while programming it and converting it into a functioning, verifiable, replicable instantiation*?"

With the problem(s) identified and our objective—embodied in our research question—defined, we recursively undertook the three remaining steps of the process proposed by

Peffers et al. (2007). The first sequence of this design and refine, demonstrate, and evaluate phase was carried out in collaboration with the experts from Origo, who assessed the conceptual solution and the artifact's smart contracts, constructively criticizing those characteristics of the solution that would hinder the system's implementation in a corporate environment. Their contributions proved essential.

Thanks to their corporate experience we were able to identify elements of the initial solution that required improvement if the solution was to be implementable. We refined the solution in further loops. The focus of one of these refinement loops was the dynamic compensation system that we propose here. To our knowledge, all previous blockchain-based KYC solutions propose only static compensation between participating FIs, neglecting the fact that update costs might be incurred over time due to the demands of continued compliance with KYC regulations. Therefore, in the aforementioned loop we focused solely on establishing a smart contract that could accommodate dynamic compensation among participating FIs, and developed a solution that both allows for dynamic payments and respects proportionality in the payments made among FIs. We conducted another refinement loop focusing on the role of the TTP, a role that is central to all the solutions published thus far. We concentrated on reducing the reliance of the system on controls conducted by the TTP, ensuring instead intrinsic incentives for participating FIs. Our aim in suggesting this improvement is to increase the autonomy of the system by transforming the artificial incentives present in previously proposed systems into intrinsic incentives and to ease the implementation of the system by means of distributed a database architecture like the one proposed by Siegenthaler and Birman (2009a and 2009b), which is used in the healthcare sector to securely share patients' private information among hospitals. Combining blockchain technology and the distributed database architecture suggested by Siegenthaler and Birman (2009a and 2009b), in general and in this particular manner, has already been suggested by Parra-Moyano and Schmedders (2018).

6 Optimized, dynamic KYC system

In this section we describe the process and logic of the optimized, dynamic system we propose for reducing the cost and proportional cost share of KYC. We start with a description of the assumptions and conditions that must be fulfilled by the system and continue with a non-technical description of the system. We conclude with a description of the proposed system's technical aspects.

6.1 Assumptions and conditions

The solution that we propose in this section combines elements of the proposals that we describe in Sections 3 and 4, but specifically takes as its basis the system suggested by Parra-Moyano and Ross (2017) and combines it with the distributed database architecture suggested by Siegenthaler and Birman (2009a and 2009b). The four key assumptions of Parra-Moyano and Ross (2017) are:

1) All FIs with access to the system respect and work in the same regulatory framework.

2) All customers can be categorized by the effort required of an FI to conduct the KYC process for them.

3) All FIs with access to the system agree on an average cost of conducting the KYC process per category of customer.

4) A TTP, such as a regulator, maintains the system and approves the FIs that have access to the system.

Assumptions one and four are required in order to achieve the goal of proportional cost sharing among participating FIs working in the same regulatory framework. In the present paper we specify assumptions two and three in greater detail such that a system that fulfills these assumptions is able to compute not only the average cost—per category of customer—of conducting the KYC process once, but also the cost of updating, where necessary, the KYC-related information of any existing customer. Specifically, we suggest using measurable parameters that are derived from the time spent on conducting the KYC process (e.g., the size of a corporation, number of documents required, and number of beneficiaries) and that can be used to dynamically determine both these sets of costs. We state the revised assumptions two and three as follows:

1) All customers can be categorized by the effort spent on the KYC process*, and by the effort required to update their KYC status*.

2) All FIs with access to the system agree on the cost of conducting the KYC process per category of customer, *and on the cost of updating the KYC status of a customer*.

The greater detail contained in these two revised conditions implies a significant improvement on the system proposed by Parra-Moyano and Ross (2017) since it enables us to replicate, in a closer manner than proposed by Parra-Moyano and Ross (2017), the current nature and requirements of the FATF Recommendations (2017). More specifically, the fact that we allow for the dynamic but transparent measurement of KYC costs and—more importantly—for the dynamic correction and updating of the KYC status of any existing customer enables us to propose a less rigid system than that put forward by Parra-Moyano and Ross (2017).

