


Chain-based big data access control infrastructure

Emmanuel Boateng Sifah¹ · Qi Xia²  ·
Kwame Opuni-Boachie Obour Agyekum¹ · Sandro Amofa¹ · Jianbin Gao³ ·
Ruidong Chen⁴ · Hu Xia⁴ · James C. Gee^{5,6} · Xiaojiang Du⁷ ·
Mohsen Guizani⁸

Published online: 14 March 2018

© Springer Science+Business Media, LLC, part of Springer Nature 2018

Abstract Technological advancements have brought about the rise of data and other digital assets in our world today. The major problems with data today are its security and management, more importantly access control. These factors when not tackled

✉ Qi Xia
xiaqi0769@gmail.com; xiaqi@uestc.edu.cn

Emmanuel Boateng Sifah
emmanuelsifah@yahoo.com

Kwame Opuni-Boachie Obour Agyekum
obour539@yahoo.com

Sandro Amofa
ikwame.amofa@gmail.com

Jianbin Gao
gaojb@uestc.edu.cn

Ruidong Chen
crdchen@163.com

Hu Xia
Xiahu@youedata.com

James C. Gee
gee@uestc.edu.cn

Xiaojiang Du
dxj@ieee.org

Mohsen Guizani
mguizani@ieee.org

¹ School of Computer Science and Engineering, UESTC, Chengdu, China

² Center for Cyber Security, UESTC, Chengdu, China

³ School of Resource and Environment, Center for Digital Health, UESTC, Chengdu, China

effectively can lead to many compromises. The blockchain is an effective technology that ensures utmost security, trust, and maximum access control in big data systems. However, almost all the transactions on a blockchain network are stored in the platform. This process reduces the data storage, as the storage of all transactions sometimes creates unnecessary overheads. In this paper, an off-chain-based sovereign blockchain is proposed, where a virtual container is created for parties to transact in. At the end of a transaction, and satisfying each party, the container is destroyed but the results are stored on the sovereign blockchain network. This effectively decreases the amount of data that would have been stored on the network. The effectiveness of our system is compared with other schemes, and we could infer that our proposed system outperforms the already-existing ones.

Keywords Big data · Sovereign blockchain · Access control · Smart contracts

1 Introduction

The entrance of blockchain technology has forever changed the way data are perceived, accessed, controlled, and managed [1, 2]. With applications to everything data, the technology is currently being adapted to suit several fields where trust and other challenges that usually malign collaboration come into play [3, 4]. The attraction of the trust-less or trust-minimized environments they create notwithstanding, blockchains are generally inefficient processing platforms for data [5, 6]. With the Internet of Things (IoT) and Big Data still driving up data amounts, it has become necessary to find practical and efficient means of accessing data [7, 8], handling all its necessary computations [9], storage, and management without sacrificing security and the ability to collaborate [10–12]. We thus propose the sovereign blockchain, a platform for the control and regulation of transaction behaviors and other attributes.

The lack of speed is a difficult challenge with blockchains. The bitcoin blockchain by design takes an average of 10 min for the creation of new blocks, and about six blocks layered on top of it for transactions to be deemed to have settled [12–14]. This is for economic reasons so that miners with greater computing power [15] do not exhaust the cryptocurrency's supply before a desired set time [16–18]. It is an entirely different matter for other blockchains that serve other purposes. Consider a blockchain that is engineered to facilitate collaboration between critical systems in real time [19, 20]. An hour for confirmation of or servicing of a request may be deemed unacceptable. If we consider collaboration between government departments that are data intensive,

⁴ Youe Data Co. Ltd., Beijing, China

⁵ Department of Radiology, University of Pennsylvania, Philadelphia, USA

⁶ School of Computer Science and Engineering, UESTC, Chengdu, China

⁷ Department of Computer and Information Sciences, Temple University, Philadelphia, USA

⁸ Department of Electrical and Computer Engineering, University of Idaho, Moscow, ID 83844, USA

suffice it to say correctness of data, efficiency of processes, and accountability of users are of critical importance and the life cycle of data would have to be managed in compliance with applicable frameworks.

Aside speed, there is also the constraint of data security. Transmission, updating, and other operations affect the security of data through exposures of business logic and proprietary algorithms generated at each turn. Unauthorized disclosures, access and modification, inconsistencies, and errors are just a few of the threats data face.

In this work, we propose a sovereign blockchain infrastructure to provide security management and necessary logic to secure data blocks and derive greater value from already-processed data by a mechanism referred to in this work as recycling processing. We propose a novel data processing structure by initiating suitable data formats known as BlockData to inhibit the transmission of data by custodians for data processing. The design of this structure implies that data will reside in its original repository and would be encapsulated in a BlockData format so as to preserve the originality of the data before, during, and after processing. To support BlockData processing, we provide a secure data processing environment known as a virtual computational workspace. Integrating these pieces together results in the sovereign blockchain infrastructure to derive great value for the business logic of a system.

2 Background

The background describes the key technologies to foster the understanding based on blockchain implementation on big data environments. Key technologies such as the blockchain and smart contracts aiding in the design of a suitable solution to the already-stated problem are discussed.

2.1 Sovereign blockchain

The sovereign blockchain is identified by a point-to-point immutable and reliable transfer of asset value preceding other blockchain systems. In describing the governance of its sovereignty, it stresses the fact that the connected community of nodes adheres to the rules of state regarding the sovereignty of connections between nodes. This allows for the delivery of publicly owned values under the framework of economies and other infrastructural developments.

For regulations, the sovereign blockchain enforces on policies that networks and accounts should be monitored, controlled and regulation nodes should be efficient and technically feasible. With regard to the structure of the network, the dispersive and multicentered characteristics of the network are enhanced. There is also the technical provision of identity authentication and efficient account management for each of the nodes in the network's sovereign system.

The consensus algorithm is also emphasized in this system, by creating a greatest common divisor between the intention and the requirement for each of the nodes and providing the ability to inculcate multiple consensus algorithms, rather than focusing on efficiency. There is also a value measurement system provided in such systems that balances social value with material wealth incentive. Smart contracts are also

embedded in these platforms to regulate how participants on the network are supposed to transact in order to provide maximum transparency and trust.

