Blockchain-based federated identity and auditing

Mahmoud M. ElGayyar and Hany F. ElYamany

Department of Electrical and Computer Engineering,
Western University,
London, ON, Canada
Email: melgayya@uwo.ca
Email: helyama@uwo.ca
and
Department of Computer Science,
Suez Canal University,
Ismailia, Egypt

Katarina Grolinger and Miriam A.M. Capretz*

Department of Electrical and Computer Engineering,
Western University,
London, ON, Canada
Email: kgroling@uwo.ca
Email: mcapretz@uwo.ca
*Corresponding author

Syed Mir

London Hydro,
London, ON, Canada
Email: mirs@londonhydro.com

Abstract: A federated identity is a single identity that enables users to access multiple services across a network of business parties. Such identities are subject to various threats and attacks and face diverse challenges including identity leaks, centralised management, auditing limitations, and long breach investigation processes. This paper proposes a framework aimed at automating and decentralising the generation and auditing of a robust and secured blockchain-based federated identity in a marketplace. Business parties participating in the marketplace form the nodes of a distributed blockchain network and participate in the creation of federated identities. Users of this network can access services provided by any one of the participating parties using a single federated identity. All transactions are fully audited in the blockchain, meaning that participating parties can monitor access to their service and users can trace the use of their identities. The proposed framework has been evaluated using two blockchain technologies (Ethereum and Hyperledger Fabric) to measure its performance in public and permissioned blockchain environments.

Keywords: blockchain; smart contracts; federated identity management; auditing.

Reference to this paper should be made as follows: ElGayyar, M.M., ElYamany, H.F., Grolinger, K., Capretz, M.A.M. and Mir, S. (2020) 'Blockchain-based federated identity and auditing', *Int. J. Blockchains and Cryptocurrencies*, Vol. 1, No. 2, pp.179–205.

Biographical notes: Mahmoud M. ElGayyar is a Postdoctoral Associate (PDA) at Western University, Canada. Previously, he was an Assistant Professor in the Faculty of Computers and Informatics at the Suez Canal University, Egypt. He received his PhD in Distributed Systems from the Bonn University, Germany, his MSc in Net-Centric (Computer Science) from the Trento University, Italy, and BSc in Computer Science from the Suez Canal University, Egypt. His research activities are related to blockchain, cloud computing, the internet of things, big data analytics, web services, visualization, and software engineering. He has approximately 12 years of practical experience in the IT industry in various roles: software developer, system analyst, and project manager.

Hany F. ElYamany is a Postdoctoral Associate at Western University, Canada. He is also an Associate Professor in the Computer Science Department at the Suez Canal University, Ismailia, Egypt. He obtained his PhD in Software Engineering from Western University, Canada, MSc in Computer Science from Ain Shams University, Egypt and BSc in Computational Sciences from the Suez Canal University, Egypt. He has been working in the software engineering area in both academia and industry for 20 years. His research interests include software engineering, service-oriented architecture, blockchain, cloud computing, and security. He is an IEEE senior member.

Katarina Grolinger is an Assistant Professor in the Department of Electrical and Computer Engineering at Western University, Canada. She received her MEng and PhD in Software Engineering from Western University. Previously, she obtained her BSc and MSc in Mechanical Engineering from the University of Zagreb, Croatia. She is also a Certified Oracle Database Administrator with over ten years of industry experience in database administration and software development. Her research interests include sensor data analytics, big data, internet of things, machine learning, and data management.

Miriam A.M. Capretz is a Professor in the Department of Electrical and Computer Engineering at Western University, Canada. Before joining Western University, she was with the University of Aizu, Japan. She received her BSc and MESc from the UNICAMP, Brazil, and her PhD from the University of Durham, UK. She has been working in the software engineering area for more than 35 years and has been involved with the organization of workshops and symposia and has been serving on program committees at international conferences. Her current research interests include cloud computing, big data, blockchain, service-oriented architecture, privacy, and security.

Syed Mir is the Vice President of Corporate Services and Chief Information Officer at the London Hydro, where is responsible for 'metre to cash' services and enabling a digital utility. He is also the Chair of Green Button Alliance

promoting an open standard for data access across North America. He has over 30 years of utility experience and has served in various roles in several companies in the energy sector. He received his BSc in Computer Science from the Western University. His research interests are cloud computing, mobility, smart grids and social responsibility.

1 Introduction

Identity is a piece of information that uniquely identifies a person requesting access to a service or location. This information is usually composed of distinguishing entities that may include personal information (e.g., name and social insurance number), biometrics (e.g., fingerprints), or personal documents (e.g., a passport or driving license) (Bertino and Takahashi, 2011). Identities are used in our daily activities, including entering buildings, accessing computers and bank accounts, and for a variety of other services. They can take different forms ranging from government documents to smart cards and digital identities. Governments and business parties are continuously working to protect these identities and prevent severe damage including loss, fraud, and theft. New technologies are needed to provide robust and secure identities and to reduce the cost of possible breaches.

With the massive growth of the internet and mobile devices, the importance of *digital identities* has grown because they enable users to interact with online services efficiently, remotely, and inexpensively. A digital identity is structurally similar to a regular identity; however, it is issued and verified electronically by an identity provider (IdP). Use of identities inside the digital space is often associated with high risks including fraud and theft, resulting in monetary and reputation losses. For example, a study done by Javelin Strategy and Research showed that the identity fraud rate is gradually increasing every year (Javelin Strategy and Research, 2017), and government statistics in the USA and Canada have emphasised the same trends (U.S. Department of Justice, 2017; Canadian National Bank Insurance, 2015).

Technically, traditional digital identity management systems are faced with several challenges. First, when identity fraud or theft occurs, the user is rarely notified at the time of the attack, and it may take up to several years to investigate and determine the negative impact, as in the case of Yahoo (Wall Street Journal, 2017). Second, many companies are still faced with identity leaks in spite of the strong encryption techniques that they have used to secure these identities. For example, Equifax, a credit monitoring company in North America, was a recent victim of identity theft (Equifax, 2018). Identity creation and verification are usually accomplished through a centralised service provider (SP) that represents a critical point of failure or attack (Werner et al., 2017).

Federated identity refers to a single identity created for accessing services or platforms provided by different business parties. Users with a federated identity can navigate among several applications distributed across different parties (Malik et al., 2015). Those parties can be scattered or connected to each other in the form of an integrated marketplace. For example, a healthcare marketplace including hospitals, clinics, and insurance companies may use a federated identity system to enable users to access their services through a single identity. This identity acts as a single sign-on

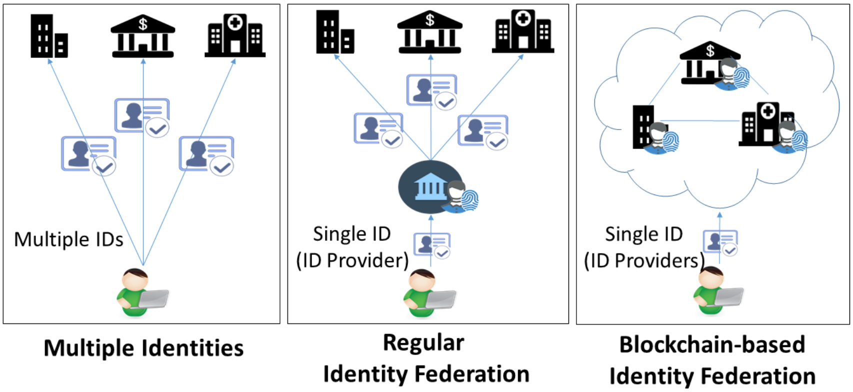
(SSO) that authenticates users with the marketplace. However, current federated systems in such marketplaces often lack the ability to track and record all ongoing transactions or to track access to services or data. Auditing facilities can provide an irrefutable chain of evidence enabling traceability and accountability of each party's actions, thereby improving the security of businesses operating within the marketplace.

Similar to digital identity, federated identity suffers from various issues including breach investigation, identity leaks, and centralisation. Consequently, there is a need to investigate new technologies for building a more robust federated identity system that can deal with these challenges. Blockchain technology is an excellent candidate for this task due to its decentralisation and security characteristics.

