International Journal of Computer Networks and Communications Security

VOL. 5, NO. 7, JULY 2017, 137–147
Available online at: www.ijcncs.org
E-ISSN 2308-9830 (Online) / ISSN 2410-0595 (Print)

A Blockchain-based Access Control for Big Data

Hamza ES-SAMAALI[1], Aissam OUTCHAKOUCHT[2] and Jean Philippe LEROY[3]

[1, 2] Ph.D Student, Laboratory LISER, IPI, Paris, France

[3] Professor, Laboratory LISER, IPI, Paris, France

[1]hamza.essamaali@gmail.com, [2]aissam.outchakoucht@gmail.com, [3]jpleroy@groupe-igs.fr

ABSTRACT

Big Data era is upon us, a huge amount of data is generated daily, analyzing and making use of this huge amount of information is a top priority for all kinds of businesses. However, one of the most important problems that hinders the unanimous adoption of Big Data is the lack of security and privacy protection of information in the Big Data tools. In this paper we contribute to reinforcing the security of Big Data platforms by proposing a blockchain-based access control framework. We define the concept of blockchain and breakdown the mechanism and principles of the access control framework.

Keywords: *Big Data, Access Control, BlockChain, Information Security, Privacy, Authorization*.

1 INTRODUCTION

Big data designate a voluminous set of data that no conventional database management or information management tool can really analyze. In fact, we generate about 2.5 quintillion bytes of data every day [1]. These are the informations coming from everywhere: messages that we send, videos we publish, climate information, GPS signals, transactional online shopping records and many more. These data are called Big Data.

The arrival of the Big Data is now presented by many articles as a new industrial revolution similar to the discovery of steam (early 19th century), electricity (late 19th century) and computer science 20th century). Others, a little more measured, call this phenomenon the last stage of the third industrial revolution, which is actually "information." In all cases, Big Data is seen as a source of profound upheaval in society.

With Big Data comes big responsibilities, one of the most discussed issues is the security of information and protection of privacy in the Big Data platforms. Since all the tools used in the industry are relatively new (Hadoop, Hive, Spark ...) and the NoSQL databases are recent too. This leaves us with a high risk of potential attacks and security breaches since none of those tools have robust built-in security policies and mechanisms. This lack of security is the reason behind the relatively slow adoption of Big Data by enterprises to store and analyze their data.

Generally, the systems are not compromised due to the exploitation of the vulnerabilities of communication protocols or cryptographic primitives. Most of the breaches come from the bad configuration of access control and authentication policies.

So Authentication and access control technologies are the main elements to address the security and privacy issues in Big Data. Actually, any effective access control system should satisfy the main security properties of confidentiality (preventing unauthorized divulgation of resources), integrity (preventing resource to be modified without authorization resources), and availability (assuring access to resource by legitimate users when needed). In this paper, we focus only on Authorization, authentication and accountability are out of our scope for now.

Big Data requires a privacy aware access control (PAAC) suitable to the needs and specificities of the technology. Due to the recent explosion of interest around the blockchains, we will study whether they are capable of building a lightweight and robust access control framework in the Big Data sector. The Blockchains enable us to have a distributed peer-to-peer network in which non-trusting members can interact with each other without a trusted intermediary, in a cryptographically verifiable manner.

In section 2 we will talk about the access control already used in the Big Data technologies, in section

3 we will define and explain how blockchain works, then we will present our framework in the 4th part of the paper, the fifth and last part will contain a conclusion and future works.

2 ACCESS CONTROL IN BIG DATA

According to [2] Big Data has basic forms of access control (mainly RBAC) the majority of platforms only supports access control at the application level and none of them have privacy policies.

RBAC (Role-Based Access Control) [3] is an access control model and framework for controlling user access to resources based on roles. This model consists of four different components and each one of them assigns to RBAC a number of functionalities. These components are the core RBAC, the hierarchical RBAC, the static separation of duty relations and the dynamic separation of duty relations. The core RBAC model is composed of five static elements. These elements are the users, roles, and permissions, with the latter being composed of operations applied on objects. The relationship among the elements of the core model is straight forward many to many. Roles are assigned to users and permissions are assigned to roles. Moreover, we identify two distinct phases in RBAC. The first is the design where a system administrator can define a number of assignments between the elements in the computer system. The second phase is the run-time phase where the assignments in the system are enforced by the model as it is specified by the security policy of the system, which was prescribed during the design phase. More of RBACs virtues are the support of important principles, namely the least privilege, separation of administrative functions and separation of duties [4]. However pure RBAC model is inappropriate to model security policies that interpret complex and ambiguous Big Data scenarios. It must be extended to face the new security challenges of this technology.

3 BLOCKCHAIN CONCEPT

Originally introduced by Satoshi Nakamoto in 2008 [5] to underline the Bitcoin cryptocurrency network, the blockchain is a computational paradigm that consists of a distributed ledger which contains all transactions ever executed within its network, enforced with cryptography and carried out collectively by a peer-to-peer nodes. The blockchain has taken up the bulk of technology industry and financial world attention. As a secure and decentralized computational infrastructure, it is widely acknowledged as a disruptive and efficient solution for the problems of centralization, privacy and security when recording tracking, monitoring, managing and sharing not only financial transactions but also any other value such as birth and death certificates, marriage licenses, deeds and titles of ownership, educational degrees, financial accounts, medical procedures, insurance claims, votes, provenance of food, and anything else that can be expressed in code [6] [7] [8].