2.2 Smart contracts

Smart contracts are computerized algorithms that bind the participants on a platform with regard to a particular transaction involving the parties. In this paper, smart contracts are employed to report transaction details on data to the data block. Aside reports, logs on every transaction are also generated and sent to the processing node(s), to be forwarded and kept on the blockchain network. This mechanism allows efficient monitoring and makes data owners gain complete assurance and control of the data.

The reported data are processed, indexed, and broadcast unto the blockchain network. There is a file storage unit that also stores every transaction detail. Sets of actions are also initialized, which are applicable to any data type to be retrieved from the blockchain network. These actions trigger the smart contracts when they are invoked, to send reports on the rules pertaining to a particular data set. Scripts convey the actions that are initialized by a smart contract.

2.3 Business logic

The blockchain has fast gained an enviable reputation as a highly malleable technology that can be adapted to solve several problems, chief among them trust in collaboration among mutually suspicious parties. Considering the sharing economy and increased reliance on data and data-driven services that enhance or facilitate global business and enterprise, the requirements of data security, efficiency of operations, and transparency of transactions now transcend simply detecting attacks and reprimanding offenders. Collaborative research, which turns data into a product of value, often requires transfers of data, which come with man-in-the-middle attacks, errors, incomplete data, or other challenges. Researchers, depending on the sensitivity of the data at hand, encrypt it before transmission, introducing overhead processing costs. Regardless of the protection mechanisms employed, data security management is still challenging because solving one problem creates another. Again, reliance on data-driven service or processes places severe constraints on data integrity as trusted results cannot be derived from faulty, defective or fragmented data.

In many instances, the blockchain is being used as the supporting infrastructure for secure, scalable, and attack-resilient services. However, collaborators still require transmission of data to participating systems or at least giving out of duplicates. In the end, once out, there is little to no control over the data life cycle and hence monitoring of compliance with acceptable use is problematic. Among others, this means costly duplication of data occurs, data lacks finality and without consensus and immutability offered by blockchain protocols, parties in a system will disagree on the supposed state of the data. Any experimental result from the said data can be contested precisely because of the lack of trust and finality that underlies its creation. This is where transaction-oriented technologies like the blockchain step in to provide a means of

controlling access and monitoring data and a mechanism to incentivize and control behavior of participants.

3 Related work

In this section, research trends pertaining to data access control mechanisms via cloud service providers with emphasis on the improving blockchain technology are outlined.

Zyskind et al. in their work [21] addressed privacy preservation pertaining to third-party mobile services. For system setup on new members, a set of user data attributes are required to be granted for the effective functioning of the system. Users are allowed to change permission in accordance with access control policies stored on a blockchain network. The proposed system is comprised of three main entities, namely mobile phone users, service providers (applications), and the nodes maintaining the blockchain network. T_{access} transactions in their work store user identities associated with system permissions generated on being a verified member in the system. Data aggregated from users are encrypted and stored off-chain with hashes of the data being stored in the blockchain network. They again specified T_{data} transactional structure, which seeks to query stored data based on access rules by the service and system users. Their approach to resolving privacy problems is, however, hampered by allowing data users to request for stored data for processing. Personal and sensitive data in general should not be entrusted to secondary parties. Such secondary parties either aggregate these data to de-anonymize the data acquired or may loose the data to a malicious entity due to mismanagement. In resolving this problem, we propose a mechanism where data reside in its original repository and only results of computed data are made available to the requested parties.

Yue et al. [22] in their research briefly tackled access control management within the healthcare data sharing systems. The authors proposed a purpose-centric access control model which specifies two kinds of data users, namely r-user and p-user. R-users define a user set seeking to read raw data from the system while p-users represent user sets attempting to read data and retrieve results. Transactional data are created for each data request and are made up of a requester with the need to access data of a specific category within a limited period of time depending on a stated purpose. Xiao Yue et al.'s solution achieves data security by the use of access control mechanisms but does not take into consideration the originality of the data and thus does not achieve data provenance. Data processing on their system is not secure, as anonymized data sets of the different data output from nodes available to authorized users of such data can be de-anonymized. Again, actions based on policies to manage the behavior of these users are not enforced by any mechanism.

Zyskind et al. [23] again used the blockchain for access control management and audit log security purposes to serve as a mechanism for tamper proof logging of events. The system proposed was Enigma, which is a decentralized computation platform based on an optimized version of the secure multi-party computation (sMPC). The system users store and run computations on data while keeping the data completely private. However, data get degraded after periods of transfers and modifications and result in dirty read. The original data are made available to permissioned users for

processing and computations. Permissioned user behaviors are again not enforced by any mechanisms to prohibit them from acting maliciously on data.

Blockchain-based access control management was described in more detail in Thomas and Alex's [24] proposed system called ChainAnchor to provide anonymous but verifiable identities to users trying to perform transactions on a permissioned blockchain. The Enhanced Privacy ID (EPID) zero-knowledge proof scheme is used to achieve and prove a participants' anonymity and membership in the system and permits entities to register unique public keys for transacting on the blockchain. ChainAnchor does not provide a processing model for accessed data. Therefore, there is no process management as well as data provenance and auditing. Data handled by users must be well traced and enforced by mechanisms that bind actions of the data users. This mechanism is absent from Thomas and Alex's system.

Sundareswaran et al. [25] proposed ensuring distributed accountability for data sharing in the cloud. Their design exhibited an approach for automatically providing access to data in the cloud while enhancing secure auditing on the trails of the data. Data owners are allowed to attach mechanisms to yield secured back-end protection to data when requested. The problem with this mechanism, however, is similar to previously reviewed works. System user actions are not bound to specific roles, therefore allowing malicious entities to perform computations on different received data sets to retrieve the original data. Mechanisms to adequately monitor user actions as well as process data on behalf of system users should be implemented to achieve accountability for data sharing.

Ferdous et al. [26] presented Decentralised Runtime Monitoring for Access Control Systems in Cloud Federations, also known as DRAMS, a blockchain-based decentralized monitoring infrastructure for a distributed access control system. The key motivation of the system is to deploy a decentralized architecture that detects policy violations in a distributed access control system under the assumption of a well-defined threat model. The output of the mechanism is de-obfuscated by a privacy manager to achieve access control. However, the privacy manager provides only limited features in that it does not guarantee protection once the data are being disclosed.