Blockchain is a decentralised digital system that records transactions or anything else in an incorruptible digital form (Nakamoto, 2009). Smart contracts are logical components running inside the blockchain network that control business logic and transaction execution (Wood, 2017). Blockchain technology has recently become popular in many business domains, including modern identity management systems. Encrypted transactions and immutability features prevent hackers from altering the stored identities (uPort Inc., 2019; Sovrin Foundation, 2019); however, use of blockchain for federated identity management has been quite limited (ShoCard, 2019; Faidella and Schukai, 2017).

This paper introduces the blockchain-based federated identity framework (BFIF) for managing federated identity and auditing processes. Business parties within an integrated marketplace can customise the proposed framework to manage the federated identities of their users according to their needs and/or regulations as well as to cope with dynamic changes in the marketplace.

Figure 1 Federated identity vs. blockchain-based federated identity (see online version for colours)



In the proposed solution, the parties in the blockchain network act as IdPs or authenticators, removing the role of a third party and thus reducing unwanted external exposure of credentials. Any individual party can verify user credentials and generate the federated identity. Moreover, a full auditing process enables both users and business parties to track the usage of their identities and services. Figure 1 illustrates the differences between a multiple-identities approach, a regular identity federation, and a blockchain-based identity federation, with the last one providing a single identity without the need for a third-party IdP. The advantage of the blockchain-based federated system is in enabling users to access multiple services with a single identity while maintaining all transactions inside the blockchain network for auditing purposes.

The main contributions of this work include the following:

- The proposed framework generates unique and secure federated identities for accessing multiple services provided by different business parties.
- Smart contracts enable the framework to adapt to changes in the regulations and rules governing identity management processes.
- The auditing features enable users to track how their identities are used and provide business parties with the ability to track use of their services.
- The proposed framework has been implemented with public and permissioned blockchain environments. The presented experiments demonstrate the performance of the two implementations.

The rest of the paper is organised as follows. Section 2 provides background concepts, and Section 3 describes related work. Section 4 proposes the blockchain-based federated identity and auditing framework. Section 5 discusses the implementation of the proposed framework. Section 6 demonstrates and discusses the performance of the proposed framework on public and permissioned blockchain environments, and Section 7 concludes the paper.

2 Background

This work is built on four concepts and technologies: blockchain, smart contracts, federated identity, and auditing, which are further detailed in the next subsections.

2.1 Blockchain and smart contracts

A blockchain can be considered as a type of data structure that preserves the relationship between stored data within a distributed environment. The stored data are organised in the form of linked blocks maintained in a shared ledger (Nakamoto, 2009). A shared ledger is a replicated and synchronised digital data structure that is geographically spread across multiple nodes that form a peer-to-peer (blockchain) network. A node refers to any physical computing resource (e.g., a server) that runs the blockchain software and is connected to the blockchain network. Every node receives a copy of the shared ledger upon joining the blockchain network and uses the blockchain software to verify and relay blocks. A block inside the blockchain contains a set of transactions and a header that holds the block's metadata. These metadata incorporate a reference to the previous block and a fingerprint (hash) of the data inside this block. This fingerprint is used to verify the block's data. If hackers want to change the data at a certain point, they must regenerate all the fingerprints from that point forward.

Blockchain technology is structured based on a hybrid combination of other technologies, including P2P networking, cryptography, smart contracts, and others. These technologies are used to build either a public or a private (permissioned) blockchain. In a public blockchain, anyone can read or write to the shared ledger.

Conversely, a permissioned blockchain network is constructed among trusted parties, such as a group of business parties that structure a marketplace. In a permissioned blockchain network, only authorised nodes can access the shared ledger. Blockchain technology uses cryptography and digital signatures to prove identity, provide authentication, and enforce read/write access rights.

Smart contracts are a significant aspect of the blockchain technology (Wood, 2017). They are simply a piece of code deployed and stored in the shared ledger of the blockchain network. Each smart contract defines a piece of the business logic to which all parties inside the blockchain have agreed. The business logic carries out actions that are executed under certain conditions or rules. Currently, in the real world, smart contracts have been used to exchange money and other properties without involving a central authority or a third-party mediator.

2.2 Federated identity

Identity is a piece of information that can uniquely identify an entity (e.g., a user, business party or device) within a context. A digital identity is the corresponding electronic concept to the real identity of an entity. It can be used with different roles inside the digital space, such as identifying and authenticating users to perform online actions. Authentication is the process by which an entity (e.g., a user) proves its identity to another entity (e.g., a healthcare service).

Federated identity is generated based on a set of agreements among multiple parties to enable users to access various services published across various platforms managed by these parties (Grassi et al., 2017). Often, an IdP acting as a third party is responsible for generating and verifying users' identities based on their provided credentials. The parties (SPs) govern business services that consume users' identity information to check whether they should be granted access.

In a regular federated identity case, both parties and users build mutual trust with the IdP to manage the authentication process among them. In such a scenario, each party manages only transactions received from its users. Users cannot track all transactions occurring at any single party with which they are accustomed to interacting. Moreover, the IdP can become a single point of failure because when it fails, communications between users and parties also fail.

SSO is a popular authentication method (Bhosale, 2008). It offers users a single login (e.g., a username and password or a smart card) to access services within a single business party. However, federated identity enables users to share their identities across multiple business parties located in a federation domain. This means that federated identity can provide SSO, but not vice versa.

2.3 Auditing

The auditing process aims to gather information about various events within a system (Tejpar, 2010). This information is recorded in the form of a digital audit trail composed of an entity ID (who), purpose (what), timestamp (when), and logic (how) related to an event. The available audit trails are often used to run a non-repudiation process that constructs a chain of evidence. This chain of evidence ensures that no party can deny single participation in a sequence of events (e.g., a transaction).

Auditing can perform various tasks to satisfy accountability requirements, maintain proper use of personal information, and convey notifications to both users and parties with each business transaction whenever they may occur.

3 Related work

This section introduces current work on identity management using blockchain in both industry and academia. Blockchain has many characteristics such as decentralisation, security, and transparency. Several business parties have recognised the value of using blockchain technology to manage users' identities.

Dunphy and Petitcolas (2018) surveyed industrial tools that use blockchain for identity management, including uPort, Sovrin, and ShoCard. The authors analysed the strengths and gaps in each tool according to the laws of identity.

uPort Inc. (2019) is an Ethereum-based identity management platform. It enables public users to register and obtain a blockchain identity without performing any authentication. It introduces a framework for collecting users' identity attributes from other third-party trusted providers. In our work, the third party is eliminated; the parties involved in the proposed framework carry out identity proofing to keep the data and logic (coding) securely inside the blockchain. In this way, the risk of tampering with the transferred user information can be reduced.

Sovrin Foundation (2019) is another blockchain-based identity management framework. One particular characteristic of Sovrin is the use of a private (permissioned) blockchain network where only trusted parties are involved. The users must be represented by agents to interact with the Sovrin network. In our work, users can interact directly with the blockchain network. Moreover, our proposed framework enables users and business parties to track the usage of their identities and services.

ShoCard (2019) provides a blockchain identity built on Bitcoin technology. It produces a blockchain identity based on users' attributes and credentials, storing and encrypting all this information in a distributed ledger. ShoCard enables users to use their blockchain identity to authenticate themselves on other third-party applications. In our work, users' federated identities are used only inside the blockchain network representing the marketplace to prevent any possible information leakage. This scenario is preferable in a marketplace such as the healthcare system where sharing of users' identities and profiles is limited. Moreover, the auditing process is not present in the ShoCard system.

In academia, few studies have discussed the role of blockchain in identity management. Faidella and Schukai (2017) produced a system similar to the ShoCard system that introduces a generic framework for generating and storing users' blockchain identities based on their profiles and credentials. Unlike our work, neither the auditing process nor the degree of flexibility in changing federated identity management rules or policies are discussed.

Wolfond (2017) presented a generic discussion of using a blockchain solution to produce a secure identity for a marketplace. In that context, he discussed criteria of authentication, identity, and verification. He concluded in his study that a blockchain-identity solution has the potential to enhance access to online services in both the public and private sectors in Canada. In contrast, our work proposes a complete blockchain-based solution for federated identity and auditing that can be applied in any marketplace while addressing the requirements of both users and business parties.