Parra-Moyano and Ross (2017), further, define four conditions that the system must fulfill in order to ensure a correct incentive structure. These conditions are:

Proportionality: Ensure that the costs are shared proportionally among all the participating FIs.

Irrelevance: Ensure that participating FIs do not have an incentive either to be the first FI conducting the KYC process or to be one of those that use the results generated by the first FI.

Privacy: Ensure that one FI cannot infer, from the system, with which other FIs a customer is working.

No-minting: Ensure that no participating FI has an incentive to simulate having conducted the KYC process for a customer such that it can claim for compensation to which it is not entitled.

These assumptions and conditions constitute the yardstick against which we measure the applicability of the system that we propose in the following subsections.

6.2 Non-technical description of the optimized, dynamic KYC process

A consortium of FIs can use their existing database architecture to construct a system like the one proposed by Siegenthaler and Birman (2009a and 2009b). In order to grant read and write permission to FIs other than the one that conducted the KYC process, we use a private, PoA-based blockchain in which only FIs belonging to the consortium can participate. These two pieces of technology (the distributed database architecture to share sensitive data and the blockchain technology to manage permissions) constitute the main innovation of our system.

Whenever a customer has yet to be registered on the network, the first FI to onboard that customer using the network (hereafter referred to as the "Home Bank") must proceed in the following manner:

1) The Home Bank gathers all necessary documents, verifies the customer's identity, and generates a digitally signed document indicating the outcome of the core KYC process—this outcome can be either "approved" or "rejected".

2) The Home Bank stores all the documents used in the KYC process as a "document package" in its own encrypted database.

3) The Home Bank creates a smart contract for the customer and on it stores the hash of the document package, the network address of the customer, and the monetary value (m) that corresponds to the cost of conducting the KYC process for this customer (the last of these according to the predefined category to which this customer belongs in terms of the effort required to conduct the KYC process). When the smart contract is deployed by the Home Bank, its ownership passes from the Bank to the customer.

The smart contract records the address of the Home Bank in a list referred to as the list of onboarding institutions. At this point, the only FI on that list is the Home Bank. When the customer seeks to work with a second FI, this second FI—"Bank B" for brevity—must proceed in the following manner:

1) Bank B activates the smart contract of the customer and thus learns the proportion of the already incurred cost that it has to pay to the smart contract in order to get permission to read the documents related to the customer stored in the Home Bank's database. Since Bank B is the second FI that intends to work with the customer, it should pay the existing cost m divided by two.

2) Since the customer is willing to work with Bank B, they will grant permission to Bank B to pay the fraction $\frac{m}{2}$ to the smart contract such that once Bank B conducts the payment it automatically gets the right to read this customer's document package as stored in the Home Bank's database.

3) Bank B can hash the document package and verify that the documents it has received are the same as those that were analyzed and generated by the Home Bank.