Hassan et al. [27] presented a hybrid network model for efficient real-time wireless body area network (WBAN) media data transmission. Their design architecture combines WBAN and Cloud for valid data sharing and delivery. The system utilizes the combination of content-centric networking (CCN) or named data networking (NDN) with adaptive streaming technique to support uninterrupted media healthcare content delivery to multiple patients and physicians, as well as help to reduce packet loss. The focus on this work was to provide an efficient mechanism for sharing of medical data among transacting parties. Their work is, however, obscured by mechanisms to validate the originality of data received in the system. As such, data provenance and auditing is not considered in their system.

In this work, we provide secured measures to retrieve data and processing results based on a sovereign blockchain system. The main contribution of our work is to provide processing, provenance, auditing, and trailing in big data environments. The various literatures reviewed in this section provide insufficient mechanisms in achieving the above-mentioned metrics. We further employ smart contracts to effectively monitor the behavior of data being used by validated system users as well as provide

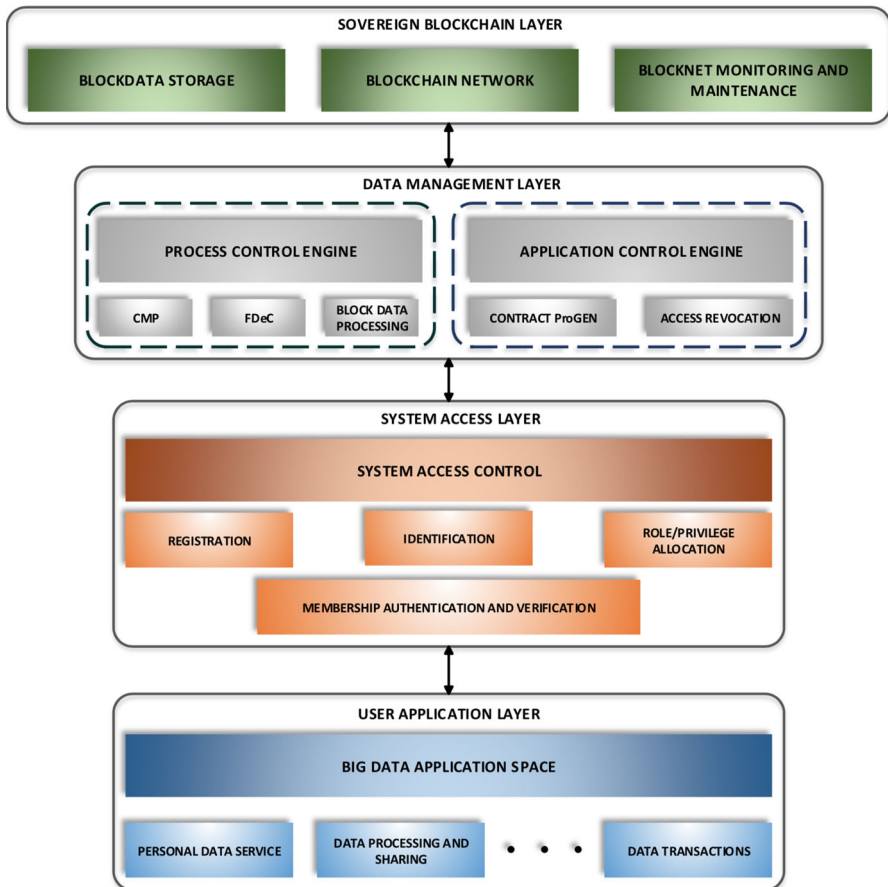


Fig. 1 System architecture layers and components

a suitable mechanism to enable the processing needs of users in the system while preserving the originality and security of the data.

4 System design

4.1 System architecture

The architectural design of the system can be categorized into four layers as displayed in Fig. 1. These sections are composed of carefully designed components and interfaces that aid in achieving transparency in the chain-based big data access control infrastructure.

1. *User application layer* The Access Control System for Big Data Applications encompasses a vast area of applications on implementation. The system framework serves as a foundation on which key system infrastructures can be built on. Users

in the system map to a user logic model, which defines the specific roles of entities directly involved with system processes. These sets of users are categorized into different user groups so as to allocate unique but generic system identification parameters mapped to user roles to achieve a degree of security for the big data system. For a given area of implementation such as medical data, system users would be the patient, the doctor, the pharmacist, and the financial department.

These entities in the medical-based application have certain privileges and roles applicable on data as the data traverse from its origin (patient) through the doctor and pharmacist till the data repository. It is only deemed right that such users should have specific identities that identify them as being part of a larger category-based user group with unique identities (parameters). As such, we make provision for designing unique identification for the access control-based system for big data environments. This process limits the intention of malicious activities by individual players in the system.

2. *System access layer* The system access layer is composed of entities that register, authenticate membership, provide identification and an entity allocating specific roles and privileges to verified system users. The importance of this layer is to maintain accurate identification and authentication of individuals requesting to be a valid member of the systems. The entities of the system access layer are:
 - *Membership registration* The responsibility of this entity is to accept user requests to participate in transaction in the big data environment. Cryptographic keys are distributed to members, which need to be validated to become an established system member. The registration entity stores request logs in a local database, which is shared to processing nodes to generate blocks based on the logs.
 - *Identification* The role of this entity is to allocate strings of alphanumeric characters to registered members in the system. These are mapped to their identities and log-in details and specify their roles and privileges.
 - *Role and privilege allocation* The role and privilege allocation entity is responsible for specifying roles of registered system users. These are specified as part of the registration process as unique key identifiers are associated with validated users. This mechanism enforces policies established in the system. Modified policies infer that certain users would have access rights to requested data since their privilege set would be modified.
 - *Authenticator* The authenticator is responsible for verifying the legitimacy of a user's membership to the system. This is achieved by presenting sets of cryptographic parameters assigned to a user during the registration phase. This mechanism ensures that only appropriate individuals of the system are allowed to be members and allowed to transact in the big data system. Participants with invalid key attributes are logged out of the system to prevent malicious activities.
3. *Data management layer* The data management layer consists of entities that enable a secured and efficient transfer of datablock information in the system. This layer comprises of the processing control engine, which implements a management protocol on receiving and delivering analytical result derived from data block. A component of the processing control engine enforces that relevant data traces are

deleted and system users are prevented from accessing raw files being processed. This component again ensures that BlockData duplicates are retrieved securely from storage and malicious activities on reduplication are avoided. Smart contracts, as stated throughout this paper, are generated by an application control engine, which resides in the data management layer.