Lee (2018) proposed blockchain-based ID as a service (BIDaaS). The author defined three entities: user, partner, and BIDaaS provider. The BIDaaS provider is usually a telecommunication company that can verify user credentials. The partner offers the services that a user is eager to access. The BIDaaS provider creates a virtual identity that enables the user to access the services provided by the partner through a mobile application. In our system, an external IdP is not needed. Any business party involved in the blockchain marketplace can verify the users' credentials and profiles. In addition, our solution addresses the auditing process.

Ebrahimi (2016) introduced another framework for blockchain-based identity management to enable different business parties to exchange generated identities. This framework depends on collecting users' profiles from third-party IdPs. The term 'link' was used to refer to data transfer between blockchain parties. In our work, the proposed framework can be implemented by modern blockchain technologies such as Hyperledger that can establish a private session to exchange sensitive information between the involved business parties when needed. In contrast to our work, the framework proposed by Ebrahimi (2016) does not consider the auditing process.

Kikitamara (2017) introduced a blockchain-based framework for hybrid digital identity. The suggested hybrid identity is a combination of the federated and user-centric identity concepts. The framework discussion covered five components that are needed to construct a blockchain-based digital identity for an open energy model. The defined components are entities, attributes, lifecycle, policies, and technology. For instance, the author identified that QR code and transport layer security (TLS) protocols are vital to building a robust blockchain-based identity. Our paper introduces a customisable blockchain-based framework for building a federated identity that can be implemented by different blockchain technologies.

Zyskind et al. (2015) proposed a modified computational version of blockchain technology to ensure privacy and share identities across different platforms. It provides different computational models that enable users to manage their identities safely according to privacy rules. This approach also removes the need for a third party in the suggested models. Zyskind et al. described several business scenarios that may accommodate their new model. However, the focus of our work is to introduce a generic blockchain-based framework for federated identity management and auditing in a marketplace.

Xia et al. (2017) and Azaria et al. (2016) proposed a permissioned blockchain network for sharing medical records among users and business parties. Both studies demonstrated the roles of strong blockchain properties like using engaged cryptographic keys as identities in sharing patients' sensitive data. Moreover, both papers declared the importance of the immutable shared ledger that enables users to access and follow their historical data. Our work exploits the same features of blockchain, such as immutability and cryptography, but for identity management and auditing, not only for data sharing.

Dinh et al. (2017) provided a survey of blockchain technology that showed the performance differences between private (permissioned) and public blockchain networks in addition to the regular database system. That paper also described the identity-sharing capabilities of both networks. Our work proposes a generic blockchain-based framework for federated identity management and auditing that could be implemented with different types of blockchain networks (e.g., Ethereum and Hyperledger).

Andoni et al. (2019) also reviewed blockchain technologies, but focused on the energy sector. Their study discusses 10 consensus protocols, contains 140 blockchain research projects, and presents various energy-sector use cases. They conclude that most developments are in early stages and that future advances are needed especially with respect to scalability, security, and decentralisation.

Existing consensus protocols have different advantages and drawbacks; consequently, efforts on improving them continue. For example, Tang et al. (2019) proposed reputation-based mechanisms for PoW protocol in order to incentivise honest mining and Wang et al. (2020) presented reputation incentive scheme for industrial IoT.

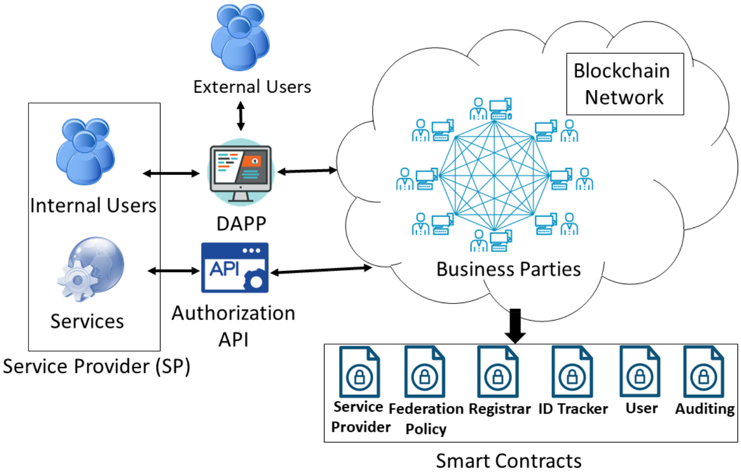
Efforts have been made to compare different blockchains. Dinh et al. (2018) analysed systems in respect to four characteristics: distributed ledger, cryptography, consensus protocol, and smart contract and presented BLOCKBENCH, a framework for benchmarking private blockchains with different data processing workloads.

Whereas works of Andoni et al. (2019), Tang et al. (2019), Wang et al. (2020), and Dinh et al. (2018) review the state of the art, improve consensus protocols, and evaluate existing systems, our work presents a framework for identity management and auditing. BFIF framework presented here could be implemented with different technologies discussed in those reviewed works.

4 Federated identity and auditing framework

This section proposes the BFIF for generating and managing a unified digital identity as well as for enabling users and business parties to track the use of their identities and services. In this framework, there are two main stakeholders: business parties and users. *Business parties* can play two roles: *SPs* deliver business services for users, such as utility services, whereas *IdPs* act as authenticators for users when they access business services.

Figure 2 Blockchain-based federated identity framework (see online version for colours)



As depicted in Figure 2, a group of trusted business parties that have a common user base or provide similar or complementary services comes together to form a blockchain-based circle of trust. The goal is to simplify service access for their users and to achieve immutable auditing. Examples of parties that may form such a federation include utilities, communication providers, and government organisations.

These parties become involved in a blockchain network to generate federated identities for users and to carry out auditing. Any of these parties can authenticate users to register in the blockchain network. Once the authentication process has been verified, a federated identity is generated to enable a user to gain access to the parties' services. Users can then track all operations or actions that their federated identities were involved in, and business parties can monitor the use of their services.

In a marketplace, two types of users are identified: *internal users* (administrators and auditors) and *external users* (customers). Both types of users communicate with the blockchain network through a decentralised application (DAPP) (Wood, 2017). A DAPP is the front-end that is used to make calls to the back-end running on a decentralised peer-to-peer blockchain network.

BFIF back-end is based on six smart contracts deployed on the blockchain: *SP*, *federation policy*, *registrar*, *ID tracker*, *user* and *auditing*. These smart contracts govern and manage BFIF federated identity and auditing modules as shown in Table 1:

1 *Federation administration module* deals with the circle of involved business parties and identity management policies.

2 *Federated identity management module* issues, updates, reactivates, and authorises user identities.

3 *Federated identity tracking module* provides auditing features.

These modules and the roles of each contract in their operation are discussed in the following subsections.

4.1 Federation administration module

The *federation administration* module uses two smart contracts: *SP* and *federation policy*, which are depicted in Listings 1 and 2 respectively. The *SP* smart contract enables the network administrator to build the circle of trust by adding (and removing) involved business parties as SPs using addServiceProvider and removeServiceProvider in Listing 1. This contract also enables SPs to add and remove services that they provide to the network, using addService and removeService in Listing 1.

Listing 1 Service provider smart contract

```
contract ServiceProviderSC{
   struct ServiceProvider {
      string publicKey;
      string name;
      string city;
      bool isActive;
   }
   struct Service{
       address serviceProvider;
       string redirect_url;
```

```
        string serviceId;
    }
    mapping(address=>ServiceProvider) serviceProviders;
    //mapping serviceId to Service
    mapping(string=>Service) services;
    address owner;
    modifier onlyOwner {
        require(msg.sender == owner);
        _;
    }
    modifier onlyServiceProvider() {
        require(serviceProviders[msg.sender].isActive);
        _;
    }
    constructor() public {
        owner = msg.sender;
    }
    function addServiceProvider(address providerAddress,
        ...) public onlyOwner{ }
    function removeServiceProvider(address providerAddress,
        ...) public onlyOwner{ }
    function isServiceProvider (address providerAddress)
        public returns(bool){  }
    function addService(string serviceName,...)
        public onlyServiceProvider{}
    function removeService(string serviceName)
        public onlyServiceProvider{}
}
```

Listing 2 Federation policy smart contract

```
contract FederationPolicySC{
    enum IDENTITY_PROOFING {
        SELF_ASSERTION,
        REMOTE,
        IN_PERSON
    }
    enum VERIFICATION_MECHANISM{
        SMS,
        EMAIL,
        DOCUMENT
    }
    struct UserProfileField {
        string name;
        bool required;
        bool updatable;
        bool isActive;
        VERIFICATION_MECHANISM verifyMechanism;
    }
    IDENTITY_PROOFING proofingMechanism=
                    IDENTITY_PROOFING.SELF_ASSERTION;
    UserProfileField[] fields;
    event policyUpdated(string policyName,string
                           oldValue,string newValue);

    function getProofingMechanism() public returns (string){}
    function updateProofingMechanism() public {}
    function addUserProfileField(...) public { }
    function updateUserProfileField(...) public { }
    function getFields() public returns (string){}

 }
```

The *federation policy* smart contract illustrated in Listing 2 enables SPs to design identity management policies. Each marketplace can adjust these policies according to its established business practices.