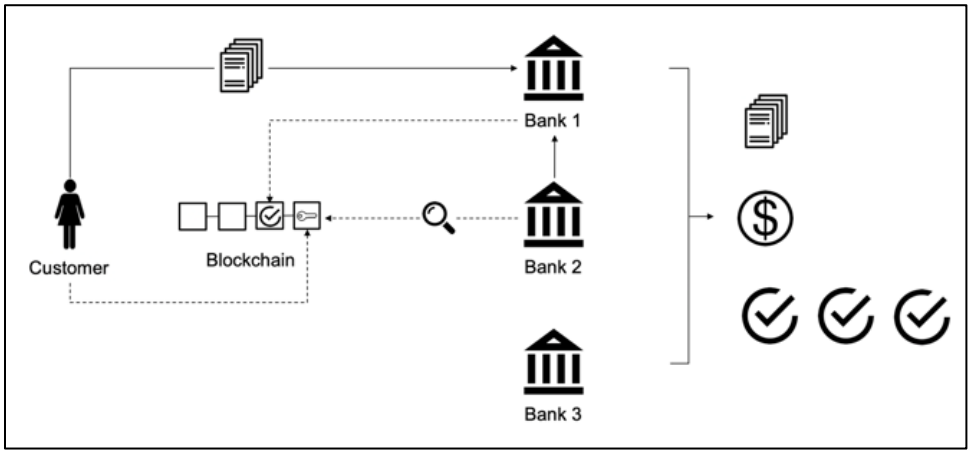

Figure 3. Schematic Representation of Our System

One of the difficulties in applying the system put forward by Parra-Moyano and Ross (2017) is caused by the fact that everything written on a distributed ledger is visible to all those who participate in the network. This visibility introduces a vulnerability to the system's incentive structure because the hashes stored on the ledger can be obtained by a user with enough technical knowledge, and that user is then able to see the KYC status of a customer ("approved" or "rejected") without having paid the corresponding contribution. In order to solve this issue, Parra-Moyano and Ross (2017) propose that all the FIs that work with a customer must pay the corresponding portion of the KYC process to a smart contract in order to appear in the list of onboarding institutions. This setup ensures that the TTP can always conduct controls in order to verify that all necessary contributions have been paid by all those FIs that are benefiting from a KYC process or update carried out by another FI. This is, however, an artificial and not an intrinsic incentive for participating FIs: they could choose not to pay the contribution if they know that the TTP might not control all transactions for all customers. In order to transform this artificial incentive into an intrinsic one, we make use of a distributed database architecture, which allows us to

eliminate the TTP and ensure that only FIs that have paid their proportion of the cost can actually read information regarding the customer.

Respecting the contribution structure derived by Parra-Moyano and Ross (2017), the system we propose here would allow all the FIs working with a customer to proportionally share the cost of the onboarding process because they would all pay $\frac{m}{k}$; m being the cost of the onboarding process and k the number of FIs working with the customer.

Once a second institution—Bank B, in this case—has followed the steps necessary to verify that a customer has already been onboarded, two possible situations might arise. Either no update or correction to the document package of this customer is required (as suggested by Parra-Moyano and Ross (2017)) or—and this is the more likely case—a KYC update process must be followed by Bank B in order to comply with the national regulatory regime. If no update is required, the payment structure remains as presented so far. If Bank B must pay the contribution $\frac{m}{2}$ to the Home Bank via the smart contract and at the same time realizes—after checking the document package—that an update is required, the Home Bank (and, later in the process when multiple FIs have worked with the customer, all those FIs that have previously worked with this customer) must somehow be informed of this and must also proportionally contribute to the cost of the update process—a process that Bank B will carry out and from which the Home Bank is (and later, all relevant FIs are) going to benefit. In order to allow for the sharing of updated information and to enable a dynamic but private communication and compensation channel connecting FIs, we define a further variable—namely, c—which represents the "Cost of Update". We assume that $c < m$ because when following the update process Bank B already possesses the previously available information and does not need to start from scratch. Nevertheless, c can be subject to specific factors and conditions. Were this the case, the situation would then look as follows:

1) Bank B would have to pay the contribution $\frac{m}{2}$ to the Home Bank via the smart contract and would also realize that an update process is required in order for it to comply with the regulatory regime.

2) Bank B would follow the update process and would generate a digitally signed document that indicates the process' outcome, which again can be either "approved" or "rejected".

3) Bank B would store all the documents used in the KYC process as an "updated document package" in its own database. This updated document package would now be readable by all those FIs that are listed in the smart contract and that have, directly or indirectly, paid for the update (at this stage, Bank B and the Home Bank).

4) Since Bank B has carried out the update process and borne its costs, the appropriate proportions of these costs must be paid by the institution(s) listed in the list of onboarding institutions (at this stage only the Home Bank) to Bank B.