4. *Sovereign blockchain layer* The sovereign blockchain layer consists of the block-data storage, blockchain network, and the blocknet monitoring and maintenance. The function of this spans from the accumulation of logs and indexes to storage of a contiguous aggregation of data processes in a immutable ledger. The blockdata storage ensures that actual data reside at points of its current trace in which it exists. The blocknet monitoring and maintenance aid in ensuring the appropriate functioning of components in the sovereign blockchain layer.

4.2 Design formulation

This section discusses the mechanism used by the architecture for data analysis. It comprises of the setup of the system for effective and efficient transaction, the data management layer, which is the heart of the system, and the file retrieval and storage platform.

- *System setup* A registered user sends a request to access data from the system. The data request is signed by using a set of pre-generated private keys from the user. The request is received by an entry point in the processing node in the data management and processing layer, which records and processes requests. The authentication unit receives the request and verifies the legitimacy of the users by double-checking the signature using their public keys. If the signature is a valid one, the process is accepted and access is granted, else invalid requests are dropped. A valid request is then forwarded to the blockchain network for data retrieval. A duplicate of the data is provided in a virtual container for the users to work on. After the necessary computations on the data, and to the satisfaction of the participating parties, a copy of the result is stored on the blockchain network for reference, and the duplicated data are destroyed.
- *Data management process* This unit comprises of two compartments: the process control engine and the application control engine. In the process control engine, there is a consensus monitoring platform which enables an agreement to be reached between transacting parties. The agreement may vary from data usage to participant requirements for a transaction. There is a fault control system, which is embedded in this layer, that binds the participants and checks for correctness or fairness of a transaction. When all these processes have been verified to be accurate, the processing node then acquires the data to be operated on from the blockchain network.

Moreover, there is a contract engine that specifies the various conditions under which accessed data should be manipulated. When any of the participants go contrary to such policies binding the transaction, access to the data is revoked. Some penalties are also paid by perpetrators.

- *Data retrieval and storage* When a request is made by a user to access data, the processing node first inspects the ID and other parameters of the user. The parameters which are provided by the user have to undergo authentication in the authentication unit. After a successful authentication is done, the user now has access to the data it requires. The processing node fetches the requested data from the blockchain network, but provides a duplicate of the data. The user then works on the copy provided, in a virtual container, after which results are reported to the blockchain network. The virtual container is destroyed after an operation on a data set. The storage handles the efficient storage as well as organization of the data. The core of this storage is the deployment of storage structures in the blockchain network that adapt to various types of data and the frequency of data capture.

4.3 Registration and authentication

Users are registered in the system through system registration request, which involves the aggregation of information pertaining to a specific user. This will account for the category of a user set to which the registered user will belong. On achieving successful grouping, unique string of computed characters is allocated to the user signifying membership into the access controlled system. From the information of identities, the system can determine the specific category of users and allocate specific roles and privileges to system users. These procedures define each user and tie each entity to a region of operation. Each user is then given a pair of keys, generated by the system and associated with user identities and multiplicatively shared between the server and each user.

In order to access data from the big data system, an authentication procedure needs to be followed. A request is sent to the server by a user for access to data from the system. The request contains the information on the data needed by the requestor, which contains a digital signature reflecting the identity of the requestor. A Transport Layer Security (TLS) connection is created from a public key, which conforms to a verified server certificate for the connection. Malicious attacks such as man-in-the-middle attack is thwarted due to verification and matching of public keys from the connection between the client and the server. On successful computations, a user gets part of a computed private key from the system with the system keeping part of the verifiable parameters of the distributed private key. This initiates the start of a two-party signature protocol over the connection.

A zero-knowledge proof based on cryptographic hashes by the use of an Elliptic Curve Digital Signature Algorithm (ECDSA signature) is computed by the client (user) and server (system) after which a comparison is made to check for authenticity and feasibility of the solution. With successful computation based on the fact that computational results of the client and server are the same, the server delivers the other half of the key to the client before secure log-in and authentication are provided. Users are now allowed access to the requested data based on their respective allocated roles and privileges. On malicious activities, a client is revoked from accessing the system and their identity is blacklisted and stored in a secure local database, which is referenced and mapped on block data structures in the blockchain network. The key

generation and authentication of our system is based on a two-factor authentication protocol provided by Mann and Loebenberger [28] and is presented in the “Appendix” of this paper.

4.4 Data management engine (DME)

This section discusses the various design considerations to enable the secured extraction and processing of data in the blockchain-based access control system. Structures defining the implementation of DME are: Consensus Management Process (CMP), Fault Detection and Correction (FDeC) processing model, and a block data sharing platform.

4.4.1 Consensus management process (CMP)

The Consensus Management Process directly deals with the verification of requests made by system users for information retrieval or processing. On receipt of the request, the system access control components validate the authenticity of the user by comparing the hash of the user’s identity to already-existing hashes of registered participants. Successfully verified user identities advance to request processing by the CMP since the CPM is directly interfaced with the system access control layer. Invalid user requests are ignored, and users are blacklisted from using the system. Blacklisting of users in the system is based on invalid verification and malicious operations by valid users of the system. For validated requests, CMP processes these in two stages, thus, request validation and role mapping and request formatting.

For request validation and role mapping, the CMP extracts the details (purpose) of the request filed (a standardized format defined from system setup). The roles and privileges mapped to user identities stored in the local database are examined in relation to policies derived from the setup on the system. For valid roles and privileges on requests, the CMP activates an action on request formatting; otherwise, user requests are ignored and an error message is sent to the user. On request formatting, CMP reprocesses data request into a desired format to be sent to the block data sharing platform. The final role of the CMP is to enforce an FDeC scheme on a computational workspace generated by the block data sharing platform

4.4.2 FDeC processing model

FDeC is a processing model that ensures a secure supervision and management of data among entities in the system. It exhibits properties similar to the Byzantine Fault Tolerance (BFT) algorithm by inferring on where, when, and how block data extracted from storage due to requests are being managed by the block data sharing platform. For computations based on retrieved data sets, FDeC enforces strict rules to achieve an environment formed on secure supervision and management on block data in the system. On instances of data processing, FDeC verifies the authenticity of BlockData duplicates to ensure an efficient and transparent working environment. Finally, FDeC ensures total destruction of duplicated BlockData after successful processing.