Table 1 Smart contracts per module

Module	*Contracts*	*Purpose*
Federation administration	Service provider	• Register service providers • Manage available services
	Federation policy	• Manage identity proofing settings • Manage user profile settings
Federated identity management	Federation policy	• Provide user profile information
	Registrar	• Register new users • Verify pending users
	ID tracker	• Track different identities that belongs to the same user
	User	• Manage active user accounts • Reactivate accounts • Support token-based user authorisation
Federated identity tracking	Auditing	• Generate administration reports • Generate identities' usage reports • Generate services' usage reports

The identity proofing settings (IDENTITY_PROOFING in Listing 2) enable SPs to select among three methods to authenticate user information based on information sensitivity and risk degree (Temoshok and Abruzzi, 2018):

- *Self-assertion:* The self-assertion mechanism is suitable for a marketplace with low-risk transactions. A simple authentication method (e.g., username and password) can be used to authenticate users.
- *Remote identity proofing:* This mechanism works for transactions with medium risk. The user needs to provide an additional piece of information over a remote session for a stronger authentication (e.g., uploading a copy of a driver's license).
- *In-person identity proofing:* In-person proofing is the harshest method and the best candidate for high-security levels. It requires users to present themselves in person with supporting documents in front of an employee working in one of the

available IdPs. The employee must verify the documents and scan them to the shared ledger. When the physical presence of a user is not possible, virtual (remote) in-person proofing can be applied with a similar level of confidentiality (Grassi et al., 2017).

Through the user profile settings (UserProfileField structure, addUserProfileField and updateUserProfileField methods in Listing 2), SPs establish the required fields (e.g., address, phone number, e-mail). Each field can be setup as optional or mandatory, updatable or not, and a verification mechanism (VERIFICATION_MECHANISM) can be set for each field. The verification mechanism may involve an SMS, an e-mail, or uploading a verification document. Changes to policies trigger a policyUpdate event, which records changes on the blockchain. The two smart contracts illustrated in Listings 1 and 2 demonstrate the general principles of the smart contracts used in this work. The remaining four contracts follow the same pattern, and their listings are shown in Appendix.

4.2 Federated identity management module

The *federated identity management* module handles *identity issuance/registration, identity update*, and *account reactivation*. These functionalities are accomplished by four smart contracts: *federation policy*, *registrar*, *ID tracker*, and *user* as shown in Table 1. The following subsections describe the collaboration of these four smart contracts to deliver the required identity management functionalities.

4.2.1 Identity issuance/registration

The *registration* function enables users to register with the blockchain marketplace and to obtain a unique digital federated identity. Users interact with BFIF's DAPP to initialise the registration process. Then the federation policy, registrar, ID tracker, and user smart contracts interact with each other to complete the process. For instance, if the remote identity proofing registration scenario has been selected, the registration process follows the steps illustrated in Figure 3:

1 The DAPP interacts with the *federation policy* smart contract to retrieve the required fields (*getFields*) for registration and to build the registration form (*buildRegisterForm*) for a user to input data.

2 The user enters the requested information (*setRegistrationData*), including a username and password. In addition, the user may need to submit confidential documents according to the federation policy.

3 The DAPP creates a new account for the user (*createAccount*) together with a private/public key pair. The private key is stored on the user device (*storePrivateKey*), whereas the public key is stored on the blockchain. Next, the DAPP invokes the *registrar* smart contract to register the new user (*registerUser*) with the federation.

4 If no further verification were needed (e.g., in-person identity proofing), the *user* smart contract is invoked (*createUser*) to create a digital identity

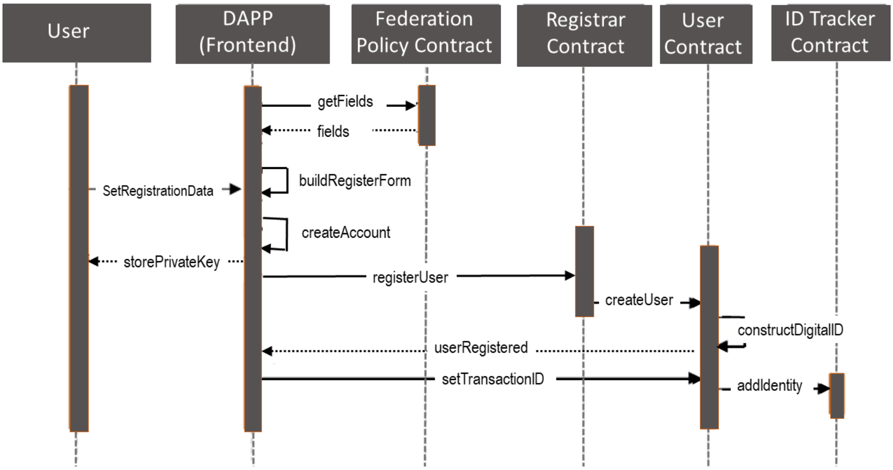
(*constructDigitalId*) in the form of the hash value of the user's profile fields. If further verification were needed, the *registrar* smart contract would add the user in the pending users list and wait for user verification through one of the available SPs. Once the identity is created, the transaction is created on the blockchain, and the *user* smart contract emits an event notifying the DAPP about the completion of identity creation (*userRegistered*).

5 The DAPP extracts the blockchain transaction ID and invokes the *user* smart contract (*setTransactionID*) to update the user profile.

6 Finally, the *ID tracker* smart contract is invoked to add the transaction ID to the records of identities.

It is important to highlight the roles of public/private key pairs. Users' public keys, as well as SPs' public keys, are stored on the blockchain. Although this increases the quantity of data stored on the blockchain, it adds an additional layer of security by ensuring these data are transported over the network encrypted, provide all blockchain nodes with access to user's and provider's public keys, and place these keys under the blockchain security mechanisms. As illustrated in Figure 4, any data sent from the user are encrypted using the user's private key. On the blockchain network, these data are decrypted using the user's public key, then encrypted using the SP's public key and sent to the SP. Finally, the SP decrypts the data using the SP's private key. This approach of using two private/public key pairs was chosen over a single key pair to enable the blockchain to deal with unencrypted data and apply the blockchain mechanisms for the storage.

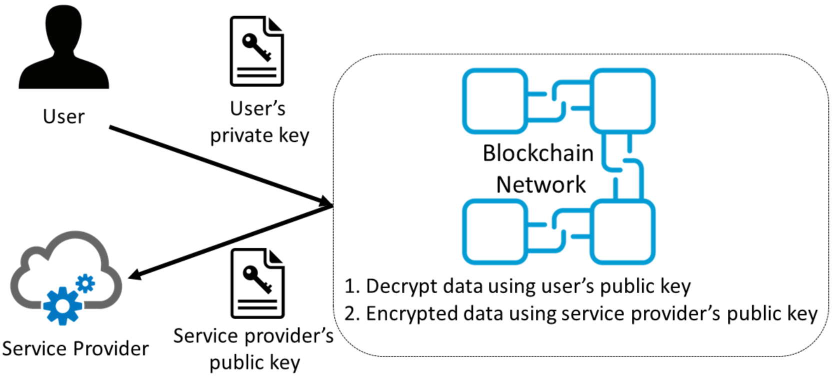
Figure 3 Remote identity proofing registration process (see online version for colours)



4.2.2 Identity update

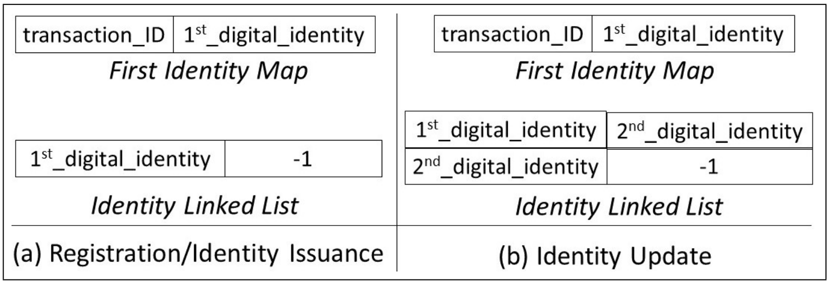
The *identity update* function is initiated when the users want to modify their profile information or submitted documents, for example, replacing the passport with a driver's

Figure 4 Data encryption in BFIF (see online version for colours)


User
User's private key
Blockchain Network
1. Decrypt data using user's public key
2. Encrypted data using service provider's public key
Service Provider
Service provider's public key

license or simply adding a second identification document. This phase is present in both remote identity proofing scenarios and in-person identity proofing. A new federated identity is created when new confidential documents are introduced. The *ID tracker* smart contract keeps track of users' identities (Figure 5). The change is captured using two maps: the first map connects users to their federated identity constructed during the registration process, and the second one constructs a linked list of federated identities that are associated with the same user.