In order to ensure that step four is respected, we suggest a time lag (or block sequence that needs to be added to the blockchain) between the moment at which an FI, k, other than the Home Bank pays its contribution $\frac{m}{k}$ via the smart contract and the moment at which the previous $k-1$ institutions can redeem the corresponding funds from their accounts. This time lag enables us to ensure that if an update to the KYC process needs to be carried out by the most recent FI that onboards the customer (the FI paying $\frac{m}{k}$ to each FI that previously worked with that customer), a proportional part of the update costs is deducted from the original $\frac{m}{k}$ contribution paid by the FI conducting that update. All the FIs working with the customer thus contribute to the cost of this KYC update and pay an amount equal to $\frac{m}{k} - \frac{c}{k}$. The system thus guarantees that all participating FIs that work with a given customer have access to the most recent, updated version of the document package.

This procedure for immediate and automatic proportional cost sharing also allows all FIs to have access to the most recent documents at all times. Assuming that k FIs have onboarded a given customer prior to an update being required, the FI that carries out the KYC update process assigns a value c to the update. This value represents the amount, in monetary terms, that the FI has spent on the update. All other $k-1$ FIs are then required to pay $\frac{c}{k}$, which they transfer to the FI that has carried out the update. In this setup, the total cost of onboarding and updating the customer, $m + c$, is proportionally distributed among the k FIs, which pay $\frac{m}{k} + \frac{c}{k}$ each. This process is repeated each time an update is required.

We acknowledge that the implementation of this update-costs element depends on agreement among participating FIs with regard to which documents are being updated. The method applied in the PoC presented in this paper assumes fixed update costs that are dependent on the initial KYC cost m. For each smart contract there are three types of update cost that can be assigned to the update. Each type is a fraction of the initial KYC cost. For purposes of illustration, we choose to use $c = \frac{m}{2}$, $c = \frac{m}{4}$, and $c = \frac{m}{6}$.

After the update has been carried out and the update cost has been distributed proportionally among all onboarding FIs, the total cost of the customer increases from m to $m + c$. Assuming that a total of k FIs have access to the system, all new onboarding FIs are required to pay $\frac{m+c}{j}$, where $j = k + i$ and $i = \{1,2, \dots, K-k\}$.

This system would respect the four (revised) assumptions and the four conditions that we define earlier in this section, and would constitute a significant improvement—in terms of efficiency, applicability, incentive structure, and maintenance costs—over all the previous solutions that propose improving the KYC process by means of DLT.

6. 3 Technical description of the optimized, dynamic KYC process

This section shows how we implemented the desired properties of the smart contract as functions programmed in *Solidity*, a programming language for smart contracts deployed on the Ethereum network. Here we describe the essential code for the implementation of the proposed system. The proposed system relies on two smart contracts: the KYC smart contract and the token smart contract. For the token smart contract an open-source code based on ERC-20 recommendations is used (Buterin & Voglsteller, 2015). We made slight

adjustments to the ERC-20 code to simplify communications between the two smart contracts.

The smart contract tracks the FIs addresses, the amount of tokens each FI transfers to the contract, and the hash imported initially or updated as a struct Bank.

```
struct Bank {
  address bankAddress;
  bytes32 bankHash;
  uint256 balances;
}
address[] public banks_ids;
```

Figure 4. Smart Contract Structure

The system records the FIs that have onboarded the customer in a list of addresses as banks_ids. banks_ids is what we have referred to as the list of onboarding institutions in previous sections. When a customer is added to the network the FI that carries out the initial KYC process creates the smart contract for that customer. The PoC uses a separate smart contract to deploy the smart contract used for KYC purposes. The deployment smart contract uses a function deployContract that takes as input the parameters necessary to deploy the KYC smart contract.

```
contract deployCheckHash {
  function deployContract(address newAddress, uint256 typeOf,
      address _bankAddress, string _hash, address tokenAddress)
      returns (address){
    return new checkHash(newAddress, typeOf, _bankAddress, _hash,
        tokenAddress);
  }
}
```

Figure 5. Deployment Smart Contract

The necessary inputs required by deployContract are the address of the customer (newAddress), the cost of the customer ($typeOf$), the address of the FI (_bankAddress), the hash of the KYC documents ($\_hash$), and the address of the token smart contract (tokenAddress). The token contract, as well as all the code, can be consulted in the GitHub Repository "KYC-Optimized-and-Dynamic-System-using-Blockchain-Technology (Tth2549, 2017).