4.4.3 Block data sharing

The data sharing platform is directly interfaced with the active components of the system such as smart contract generator, BlockData store, and the sovereign blockchain network. The main roles of the block data sharing are to retrieve and initialize data-based requests made on BlockData platform. On receiving requests for the access of data, the block data sharing platform accesses BlockData in current data stores via indexing mechanisms established in the system.

Data privacy protection schemes such as data masking, anonymity, homomorphic encryption, and other privacy protection mechanisms are applied on the duplicated BlockData. Requests are made to the Application Contract Engine to generate scripts based on system data policies to regulate the use of information in BlockData by system users. These smart contract scripts are attached to a data format computed by the block data sharing platform from the BlockData duplicate and finally made available to the requester.

For data processing, a virtual container is created to run computations on the BlockData duplicates. It should be noted that data privacy schemes are always applied on duplicates of BlockData. Results acquired from the computations are transformed into appropriate format, and smart contracts are generated and attached to the final output, which is then made available to the requester.

4.5 Computation workspace

The computation workspace is a data analysis system enabled through virtualization for big data processing and analytics based on user requests on particular data sets. Reports generated from computations are prepared in appropriate formats and distributed to respective requesters. The computations can be expressed as a data container that takes as input BlockData duplicates which have been preprocessed regarding the privacy on the contents on each BlockData. BlockData inputs fed into the data container are locked and can never be unlocked, on entry for data processing. This feature is to ensure that input data are destroyed when a collapsed-state action is called. The outcome of the computation would be results and reports, which would be extracted from the system by relevant entities.

An action of collapsed state (disintegrate/destroy) is invoked in the virtual data container on completion. At this point, the FDeC confirms the destruction of input data and parameters necessary to invoke the creation of virtual container. The virtual computational workspace design is categorized into three sections but exists in two main layers. The layers are: Container Application Support which encompasses application support engine, the Processing Engine which includes the container Operating System (OS) manager, and the processor resource allocation. Figure 2 highlights the architecture of the virtual computation workspace.

4.5.1 Container application support

The container application support is made up of entities that specify processing requirements and support engines. Entities in this layer are:

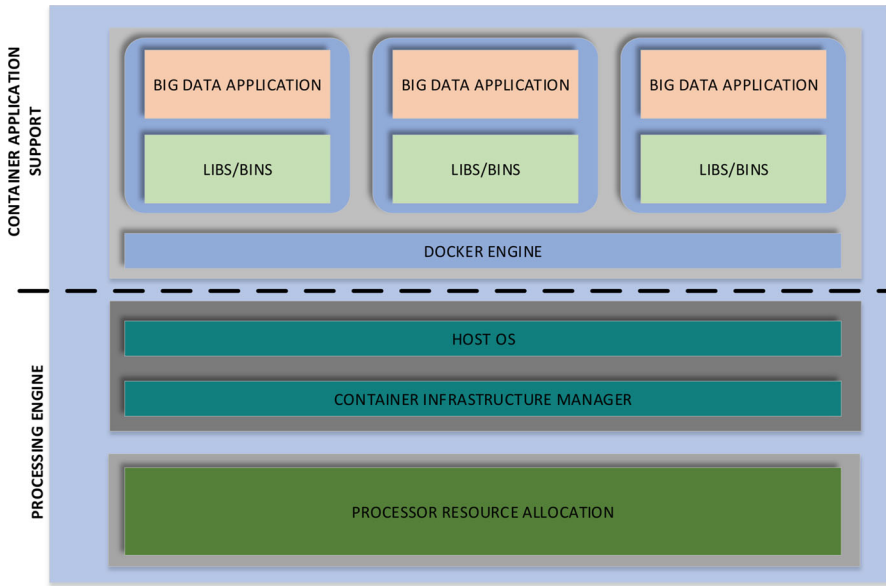


Fig. 2 Architecture of virtual computational workspace

- *Application-oriented specialization* Specifies application processing requirements based on data analysis and reports. The steps involved in this process include acquisition and locking of transactional data (BlockData duplicate) and data cleansing to extract useful fields needed for particular computations based on requests. The output is a clean data, which are aggregated and reported (logs for optimization). Further stages include advanced processing drawn from data extraction and sampling on clean data, data modeling, partitioning, and validation to deduce adequate results for users. These processes are modified based on desired output from each requester on the setup of the system based on big data analytics.
- *Libs/bins* These specify where the library and binary files are stored. These are stored as compressed directories and are classes and tools supplied to support container applications.
- *Docker engine* Docker Engine is a client–server-based technology that sets up and executes containers using services and components provided by Docker. The engine supports tasks associated with building, transporting, and running container-based applications. A server-side demon hosting images, networks, and containers are created to operate alongside a client-side command-line interface enabling users (block data sharing platform) to interact with the demon through an Application Programming Interface (API). Conditions on desired states are specified to exhibit the declarative nature of the engine and ensure actual states and desired state match on all instances of processing.

4.5.2 Processing engine

Features that support the entire application support specifications on processing Block-Data. Entities in this layer include:

- *Host OS* OS containers are virtual environments that offer the part of the kernel of the host OS, however, ensuring client space segregation. OS containers are considered as virtual machines where applications and libraries are specified and configured to achieve processing requirements on client requests. Processes running in the container can be assigned resources specified by the initiation of the application.
- *Container infrastructure manager* The container infrastructure manager is a key component of the virtual computation workspace architecture that is responsible for controlling and managing processing, storage, and network resources. The ability to allocate and create a well-optimized system is managed by the container infrastructure manager to obtain relevant data. Load balancing on resources is managed and controlled by the infrastructure manager.
- *Processor resource allocation* The processor resource allocation is directly interfaced with the container infrastructure manager and receives information on the amount of resources to allocate to different processes based on a number of executions per second. For idle state of available processors, heartbeat messages are sent to the container infrastructure manager to effectively maintain a consistent log on process executions to create room for well-optimized system processing on BlockData.

4.6 Application contract engine (ApCE)

The application control engine is made up of entities that generate smart contracts on BlockData request, revokes access to malicious system users, and blacklists users from creating actions in the system. These structures are combined to ensure a secure supervision on the entire system processes. The ApCE consists of the Contract ProGen and an Access Revocation center.