Figure 5 Data structures in ID tracker smart contract



(a) Registration/Identity Issuance

First Identity Map

transaction_ID	1st_digital_identity

Identity Linked List

1st_digital_identity	-1

(b) Identity Update

First Identity Map

transaction_ID	1st_digital_identity

Identity Linked List

1st_digital_identity	2nd_digital_identity
2nd_digital_identity	-1

As illustrated in Figure 5(b), the second map is updated when a new federated identity is produced. The map stores the chain of identity changes for that user. The latest identity is the only active identity, identified by –1 for its second entry. The private and public keys remain the same, and the blockchain triggers an event to inform the user about the identity changes.

4.2.3 Account reactivation

Users may lose their identities or private keys due to malicious attacks or simply by forgetting, resulting in an inability to access their blockchain accounts and services. However, if an intruder gains access to the user's federated identity and tries to access the blockchain network, notifications are sent to the original user along with any transactions that occur using that federated identity.

If a breach occurs, a user can block the account and ask for an account reactivation. A user can initialise this by answering specific questions related to the federated identity profile or use of federation services.

Algorithm 1 Account reactivation algorithm

```
▷ Input userName (uname), newAccount (acc), questionAnswers (answers)
procedure REACTIVATEACC($uname, acc, answers$)
    $oldUser \leftarrow findUser(uname)$
    if $verifyUser(answers)$ then
        ▷ copy data from the old user to the new one
        $newUser \leftarrow createUser(acc, oldUser)$
        $deactivateUser(oldUser)$
        return true
    else
        return false
```

Whenever the user submits the verification answers, the DAPP creates the new user account and submits the request to the *user* smart contract for fulfillment, as described in Algorithm 1. First, the old user account is located, and access verified based on answers to user-specific questions. Next, a new user account is created from the old user's data. Finally, the old account is deactivated.

4.2.4 Federated identity authorisation

The goal of the authorisation process is to grant users access to one of the SPs' managed services by authenticating their federated identities. The *user* smart contract provides a SSO authorisation to give users seamless access to the services provided by the participating parties. Whenever the user logs in, a token is generated and used to grant access to the service. If the user tries to access the same service again or a different service and the token exists, the user is redirected directly to the service without login prompts.

Each access attempt is represented by a transaction and is verified by the nodes in the blockchain network and recorded in the blockchain ledger. Besides, once access has been granted, a blockchain event is triggered to notify the user.

4.3 Federated identity tracking module

Federated identity enables users to access online services allocated in a marketplace. These services encapsulate the processes of business parties and are usually developed as stand-alone components regardless of the technical structure of the encompassing business environment. Hence, auditing of services running inside business environments such as a cloud or service-oriented architecture (SOA) are addressed in the defined blockchain-based marketplace. Tejpar (2010) has discussed several dimensions that an auditing process in a business service-based environment (e.g., SOA) should fulfill. The proposed auditing process running within BFIF framework enables five auditing dimensions, as shown in Figure 6.

- *Record keeping* is the first dimension and the key aspect of the auditing process. According to NIST standards (Swanson and Guttman, 1996), record keeping can be seen as a digital log that should track the user associated with an event (who), the event itself (what), the event timestamp (when), and the causes that trigger that event (how). Similarly, in the blockchain technology, the transaction history can be seen as an audit log. It keeps track of the transaction's parties (who), information about the transaction (what), transaction timestamp (when), and the transaction type (how). For instance, in BFIF, a transaction type can be identified as a smart contract function call that aims to accomplish a particular task (e.g., register a new user or access a business service).

- *Lifetime*, or the duration of the audit data, represents the second dimension of auditing. Notably, the blockchain, and consequently BFIF based on it, is by design an immutable distributed ledger where all transactions are permanently stored securely; recorded transactions cannot be updated or deleted, but only new transactions can be added.

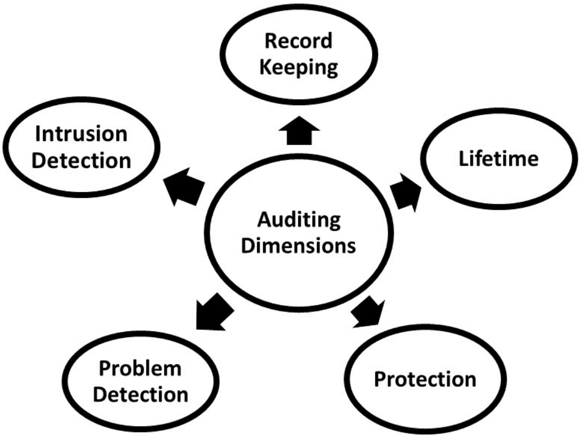
- *Protection*, the third auditing dimension, is achieved through the tamper-proof characteristics of blockchain data. Only identified peers inside the blockchain network can have read-only access rights to the audit logs. All recorded transactions are encrypted by cryptographic keys that protect them against any unauthorised altering.

- *Problem detection* is the next auditing dimension. Exploring and scanning audit logs can help to identify any potential problems that might have occurred. For example, in the case of incorrect logic inside a smart contract in BFIF, the audit logs can help to trace and identify what has been run incorrectly. Moreover, the blockchain technology maintains a history of all failed transactions and their correlated exceptions. This information could also be used to analyse and solve hidden problems during the implementation of different processes in BFIF.

- *Intrusion detection* is the last, but not the least dimension. Again, the audit logs can also be used to detect intrusion into the constructed blockchain-based marketplace. For instance, an intruder might gain access to users' federated identities or any relevant data. Intrusion detection often requires detailed data analysis of stored logs to obtain an accurate picture of how the intrusion was accomplished. For example, if an intruder compromises the private key and the federated identity of a user and uses them to access the blockchain network, the audit logs in BFIF can be analysed to follow the intrusion and its impact. Some intrusions can also be detected in real-time, in which case the transaction will be marked as invalid and will not go through the blockchain network.

In BFIF, the *auditing* smart contract is responsible for implementing the auditing functionalities mentioned above. First, it enables SPs to add/remove auditors' accounts to/from the blockchain network. Second, the *auditing* contract permits the auditors to generate readable reports for both users and business parties. Users' reports contain tracks of each use of users' credentials to access business services, modifications to their profiles, and updates to their digital identities. The business party report contains records about service access and updates to federated identity policies.

Figure 6 Auditing dimensions



5 Prototype implementation

As a proof of concept and to evaluate performance, two demonstration prototypes have been developed, one on the public and one on the permissioned blockchain network. The Hyperledger Fabric and the Ethereum platforms were chosen as representatives of the public and permissioned blockchain networks. Moreover, as those platforms use different consensus protocols, it is expected that they will differ in performance characteristics. For both prototypes, two servers with the following specifications were used: Intel i7 3.5 GHz, 250 GB solid-state drive, 32 GB RAM, with Ubuntu-16.04. Docker (2019) images were used to setup multiple blockchain nodes on those two machines.