The deployment smart contract uses its inputs to deploy the KYC smart contract as checkHash. The checkHash smart contract uses an initializing function to store the required information and transfer the ownership of the smart contract from the FI to the customer.

The initializing function is only called when the smart contract is created. After ownership has been transferred the function calls storeProof to store the hash in the smart contract initializing function. Next, the FI is added to banks_ids and the hash is linked to the FI to

record what hash the first FI imported. Further, the function uses the input typeOf to set the cost of the customer. Note that in this PoC we assume that there only exist three types of customers, which cost 100,000, 200,000, and 300,000 tokens, respectively. These values were chosen arbitrarily for illustration purposes.

The customer is registered as the owner of the contract and therefore reserves the right to erase/kill the smart contract. To do this the customer needs to approach an FI that has access to the network. Ownership of the contract is then transferred from the customer to that FI so the function kill can be called and the contract erased.

```
function checkHash(address newOwner, uint256 typeOf, address
    _bankAddress, string _hash, address tokenAddress) {
  transferOwnership(newOwner);
  banks[_bankAddress].bankAddress = _bankAddress;

  TokenAddress = tokenAddress;

  bytes32 proof = proofFor(_hash);
  storeProof(proof);
  banks[_bankAddress].bankHash = proof;
  banks_ids.push(_bankAddress);

  if (typeOf == 1) {
    contract_cost = 100000;
  }
  else if (typeOf == 2) {
    contract_cost = 200000;
  }
  else if (typeOf == 3) {
    contract_cost = 300000;
  }
  else throw;
}
```

Figure 6. Smart Contract initializing Function

After the KYC smart contract has been successfully deployed, it can be used by other FIs. The smart contract has a function, payment, that can be called by any FI. The payment function is used to inform FIs of the sum they must pay in order to interact with the smart contract.

```
function transferOwnership(address newOwner) onlyOwner {
  owner = newOwner;
}

function kill() {
  if (msg.sender == owner) {
  selfdestruct(owner);
  }
}
```

Figure 7. Smart Contract Function transferOwnership

The payment function simply takes the cost of the customer and divides it by the number of FIs that have previously onboarded that customer plus one. If the FI accepts the amount to be paid, it can proceed to pay the given amount and use the smart contract further. The smart contract uses two functions to transfer the tokens—tokenFallback and payContract. tokenFallback stores the tokens in the smart contract and then uses payContract to distribute these tokens accordingly to other FIs.

```
function payment() constant returns (uint256){
  return contract_cost/(banks_ids.length+1);
}
```

Figure 8. Smart Contract Function payment

Note that only after the tokenFallback function has verified that the amount transferred to the smart contract is correct does it call payContract.

```
function tokenFallback(address from, uint256 amount, bytes data){
  if (amount == contract_cost/(banks_ids.length+1)){
    banks[from].balances += amount;
    banks[from].bankAddress = from;
    payContract();
    banks_ids.push(from);
  }
  else throw;
}
```

Figure 9. Smart Contract Function tokenFallback

```
function payContract() {
  TokenERC20 t = TokenERC20(TokenAddress);
  uint256 totalCost = contract_cost/(banks_ids.length+1);
  for (uint i = 0; i < banks_ids.length; i++){
    t.transfer(banks_ids[i],totalCost/banks_ids.length);
  }
}
```

Figure 10. Smart Contract Function payContract

The function payContract uses a simple loop to transfer the payment to all previous FIs. Note that the function tokenFallback adds the new FI to the list of onboarding institutions, banks_ids, after the payContract function has been called.