- *Contract ProGen and access revocation* Contract ProGen is the structure responsible for generating smart contract scripts based on script requests received from the block data sharing platform. Scripts generated can be in two forms developed from specification on processed results. As mentioned earlier, user request can be either to access information based on a BlockData structure or analysis based on specified user analysis requirements. There is the need to generate specialized scripts to oversee user actions on both forms of results so as to adequately trace the behavior of the resultant output and thus prevent malicious usage. The Contract ProGen issues receipts based on completion on processes as a form of time-stamped agreements by system and users to ensure transparency and system process consistency. Receipts are digitally signed by system users and block data sharing platform and saved in a local database with a copy sent as an attachment to the data sent to the requester. A local database accumulating process logs and

receipts ensures a synchronized system state on traces of data which are in the blockchain network.

The access revocation entity is a permissioned database designed for invalidating user actions in the system. It is an action and violation storage infrastructure, which is triggered on the reception of violation reports based on smart contract scripts attached to user-requested data. Violations on data are based on access policies established in the system. Again receipts are kept for each activity enacted to sufficiently give consistency for data auditing for future reference. Algorithm 1 describes sample scripting processes attached to a data package (result on processed data or BlockData info), which is sent to system users.

4.7 Block data infrastructure

The block data infrastructure characterizes the operations of the BlockData storage on the blockchain network and the blocknet monitoring and maintenance residing in the sovereign blockchain layer. This infrastructure identifies the local store on BlockData as the data generated in big data processes reside at the point where the data exist. Data are structured into the BlockData format to allow a careful retrieval of information without compromising the security and privacy regulations on aggregated data. Duplicates on block data are made on instances of data processing or information retrieval by a system user.

Algorithm 1 Smart Contract: BlockData Info

```

Initialise: User.ID, User.Sig, ProcessNode.ID, ProcessNode.Sig, Action, Data.Status,
DataType, Data.ID
for DataType == BlockInfo and Status == Active do
  Retrieve Policy  $\Rightarrow$  Data.Action
  Retrieve Seq  $\leftarrow$  Data.ID
  if Data.Action not in Policy then
    Data.Status = Violation
    Status  $\Rightarrow$  Inactive
    Function.Call  $\Rightarrow$  (Revoke  $\leftarrow$  User.ID)
    Call.Destroy  $\leftarrow$  Accessed data mapped to User account
    Report Log
  else {Data.Action in Policy}
    continue
  end if
end for
for DataType == BlockProcess and Status == Active do
  Retrieve Policy  $\Rightarrow$  TimeStamp.Usage
  Retrieve Seq  $\leftarrow$  Data.ID
  if TimeStamp.Usage > Policy then
    Status  $\Rightarrow$  Inactive
    Call.Destroy  $\leftarrow$  Accessed data mapped to User account
    Report Log
  else {TimeStamp.Usage in Policy}
    continue
  end if
end for

```

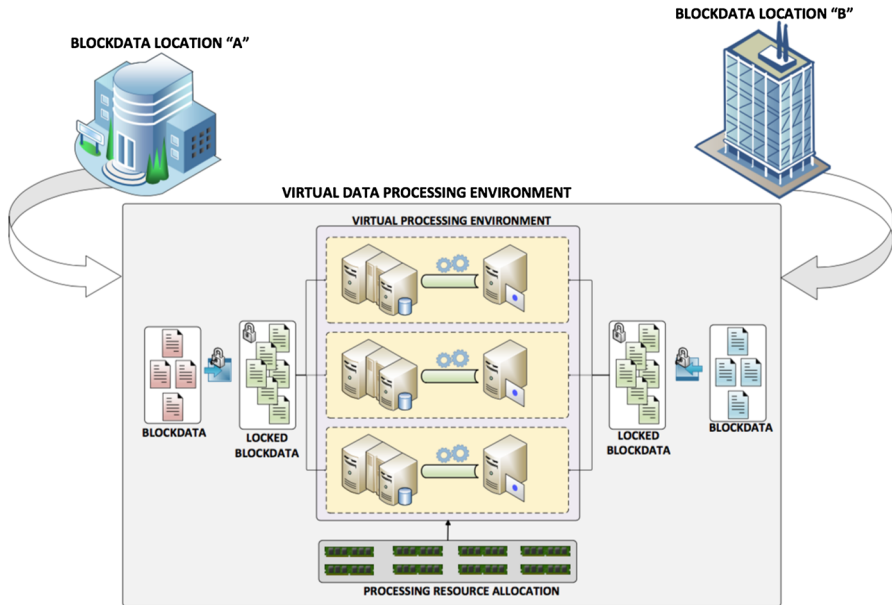


Fig. 3 BlockData processing, monitoring, and maintenance

The blockchain network maintains logs on all actions and behavior of data processes in the system by system entities and users. These logs are cryptographically secure and immutable to enhance the data forensics on malicious occurrences. These systems scale by the data structures implemented in the system design. All functions on data processing, block/database synchronization, and indexing to retrieve BlockData from their respective storage for processing are maintained by the blocknet monitoring and maintenance. These systems ensure an efficient sharing of data in the access control system for big data environments. Figure 3 exhibits the retrieval of BlockData from storage to processing.

5 Performance evaluation and analysis

This section analyzes the performance of implementing our system against already-proposed-and-in-implementation works, relating to data access control in several data management sectors. For the blockchain-enabled big data access control system, we develop a proof of concept (PoC) implementation to test and evaluate the system. The chosen blockchain technology for the PoC is based on a private Ethereum network. Under this, blocks will be specified and utilize the types of accounts specified in Ethereum. Ethereum is a programmable blockchain-based platform that utilizes the robustness of solidity, which is a state-based scripting language. Solidity can be used to build, deploy, and implement smart contracts with no restrictions on data size stored on the blockchain network.

The management and monitoring of user actions on BlockData requests to ensure a completely secure supervision of system processes are implemented as smart con-

tracts in solidity. All users in the system are enrolled as externally owned accounts, which are controlled by private keys generated in Ethereum. Smart contracts in this system are deployed under contract accounts controlled by contract code. When a contract account is activated by request, the code is activated. Enforcing contracts are completed by use of Truffle, which is used for contract management, deployment, and migrations with numerous integrated testing frameworks backed by truffle boxes. We use these as a foundation to set up environments for testing due to its pre-configured web-development environments. Since the system is posed with the responsibility of ensuring visibility and the control of user actions in addition to data and contract processing, there is no need for mining. As mentioned earlier, Algorithm 1 describes the formulation of a smart contract.