- Public blockchain network: A test network has been constructed based on the Ethereum protocol (Wood, 2017). Ethereum is a peer-to-peer network where every peer stores a shared ledger and runs an Ethereum virtual machine to maintain the network state. The creation of a new block in Ethereum requires that all members of the network conform to the proof-of-work consensus protocol. With this protocol, nodes compete against each other to complete transaction validation and get a reward. The Ethereum test network has four nodes: one boot-node and three miner-nodes. The boot-node does not keep any state of the blockchain; it helps other nodes in the network to find each other. Miners are nodes that create blocks inside the blockchain network.
- Permissioned blockchain network: The Hyperledger Fabric (Hyperledger, 2019) with Identity Mixer (Camenisch et al., 2013) implementation has been used to develop the permissioned test network with one ordering service node, one Fabric Certificate Authority Server node, one anchor node and three peer nodes. The ordering service node is responsible for ordering transactions on a first-come, first-serve basis among the connected peers within the established network. The Fabric Certificate Authority Server node represents a root certificate authority for managing the digital identities of fabric participants (e.g., users and SPs). The anchor peer node is used to initiate gossip communication between peers. The peer nodes are responsible for processing transactions and maintaining the state

and a copy of the ledger. The consensus in Hyperledger Fabric is based on the practical Byzantine fault tolerance protocol, in which ordering service nodes agree on the system's state (Androulaki et al., 2018).

Figure 7 Prototypes' application stack (see online version for colours)

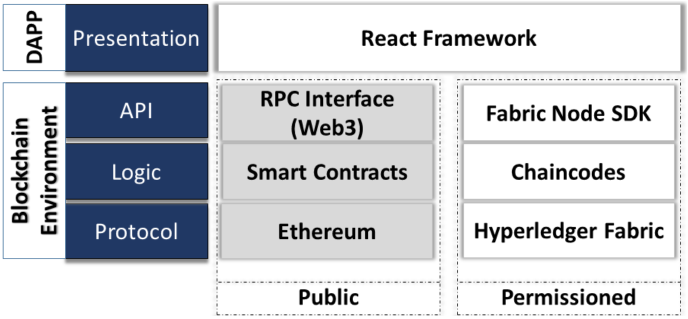


Figure 7 shows the two prototypes' application stack. The blockchain environment consists of three layers: protocol, logic, and API. The three layers are implemented differently for the public and permissioned networks. The presentation layer is the same for both approaches: the react framework (Facebook Inc., 2019) was used to build the DAPP front-end.

6 Results and discussion

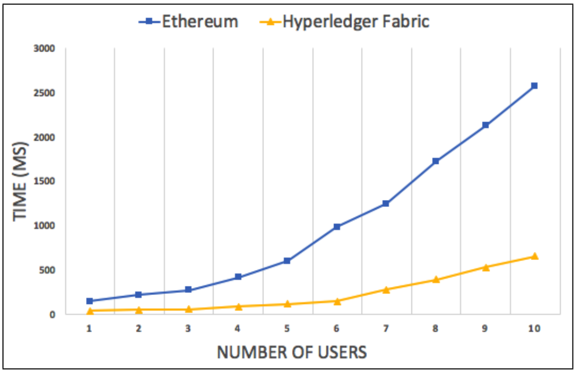
Like commonly known blockchain applications including cryptocurrency blockchains (e.g., Bitcoin), the proposed BIFF tracks the exchanged transactions among the involved participants while keeping immutable records of those transactions. However, the proposed BIFF does not exchange currency between participants, but instead BIFF transactions involve creating federated identities and accessing services provided by SPs. The processes are governed by customisable policies and rules encapsulated in smart contracts. This kind of sharing can be accomplished publicly among all the involved participants or through private sessions between particular business participant according to the sensitivity level of the sharing attributes.

The core performance aspects of federated identity management systems involve performance of registration and authorisation services because those services represent the majority of transactions in identity management systems; services such as *revocation* and *identity update* only happen occasionally and do not have a major impact on the overall performance. Consequently, this work evaluates BFIF performance on those two services. Figures 8 and 9 show the performance of the registration and authorisation services for a number of concurrent requests (users) ranging from one to ten for the two implementations.

It is clear that Hyperledger Fabric outperformed the Ethereum public network. With ten concurrent requests, Hyperledger Fabric was about four times faster than Ethereum for both registration and authorisation services.

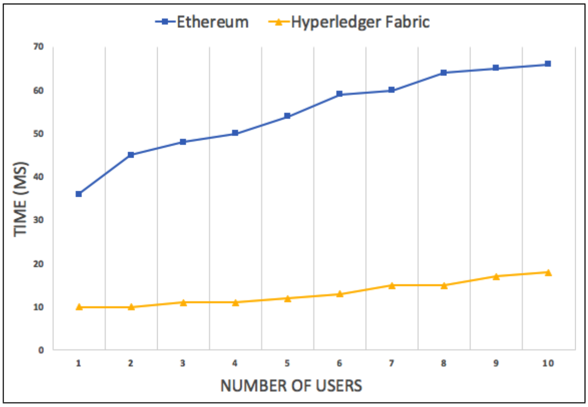
This difference is due to the nature of the consensus protocol and the transaction lifecycle in the two networks (Androulaki et al., 2018; Moubarak et al., 2017).

Figure 8 Registration service evaluation (see online version for colours)



Ethereum uses the proof of work consensus protocol: all participants must carry out time-consuming computation in the process of achieving a consensus, which negatively affects service performance. On the other hand, Hyperledger Fabric uses the practical Byzantine fault tolerance consensus protocol, where ordering service nodes come to an agreement on the state of the system. This algorithm does not require extensive computations like the proof of work, and only a subset of nodes (the ordering peers) must reach consensus. Consequently, Hyperledger achieves much better performance than Ethereum, as illustrated in Figures 8 and 9.

Figure 9 Authorisation service evaluation (see online version for colours)



The second reason that Hyperledger Fabric outperforms Ethereum is the difference between their transaction lifecycles. Ethereum uses the order-execute architecture to update the shared ledger, where all submitted transactions are ordered and executed on all peers sequentially. On the other hand, Hyperledger Fabric uses three phases to achieve the same goal: execution, ordering, and validation (i.e., updating the ledger). The three-phase approach enables peers to execute submitted transactions in parallel, which helps to improve network throughput.

The use of blockchain technology supports the proposed federated identity management system with several benefits, including:

- *Transparency:* Designed smart contracts in BFIF trigger blockchain events to send notifications to users about the use or update of their credentials.
- *Consistency:* BFIF keeps the distributed ledger consistent using the consensus protocol. The user can be validated at any node within the blockchain network.
- *User privacy:* BFIF addresses users' privacy through anonymity: individuals in BFIF are represented by anonymous keys, which help users hide their private information and credentials.
- *Auditability:* BFIF provides an auditing mechanism that makes it possible to generate reports about the use of identities and services. Transactions become part of blocks; validated blocks are added to the shared ledger and consequently become immutable.
- *Availability:* BFIF enables users to access services from any node in the network, so that if a single node fails, the user will be able to access the blockchain from another node. This capability is supported through the blockchain replication features.

Moreover, BFIF implementation using the Hyperledger Fabric with Identity Mixer module improves privacy in two dimensions: unlinkability and data exposure minimisation. When each user is presented with an anonymous key, as is the case with Ethereum-based BFIF, this key could be used to trace all transactions performed by the user. However, the membership service provider (MSP) with Identity Mixer in Hyperledger Fabric helps to avoid such linkability (Camenisch et al., 2013) by providing different presentation tokens for the same key. The Identity Mixer algorithm also makes it possible to minimise the amount of user data that need to be revealed to access a service by helping to verify facts about an attribute without revealing its actual value. For example, in a driver's license scenario, when a service should be accessible only by users over a certain age, the Identity Mixer validates the birth date of the signed-in user inside the blockchain before passing a confirmation that this user meets the age restrictions without exposing his/her actual birth date.

The proposed framework, because of its foundation on the blockchain, inherits blockchain security and privacy characteristics. A number of studies focused on blockchain security and privacy: Joshi et al. (2018) reviewed the security and privacy of blockchain technologies, Jesus et al. (2018) discussed how blockchain secures internet of things, and Meng et al. (2018) examined the blockchain in respect to intrusion detection. The proposed framework assists in defending against various types of attacks such as:

- *Data tampering and eavesdropping:* The threat of illegal or unauthorised tampering or interception of user data is reduced through the immutability of the shared ledger and the data encryption feature.
- *Phishing and spoofing:* The username and password could be stolen by redirecting the user to input them in a visually similar interface or by advanced means of interception such as keystroke monitoring. In both cases, the intruder will not be able to access the blockchain through BFIF because the user account is also protected with another level of security, which consists of the private-public keys generated by the blockchain itself.