After the FI has paid the contract and the payment has been distributed, the FI can use the function checkDocument to compare the hash stored in the smart contract to the hash it has generated from the documents supplied by the customer.

```
function checkDocument(string document) alreadyPaid constant
    returns (string) {
  bytes32 proof = proofFor(document);
  return hasProof(proof);
}
```

Figure 11. Smart Contract Function checkDocument

The function checkDocument uses the modifier alreadyPaid to ensure that the FI calling it is on the list of onboarding institutions—that is to say, that the FI has already paid its share.

```
modifier alreadyPaid {
  if (banks[msg.sender].bankAddress != msg.sender) throw;
}
```

Figure 12. Smart Contract Modifier alreadyPaid

If the FI has paid its share, checkDocument converts the input hash to the same format as the hash stored in the smart contract, using proofFor.

```
function proofFor(string document) constant returns (bytes32) {
  return sha256(document);
}
```

Figure 13. Smart Contract Function proofFor

Further, the function hasProof is used by the smart contract to compare the hash that results from hashing this document with the hash stored in the smart contract.

```
function hasProof(bytes32 proof) constant returns (string) {
  if (proofs.length == 0) return "No data here.";
  if (proofs[proofs.length-1] == proof){
    return "Data is correct!";
  }
  else {
    for (uint256 i = 0; i < proofs.length; i++) {
      if (proofs[i] == proof) {
        return "Data is old, has been approved before." ;
      }
    }
  }
  return "Data has never been approved!";
}
```

Figure 14. Smart Contract Function hasProof

The hasProof function has three possible returns: “Data is correct!” if the hash is a match, “Data is old, has been approved before” if the hash that the FI is comparing has been updated by another FI and the new FI is using old documents, and “Data has never been approved!” if the hash does not match and has never been used before.

If the documents used for the KYC process need to be updated, the FI can update the hash in the smart contract using the function notarize.

```
function notarize(string document) alreadyPaid {
  bytes32 proof = proofFor(document);
  storeProof(proof);
}
```

Figure 15. Smart Contract Function notarize

Notarize converts the hash of the updated document to SHA256 and then adds the new, resulting hash to storage in the smart contract using storeProof. The function notarize notes which FI updated the hash so there is a record of which FI has imported each specific hash. Additionally, the function keeps track of all the hashes used for the customer, in storeProof.

```
function storeProof(bytes32 proof) internal {
  proofs.push(proof);
  if (banks_ids.length > 1){
    banks[msg.sender].bankHash = proof;
  }
}
```

Figure 16. Smart Contract Function storeProof

7 Conclusions

In this paper we propose a refined, dynamic, blockchain-technology-based KYC system that reduces the costs of the KYC process, allows these costs to be shared proportionally by participating FIs, eliminates the need for a TTP to manage permissions in the system, and conducts dynamic updates with regard to the status of FIs' customers over time. Further, we develop a PoC that can be used by FIs and regulators to implement the proposed system and to explore variations of the system. This system is based on previous system proposals and emerged through the application of DSR. Specifically, the system development process employed a series of loops of design, evaluation, and demonstration that served to improve the system's applicability to a real-life corporate environment and to resolve the inefficiencies of previously proposed systems.

The major contribution of the system we propose has been that it eliminates the need for a TTP, making the system truly decentralized, and that it makes possible a distributed data storage architecture that is independent of the blockchain architecture, which makes implementation more cost efficient and easier for FIs. In our system the blockchain is only used to grant and manage the distributed database's reading permissions. This strengthens the incentive structure for participating FIs, ensuring that they act in the way in which they are meant to act because of an intrinsic impulse and not simply due to the fear of being supervised by the regulator. Our system has made another vital contribution in that it allows each participating FI to dynamically update each customer's status, such that if an FI identifies—for example—a flaw with regard to the legality of a customer's activities, it can revise that customer's status and propagate this information through the system to those other FIs that work with that customer. The implications of this are, in fact, crucial because this feature not only allows FIs to revise the status of any given customer, it also increases the quality of the information—in the form of KYC documentation—available to the network, which ensures that all participating FIs remain up-to-date in terms of the validity of the KYC status of any customer.

Additionally, we prove the concept by means of an artifact—coded in the language *Solidity*—that can be easily used by any interested individual to test and develop the concept, implement it in an experimental environment, and further develop it and adapt it in order improve its applicability and usefulness. We are convinced that the conceptual system and the PoC that we propose here can serve to improve the existing KYC process and that they constitute one necessary further step toward the adoption of blockchain-based systems in the corporate environment.