We further enforce functions in the system which are Call and Invoke operations to support all forms of queries and actions in the blockchain-based big data access control system. We implement JavaScript interfaces to connect the process control engine and application control engine with the blockchain network. The interface connecting these entities to components in the sovereign blockchain layer is a web3.js library enabled through RPC calls. A CoAP-based JavaScript library is used to interact between the blockchain network and the blocknet monitoring and maintenance entity. These are made feasible through Remix, which is an Ethereum IDE in the cloud with a debugger and a simulated blockchain environment. In simulating interactions on CMP and the FDeC processing model in relation to computations, the virtual processing engine serving as the computational workspace is completed by using JMeter to simulate the number of requests processed by the process control engine.

We evaluate the integration of major entity configurations and of the DTLS library including all component connections to the processing system. The testing of the parameters of the system was completed on an Ubuntu 16.04.1 operating system using Docker and vertigo/ethereum image derived from client-go image of the Ethereum protocol ethereum/client-go. These refer to the validation of requests by systems users, the visibility and management of BlockData retrieved from data repositories, and data processing with analysis by the processing engine. We initiate the number of active requestors on the blockchain system for requests between 10 and 1500 users for periods of 2, 5, 8, 10, 20, and 30 min, which are achieved using JMeter. Vulnerabilities in the blockchain-based system are detected via FDeC processing model for violations on accessed false BlockData, which serves as threats to the system. Vulnerability analysis is achieved if the FDeC can detect any inconsistencies and violations under substantial amounts of data retrieved from data repositories based on user requests. In summary, the algorithms implemented by the components in processing control engine should be able to detect all attacks posed on the system. Benchmarking is achieved by use of Californium, obtained from CoAPBench, which is a benchmark tool. CoAP uses virtual clients to meet established concurrency factors.

The major metric being compared here is latency, which is defined as the total delay experienced when a transaction is being made. Latency in our system has been evaluated by analyzing the time taken to process requests from the blockchain system. Given variations to number of active requests in the system, we analyze the time it takes for each system to execute the total number of requests. The number of requests in the system is therefore represented as *active processes*. In our system, we could

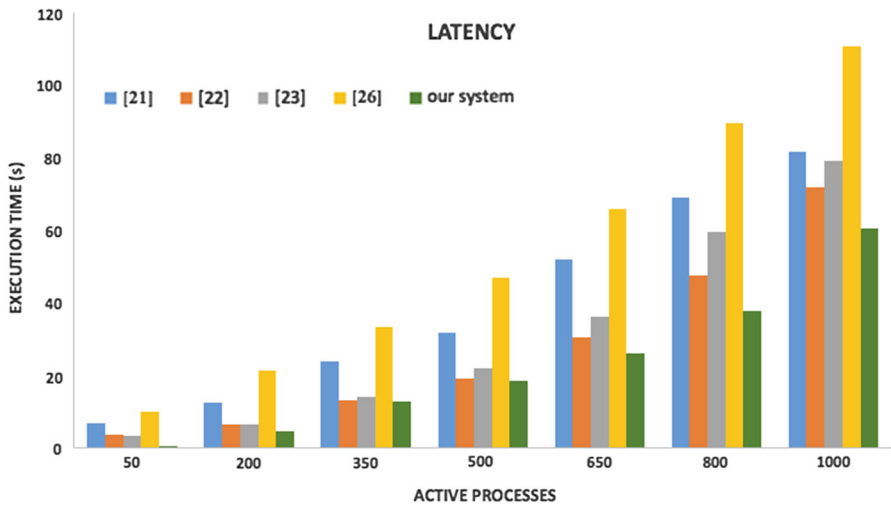


Fig. 4 Comparison on latency of existing systems to our proposed system

Table 1 Comparison between proposed system and other related systems

Metric	[21]	[22]	[23]	[24]	[25]	[26]	Our System
Blockchain-based	Y	Y	Y	Y	N	Y	Y
Data security	N	Y	Y	Y	N	Y	Y
Bandwidth efficiency	N	N	N	N	N	N	Y
Access control	Y	Y	N	Y	Y	Y	Y
Process management	N	N	N	N	Y	N	Y
Provenance and auditing	Y	N	Y	N	N	Y	Y
Transaction transparency	Y	Y	Y	N	Y	Y	Y

see that the time it takes for a transaction to be processed is much less than other already-existing algorithms. With an increasing number of data requests, the time still manages to beat other schemes. This proves that our system offers an effective and efficient transaction process time. Figure 4 presents a comparison on latency results achieved on active process executions in the various systems.

Other metrics that we based our comparison on include data security (defines the difficulty at which BlockData are accessed from the system before or after processing), bandwidth efficiency (defines the information delivery rate based on active users), access control (defines the ability of the systems to identify the categories of users and their actions/privileges to information in the system), process management and monitoring (defines how well the system can coordinate processes as well as provide visibility to actions), data provenance and auditing (defines the ability of the system to maintain a consistent and accurate data trails on aggregated data logs and files), and transaction transparency (defines the ability to detect and correct BlockData inconsistencies based on user requests). The table below shows the comparison between our system and the already-existing ones. Table 1 compares our data access and processing system to other existing systems and literature presented in this paper.

6 Conclusion and future work

In this paper, an off-chain sovereign blockchain is introduced where the participants to be involved in a transaction do so in a virtual container, which is created and monitored by the system itself. The results of a transaction are then transferred to and stored onto the blockchain network. This solves the problem of keeping excess data on the blockchain network. The security and effective access control of this scheme are enhanced. System performance is also increased, with comparison made with other existing schemes.

Acknowledgements This work is supported in part by the applied basic research programs of Sichuan Province (2015JY0043), the Fundamental Research Funds for the Central Universities (ZYGX2015J154, ZYGX2016J152, ZYGX2016J170), programs of international science and technology cooperation and exchange of Sichuan Province (2017HH0028), Key research and development projects of high and new technology development and industrialization of Sichuan Province (2017GZ0007). This work is supported by the National Key Research and Development Program of China (Grant No. 2016QY04WW0802, 2016QY04W0800, 03). This work supported by the National Engineering Laboratory for Big data application on improving government governance capabilities.