- *Repudiation:* Users cannot deny any of the actions that they have already performed due to the immutability of the distributed ledger and the auditing mechanism provided by BFIF.

The use of blockchain technologies in the proposed framework offers a variety of advantages as discussed; however, those technologies pose a number of challenges. The most prominent one is scalability. Public networks such as Bitcoin and Ethereum use the proof of work consensus protocol, which performs time-consuming transaction validation. Moreover, those networks require all full nodes to validate each submitted transaction. However, in the case of the Hyperledger Fabric, the permissioned network can scale much better because it uses the practical Byzantine fault tolerance consensus protocol and each node has a different business role such as endorser, committer, or consenter (Moubarak et al., 2017). The improved performance was also demonstrated in the experiments presented in this paper (Figures 8 and 9). Furthermore, researchers are continuously working on improving the scalability of public blockchain networks, including Ethereum and Bitcoin (Chauhan et al., 2018b, 2018a; Kim et al., 2018).

Smart contracts, like any other programmed software components, are exposed to possible software faults that may lead to different consequences. The auditing mechanism in BFIF provides the ability to review past smart contract operations as well as the capability of deploying a new version.

If the private key is compromised together with the login information (username and password), then the attacker can access the services. For such situations, BFIF introduced the account revocation feature so that the user can revoke the account if notified about an access that he/she does not recognise, as explained in Subsection 4.2.3.

7 Conclusions

The work in this paper describes a generic framework for blockchain-based federated identity and auditing. Our study has introduced an automatic mechanism for issuing and tracking federated identity in a distributed fashion for users to access online services belonging to different business parties in a marketplace. The framework uses the blockchain's smart contracts to manage the processes of federated identity creation and validation. The proposed framework also presents an auditing process to enable both users and business parties to track all transactions relevant to their activities. The performance of the framework has been tested on public (e.g., Ethereum) and permissioned (e.g., Hyperledger Fabric) blockchain environments. The results show that the framework performs better in the permissioned blockchain environment because of differences in transactions validation and ledger block construction.

While BFIF presented here includes auditing the creation, modification, and use of federated identities, future work will expand this by designing and developing a complete blockchain-based framework for any data and/or services linked to the underlying blockchain network. Data access (including read and write) or service use will be recorded on the blockchain, consequently enabling provenance, supporting access violation detection, or assisting with other audit-related tasks.

This work presented a prototype with two blockchain technologies, Ethereum and Hyperledger Fabric, and performed evaluations with a small number of users. Further experiments are needed to investigate how the framework will behave with thousands of users. Moreover, with a large number of users, many transactions and audit records, the blockchain itself will grow and studies are required to evaluate how this will affect blockhain-based software applications.

Presently there are many blockchain-based initiatives and platforms with very little standardisation or direct comparison (Andoni et al., 2019). To allow interoperability among solutions and to support selection of adequate technology for a specific use-case, standardisation and consistent evaluation procedures are needed.

References

Andoni, M., Robu, V., Flynn, D., Abram, S., Geach, D., Jenkins, D., McCallum, P. and Peacock, A. (2019) 'Blockchain technology in the energy sector: a systematic review of challenges and opportunities', *Renewable and Sustainable Energy Reviews*, February, Vol. 100, pp.143–174.

Androulaki, E., Barger, A., Bortnikov, V., Cachin, C., Christidis, K., De Caro, A., Enyeart, D., Ferris, C., Laventman, G., Manevich, Y., Muralidharan, S., Murthy, C., Nguyen, B., Sethi, M., Singh, G., Smith, K., Sorniotti, A., Stathakopoulou, C., Vukolić, M., Weed Cocco, S. and Yellick, J. (2018) *Hyperledger Fabric: A Distributed Operating System for Permissioned Blockchains*, ArXiv e-prints, January.

Azaria, A., Ekblaw, A., Vieira, T. and Lippman, A. (2016) 'MedRec: using blockchain for medical data access and permission management', *2016 2nd International Conference on Open and Big Data (OBD)*, August, pp.25–30.

Bertino, E. and Takahashi, K. (2011) *Identity Management: Concepts, Technologies, and Systems*, Artech House, ISBN: 978-1608070398.

Bhosale, S.K. (2008) 'Architecture of a single sign-on (SSO) for internet banking', in *2008 IET International Conference on Wireless, Mobile and Multimedia Networks*, January, pp.103–105, DOI: 10.1049/cp:20080155.

Camenisch, J., Dubovitskaya, M., Lehmann, A., Neven, G., Paquin, C. and Preiss, F-S. (2013) 'Concepts and languages for privacy-preserving attribute-based authentication', in Fischer-Hübner, S., de Leeuw, E. and Mitchell, C. (Eds.): *Policies and Research in Identity Management*, pp.34–52, Springer, Berlin, Heidelberg, ISBN: 978-3-642-37282-7.

Canadian National Bank Insurance (2015) *Statistics on Identity Theft* [online] https://www.nbc-insurance.ca/content/bna/en/accueil/avantages-et-conseils/statistiques-vol-identite.html (accessed 18 February 2020).

Chauhan, A., Malviya, O., Verma, M. and Mor, T. (2018a) 'Blockchain and scalability', in *2018 IEEE International Conference on Software Quality, Reliability and Security Companion, QRS Companion 2018*, Lisbon, Portugal, pp.122–128, DOI: 10.1109/QRS-C.2018.00034.

Chauhan, A., Malviya, O.P., Verma, M. and Mor, T.S. (2018b) 'Blockchain and scalability', in *2018 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, July, pp.122–128, DOI: 10.1109/QRS-C.2018.00034.

Dinh, T.T.A., Liu, R., Zhang, M., Chen, G., Ooi, B.C. and Wang, J. (2017) *Untangling Blockchain: A Data Processing View of Blockchain Systems*, ArXiv e-prints, August [online] https://arxiv.org/abs/1708.05665.

Dinh, T.T.A., Liu, R., Zhang, M., Chen, G., Ooi, B.C. and Wang, J. (2018) 'Untangling blockchain: a data processing view of blockchain systems', *IEEE Transactions on Knowledge and Data Engineering*, Vol. 30, No. 7, pp.1366–1385.

Docker (2019) *Docker Container Solution* [online] https://www.docker.com/ (accessed 18 February 2020).

Dunphy, P. and Petitcolas, F.A.P. (2018) *A First Look at Identity Management Schemes on the Blockchain*, ArXiv e-prints, January [online] https://arxiv.org/abs/1801.03294.

Ebrahimi, A. (2016) *Identity Management Service Using A Blockchain Providing Certifying Transactions Between Devices*, November [online] http://www.freepatentsonline.com/y2016/0330027.html (accessed 18 February 2020).

Equifax (2018) *Cybersecurity Incident and Important Consumer Information* [online] https://www.consumer.equifax.ca/canada/equifaxsecurity2017/en_ca/ (accessed 18 February 2020).

Facebook Inc. (2018) *REACT Business Application Framework* [online] https://reactjs.org/ (accessed 18 February 2020).

Faidella, D.C. and Schukai, R.J. (2017) *Methods and Systems for Identity Creation, Verification and Management*, June [online] https://patents.google.com/patent/WO2017112019A1/ru (accessed 18 February 2020).

Grassi, P.A., Fenton, J.L., Lefkovitz, N.B., Danker, J.M., Choong, Y-Y., Greene, K.K. and Theofanos, M.F. (2017) *Digital Identity Guidelines: Enrollment and Identity Proofing*, Technical report, June [online] https://doi.org/10.6028/nist.sp.800-63a.

Hyperledger (2019) *Hyperledger Fabric Blockchain Framework* [online] https://www.hyperledger.org/projects/fabric (accessed 18 February 2020).

Javelin Strategy and Research (2017) *Identity Fraud Hits Record High with 15.4 Million U.S. Victims in 2016, Up 16 Percent According to New Javelin Strategy and Research Study* [online] https://goo.gl/Mq1TpR (accessed 18 February 2020).

Jesus, E., Chicarino, V., de Albuquerque, C. and Rocha, A. (2018) 'A survey of how to use blockchain to secure internet of things and the stalker attack', *Security and Communication Networks*, pp.1–27, DOI: 10.1155/2018/9675050.