8 Bibliography

Buterin, V. (2013). *Ethereum White Paper*. Retrieved from https://github.com/ethereum/wiki/wiki/White-Paper

Chapelle A, Gereth W, Panayi E (2015). *Trends in crypto-currencies and blockchain technology*. Retrieved from https://arxiv.org/abs/1508.04364

Civic. (2017). *Secure Identity Authentication*. Retrieved from https://www.civic.com/developers

Digiconomist. (2017). *Bitcoin Energy Consumption Index*. Retrieved from https://digiconomist.net/bitcoin-energy-consumption

European Securities and Market Authority (2016). *The Distributed Ledger Technology Applied to Securities Markets.* Retrieved from https://www.esma.europa.eu/sites/default/files/library/2016-773_dp_dlt.pdf

The Financial Action Task Force. (2012 – 2017). *International Standards on Combating Money Laundering and the Financing of Terrorism and Proliferations*. Retrieved from http://www.fatf-gafi.org/media/fatf/documents/recommendations/pdfs/FATF%20Recommendations%202012.pdf

Glaser, F (2017). *Pervasive decentralisation of digital infrastructures: a Framework for blockchain enabled system and use case analysis*. In: Proceedings of the 50th Hawaii international conference on system sciences. https://doi.org/10.24251/HICSS.2017.186

Hyperledger. (2017). *Hyperledger Fabric*. Retrieved from https://hyperledger-fabric.readthedocs.io/en/latest

IBM. (2017). *IBM Cloud*. Retrieved from https://console.bluemix.net/docs/services/ObjectStorage/os_security.html

IEEE Computer Society (2000). IEEE Standard Specifications for Public-Key Cryptography.

Mizrahi, A. (2016). *A blockchain based property registry*. Retrieved from https:// chromaway.com/research

Nakamoto, S. (2008). *Bitcoin: A peer-to-peer electronic cash system*.

Retrieved from https://bitcoin.org/bitcoin.pdf

Parra-Moyano, José and Schmedders, Karl, The Liberalization of Data: A Welfare-Enhancing Information System (2018). Available at SSRN: https://ssrn.com/abstract=3302752

Parra-Moyano, José and Ross, O (2017) KYC Optimization Using Distributed Ledger Technology. *Bus Inf Eng*.

Peffers K, Tuunanen T, Rothenberger MA, Chatterjee S (2007). A design science research methodology for information systems research. J Manag Inf Syst

Ruce, P. (2011). The Bright Side of Knowing Your Customer. *The Banking Law Jurnal*, 128.

Shocard (2017). *How it works*. Retrieved from https://shocard.com/how-it-works

Siegenthaler M and Birman K (2009a), *Privacy enforcement for distributed healthcare queries*, in Pervasive Health.

Siegenthaler M and Birman (2009b), *Sharing private information across distributed databases*, Eighth IEEE International Symposium on Network Computing and Applications.

St. Laurent, A (2004): *Understanding Open Source and Free Software Licensing*, Sebastopol, California: O'Reilly Media, Inc.

Tth2549. (2017). *KYC-Optimized-and-Dynamic-System-using-Blockchain-Technology*. Retrieved from https://github.com/tth2549/KYC-Optimized-and-Dynamic-System-using-Blockchain-Technology

Thomson Reuters. (2017). *Know Your Customer (KYC) independent survey*. New York, NY.

UBS (2016). *Utility Settlement Coin Concept on Blockchain Gathers Pace*. Retrieved from https://bit.ly/2BWuM5U

U.S. Department of Commerce (2015). *Federal Information Processing Standards Publication: Secure Hash Standard*. Retrieved from http://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.180-4.pdf

Voglsteller, F. and Buterin, V. (2015). *ERC-20 Token Standard*. Retrieved from https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20-token-standard.md

Wood, G. (2016). *Ethereum: a secure decentralised generalised transaction Ledger*. Retrieved from http://gavwood.com/paper.pdf

World Economic Forum (2016). *The Future of Financial Infrastructure*. Retrieved from http://www3.weforum.org/docs/WEF_The_future_of_financial_infrastructure.pdf