Appendix: Two-factor authentication scheme

In this paper, an ECDSA authentication scheme, which was proposed by Christopher Mann and Daniel Loebenberg [28], is adopted. The scheme has three phases: initialization, construction of an ephemeral key, and a signature formulation.

- *Initialization* An ECDSA key pair (d, Q) is generated. The private key is multiplicatively shared between the user and the system, by selecting $d_U \in \mathbb{Z}_n^*$ pseudorandomly and computing $d_S = d \times d_U^{-1}$ in \mathbb{Z}_n^* . Then, $d = d_U d_S$ and the user gets its share of the key d_U , while the system also takes d_S . Both user and system then compute their corresponding public keys $Q_U = d_U G$ and $Q_S = d_S G$, where G is a finite base point on an elliptic curve, E . Two key pairs, (sk_U, pk_U) and (sk_S, pk_S) , for a homomorphic public key encryption scheme are generated and distributed to the user and system accordingly.
- *Key construction* In this phase, a shared ephemeral secret $k = k_U k_S \in \mathbb{Z}_n^*$ is generated together with the corresponding public key $V = kG \in E$. The user and the system also compute the public keys corresponding to their shares of this secret as $V_U = k_U G$ and $V_S = k_S G \in E$. Also, the user commits to the two values k_U^{-1} and $k_U^{-1} d_U$ in \mathbb{Z}_n^* by sending the corresponding encryptions under pk_U to the system.
- *Signature formulation* In the final phase, the system uses the two commitments together with the homomorphic property of the encryption scheme to finally compute the second part of the ECDSA signature.

References

1. Chen J, Xue Y (2017) Bootstrapping a blockchain based ecosystem for big data exchange. In: Proceedings—2017 IEEE 6th International Congress on Big Data, BigData Congress 2017, pp 460–463

2. Liu PTS (2016) Medical record system using blockchain, big data and tokenization. In: Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), vol 9977 LNCS, pp 254–261
3. Es-Samaali H, Outchakouch A, Leroy JP (2017) A blockchain-based access control for big data. *Int J Comput Netw Commun Secur* 5(7):137147
4. Zheng Z, Xie S, Dai H, Chen X, Wang H (2017) An overview of blockchain technology: architecture, consensus, and future trends. In: Proceedings—2017 IEEE 6th International Congress on Big Data, BigData Congress 2017, pp 557–564
5. Morabito V (2017) The security of blockchain systems, business innovation through blockchain: the business perspective, pp 6178
6. Swan M (2015) Blockchain thinking: the brain as a decentralized autonomous corporation [commentary]. *IEEE Technol Soc Mag* 34(4):4152
7. Nath I (2017) Data exchange platform to fight insurance fraud on blockchain. In: IEEE International Conference on Data Mining Workshops, ICDMW, pp 821–825
8. Tapscott D, Tapscott A (2016) Blockchain revolution: how the technology behind bitcoin is changing money, business, and the world. Penguin, pp 361–367
9. Qiu J, Wu Q, Ding G, Xu Y (2016) A survey of machine learning for big data processing. *EURASIP J Adv Signal Process* 1:67
10. Ranjan R (2014) Streaming big data processing in datacenter clouds. *IEEE Cloud Comput* 1(1):7883
11. Wu X, Zhu X, Wu G-Q, Ding W (2014) Data mining with big data. *Knowl Data Eng IEEE Trans* 26(1):97107
12. Dessureault S (2016) Understanding big data. *CIM Magazine* 11.1
13. Tschorsch F, Scheuermann B (2016) Bitcoin and beyond: a technical survey on decentralized digital currencies. *IEEE Commun Surv Tutor* 18(3):20842123
14. Nadarajah S, Chu J (2017) On the inefficiency of Bitcoin. *Econ Lett* 150:69
15. McGinn D, Birch D, Akroyd D, Molina-Solana M, Guo Y, Knottenbelt WJ (2016) Visualizing dynamic bitcoin transaction patterns. *Big Data* 4(2):109119
16. Wijaya DA (2017) Extending asset management system functionality in bitcoin platform. In: Proceeding—2016 international conference on computer, control, informatics and its applications: recent progress in computer, control, and informatics for data science, IC3INA 2016, pp 97–101
17. Ciaian P, Rajcaniova M, Kancs A (2016) The economics of BitCoin price formation. *Appl Econ* 48(19):17991815
18. Bhme R, Christin N, Edelman B, Moore T (2015) Bitcoin design principles enabling technologies and processes. *J Econ Perspect* 29(2):213238
19. Dennis R, Owenson G (2016) Rep on the roll: a peer to peer reputation system based on a rolling blockchain. *Int J Digit Soc* 7(1):11231134
20. Wright A, De Filippi P (2015) Decentralized blockchain technology and the rise of lex cryptographia. <http://ssrn.com/abstract=2580664>. Accessed 15 Nov 2017
21. Zyskind G, Nathan O, Pentland AS (2015) Decentralizing privacy: using blockchain to protect personal data. In: Proceedings—2015 IEEE Security and Privacy Workshops, SPW 2015, pp 180–184
22. Yue X, Wang H, Jin D, Li M, Jiang W (2016) Healthcare data gateways: found healthcare intelligence on blockchain with novel privacy risk control. *J Med Syst* 40(10):218
23. Zyskind G, Nathan O, Pentland A (2015) Enigma: decentralized computation platform with guaranteed privacy. [arXiv:1506.03471](https://arxiv.org/abs/1506.03471) [cs], pp 114
24. Hardjono T, Smith N, Pentland A (2016) Anonymous identities for permissioned blockchains. Available: <http://connection.mit.edu/wp-content/uploads/sites/29/2014/12/Anonymous-Identities-for-Permissioned-Blockchains2.pdf>. Accessed 22 Aug 2017
25. Sundareswaran S, Squicciarini AC, Lin D (2012) For data sharing in the cloud. *IEEE Trans Dependable Secure Comput* 9(4):556568
26. Ferdous S, Margheri A, Federica P, Vladimiro S (2017) Decentralised runtime monitoring for access control systems in cloud federations. GB University of Southampton, Southampton, p 11
27. Hassan MM, Lin K, Yue X, Wan J (2017) A multimedia healthcare data sharing approach through cloud-based body area network. *Future Gener Comput Syst* 66:4858
28. Mann C, Loebenberger D (2017) Two-factor authentication for the Bitcoin protocol. *Int J Inf Secur* 16(2):213226