Joshi, A., Han, M. and Wang, Y. (2018) 'A survey on security and privacy issues of blockchain technology', *Mathematical Foundations of Computing*, Vol. 1, No. 2, pp.121–147, ISSN: A0000-0001, DOI: 10.3934/mfc.2018007.

Kikitamara, S. (2017) *Digital Identity Management on Blockchain for Open Model Energy System*, Master's thesis, Radboud University [online] http://www.ru.nl/publish/pages/769526/digital_identity_management_on_blockchain_final.pdf.

Kim, S., Kwon, Y. and Cho, S. (2018) 'A survey of scalability solutions on blockchain', in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, October, pp.1204–1207, DOI: 10.1109/ICTC.2018.8539529.

Lee, J-H. (2018) 'BIDaaS: blockchain based ID as a service', *IEEE Access*, February, Vol. 6 [online] http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=8187625.

Malik, A.A., Anwar, H. and Shibli, M.A. (2015) 'Federated identity management (FIM): challenges and opportunities', in *2015 Conference on Information Assurance and Cyber Security (CIACS)*, December, pp.75–82, DOI: 10.1109/CIACS.2015.7395570.

Meng, W., Tischhauser, E.W., Wang, Q., Wang, Y. and Han, J. (2018) 'When intrusion detection meets blockchain technology: a review', *IEEE Access*, Vol. 6, pp.10179–10188, ISSN: 2169-3536, DOI: 10.1109/ACCESS.2018.2799854.

Moubarak, J., Filiol, E. and Chamoun, M. (2017) 'Comparative analysis of blockchain technologies and tor network: two faces of the same reality?', in *2017 1st Cyber Security in Networking Conference (CSNet)*, October, pp.1–9, DOI: 10.1109/CSNET.2017.8242004.

Nakamoto, S. (2009) *Bitcoin: A Peer-to-Peer Electronic Cash System* [online] http://www.bitcoin.org/bitcoin.pdf (accessed 18 February 2020).

ShoCard (2019) *ShoCard: Secure Enterprise Identity Authentication* [online] https://shocard.com/ (accessed 18 February 2020).

Sovrin Foundation (2019) *Sovrin Identity For All* [online] https://sovrin.org/ (accessed 18 February 2020).

Swanson, M. and Guttman, B. (1996) *Generally Accepted Principles and Practices for Securing Information Technology Systems*, Technical report, September [online] https://csrc.nist.gov/publications/detail/sp/800-14/final.

Tang, C., Wu, L., Wen, G. and Zheng, Z. (2019) 'Incentivizing honest mining in blockchain networks: a reputation approach', *IEEE Transactions on Circuits and Systems II: Express Briefs*, January, Vol. 67, No. 1, pp.117–121.

Tejpar, R.R. (2010) *Auditing Framework for Event-Driven Service-Oriented Architecture*, Master's thesis, Western University.

Temoshok, D. and Abruzzi, C. (2018) *Developing Trust Frameworks To Support Identity Federations*, Technical report, January [online] https://doi.org/10.6028/NIST.IR.8149.

uPort Inc. (2018) *uPort Identity* [online] https://www.uport.me/ (accessed 18 February 2020).

U.S. Department of Justice (2017) *Victims of Identity Theft*, Revised 13 November [online] https://www.bjs.gov/content/pub/pdf/vit14.pdf. (accessed 18 February 2020)

Wall Street Journal (2017) *Yahoo Triples Estimate of Breached Accounts to 3 Billion* [online] https://goo.gl/eaYvgd (accessed 18 February 2020).

Wang, E.K., Liang, Z., Chen, C-M., Kumari, S. and Khan, M.K. (2020) 'PoRX: a reputation incentive scheme for blockchain consensus of IIoT', *Future Generation Computer Systems*, January, Vol. 102, pp.140–151.

Werner, J., Westphall, C.M. and Westphall, C.B. (2017) 'Cloud identity management: a survey on privacy strategies', *Computer Networks*, Vol. 122, pp.29–42, ISSN: 1389-1286 [online] https://doi.org/10.1016/j.comnet.2017.04.030.

Wolfond, G. (2017) 'A blockchain ecosystem for digital identity: improving service delivery in Canada's public and private sectors', *Technology Innovation Management Review*, December, Vol. 7, No. 10, pp.35–40 [online] https://timreview.ca/issue/2017/october (accessed 18 February 2020).

Wood, G. (2017) *Ethereum: A Secure Decentralised Generalised Transaction Ledger (EIP-150 Revision)* [online] https://files.gitter.im/ethereum/yellowpaper/VIyt/Paper.pdf (accessed 18 February 2020).

Xia, Q., Sifah, E., Smahi, A., Amofa, S. and Zhang, X. (2017) 'BBDS: blockchain-based data sharing for electronic medical records in cloud environments', *Information*, April, Vol. 8, No. 4, p.44, DOI: 10.3390/info8020044.

Zyskind, G., Nathan, O. and Pentland, A. (2015) *Enigma: Decentralized Computation Platform with Guaranteed Privacy*, ArXiv e-prints, June [online] https://arxiv.org/abs/1506.03471.

Appendix

Listing 3 Registrar smart contract

```
contract RegistrarSC{
  struct User{
      address userAddr;
      string publicKey;
      //mapping the field to its value
      mapping(string => string) fields;
  }
  mapping (address => User) pendingUsers;
  event userPending(address userAddr);

  function addFields(string userAddr, string fields) public {}
  function getFields(address userAddr)
         public returns (string){}
  function registerUser(string userAddr, string publicKey)
         public returns (bool) {}
  function verifyUser(address userAddress)
         public { }
  function getPendingUsers() public
         returns (string){}
}
```

Listing 4 User smart contract

```
contract UserSC{
  struct User{
      address userAddr;
      string publicKey;
      string digitalIdentity;
      string transactionID;
      //fieldName -> value
      mapping(string => string) fields;
  }
  struct Token{
       string username;
       bytes32 token;
       uint lifetime;
  }
  //mapping username to User
  mapping (string => User) users;
  mapping(bytes32 =>Token) tokens;
  event userRegistered(string username);
  event userAuthorised(string digitalId, bytes32 token,
                       string url);
  function createUser(string userAddr, string publicKey,
          string fields) public{}
  function setTransactionID(string username,
          string transactionHash);
  function updateFields(string fields) public {}
  function getFields() public returns (string) {}
  function reactivateAccount(string username, string userAddr,
         string public key, string answers) public returns (bool){}
  function authoriseUser(string serviceId, string username,
          string pass) public returns (bool){}
  function validateToken(string serviceId, bytes32 token)
          public returns (string){}
}
```

Listing 5 ID tracker smart contract

```
contract IdTrackerSC{
   //mapping transactionHash to first_identity
   mapping(string => string) firstIdentityMap;

   /* mapping first_identity to second_identity
     and last_identity to -1 */
```

```
    mapping(string => string) IdentityLinkedList;
    event identityUpdated(string oldIdentity,
          string newIdentity);

    function addIdentity(string  transactionHash,
          string identity) public { }
    function getUserIdentitiesList(string transactionHash)
          public returns(string){}
 }
```

Listing 6 Auditing smart contract

```
 contract AuditingSC{
    struct AuditTrail {
       string  userIdentity;
       address serviceProvider;
       AUDIT_TASK task;
       uint timestamp;
    }
    enum AUDIT_TASK{
        REGISTRATION,
        AUTHENTICATION,
        //track update of user data
        DATA_UPDATE,
        //track update in federation policies
        POLICY_UPDATE
    }
   struct Auditor{
       string name;
       bool isActive;
   }
   mapping(address => Auditor) auditors;
   modifier onlyAuditor() {
        require(auditors[msg.sender].isActive);
        _;
   }

   function addAuditor(address auditorAddr,...)
        public {}
   function removeAuditor(address auditorAddr) public {}
   function addAuditTrail(string userIdentity,...) public{}
   //report about policy updates
   function getAdminReport (uint startTime, uint endTime)
        public onlyAuditor returns (string){ }
   function getServiceProviderReport(address providerAddr,
        uint startTime, uint endTime)
        public onlyAuditor returns (string){ }
   function getUserReportAudit(string userIdentity, ...)
        public onlyAuditor returns (string){ }
 }
```