3.1 The blockchain applications

Even though the blockchain gained an initial success with the emergence of Bitcoin and crypto-currency field, which existed now for about nine years, we are still at the beginning of fully understanding its potential. As a matter of fact, innovative solutions that go beyond crypto-currencies have emerged. The blockchain technology has been useful in building decentralized trustless applications in a lot of different fields and launching many open source projects and startup's initiatives. As shown in the table below, the Blockchain is used in a diversity of domain applications, such as public services, reputation systems, digital assets ,Internet of Things (IoT), and Big Data rather than financial systems only [9] [10].

Table 1: Blockchain application beyond the cryptocurrency field

Domain use case	**Example of applications**
Identity management	Public key management [11] Public domain name system [12] *Pretty Good Privacy (PGP) SYSTEM* [13]
Access control	Enigma [14]
Reputation system	[15]
Storing system	[16]
Financial and business services	Fintech [17]
Public services	Online education [18] Kudos [19]
Big Data	HEALTHCARE [20] SMART GRID [21] Energy saving [22]
Others	Software license [23] SECURITY : a novel anti-malware [24]

Thanks to its specific characteristics which are the following: trust-free, transparency, highly resistant to outage, tamper-proof, auditable, and self-regulating system, with no human intervention required to execute computation and in line with the principles of *privacy-by-design, and "security trough transparency".* The blockchain is able to efficiently ensure integrity, authenticity, and auditability of all transactions. It can hence play a major role in the Big Data ecosystem, reducing time and cost consuming workflows and providing an architectural layer where data can be processed and analyzed, but remain private or semi-private [25]. In this case trust is not established by powerful intermediaries like banks, governments and technology companies, but rather through mass collaboration and cryptographic mechanisms.

3.2 The blockchain technology: definition

The blockchain, in form, consists of blocks, holding batches of individual transactions. Each block contains a timestamp and a link to the immediately previous block via a reference that is essentially a hash value of the previous block called *parent* block. The first block of a blockchain is called a genesis block, which has no parent. While, in essence, it could be considered as a type of distributed database designed to process in time-ordered way data such as digital transactions. However, it is different from ordinary scalable distributed database such as such as MySQL Cluster, MongoDB and Apache HBase by two main features:

***Cryptography-by-design*:** this means that the identity of users, integrity of the ledger and authenticity of data are achieved in cryptographic way.

Lack of central party: the functionality and security of the blockchain is enforced in a distributed and public way rather than relying on a central authority to do so as it is the case for the aforementioned databases.

3.3 The blockchain building blocks and mechanisms:

Blockchain is a distributed database for transaction processing. All transactions in a blockchain are stored into a single ledger. The blockchain technology is built on top of four fundamental building blocks, each building block has key properties, and each property is achieved through specific mechanisms.

We consider that the blockchain technology relies on the following building blocks as described below:

***Identifying the source and destination of a transaction*:** in a blockchain based ecosystem, users serve from digital identities called "addresses" to send and receive transactions. Those addresses should have the following key feature:

-*Self-issued and independent*: users should be able to self-generate anonymous (e.g. using software) transaction identities (e.g. a cryptographic hash of the public-key, in Bitcoin system) in an independent way from any given authority (e.g. government, businesses, etc.)

-*Anonymous*: transaction identities should be anonymous in the sense that they would reveal nothing about the real identity of the owner to meet the needs of user privacy. True anonymity in digital identities requires more than the self-issuance feature, but also the possession of the **unlinkability** and **intractability** features.

-*Privacy-preserving verifiable*: any entities in the system should be able to verify the security of the digital identity. A publicly verification algorithms to validate the source of trust for any given (anonymous) transaction-identity is required while preserving the privacy of the owner of the identity.

-*Transactions*: A transaction records the transfer of a value from a source address to a destination addresses. Transactions are generated by the sender and broadcasted to the network of peers. Transactions are invalid unless they have been recorded in the public history of transactions i.e. the blockchain. Ultimately, transaction processing with blockchain technology should satisfy the following properties:

-*Irreversible*: Once a transaction has entered the ledger, it should be impossible to modify its information, or delete it, which would effectively reverse the transaction.

-*Publically verifiable*: The verification algorithm of a transaction should enable any node in the network to verify the validity of the transaction.

-*Immunity*: Once a transaction is recorded in the blockchain it canwon't allow any alteration without it being detected and rejected by the other nodes in the network. In addition, if a transaction conforms to a ledger protocol, it should be eventually added to the ledger.

-*Condition for auto-processing a transaction*: The transfer of any value (e.g. altcoins, tokens) with the blockchain or the execution of any function through the blockchain should be locked by logic conditions which are written as a code and automatically executed by nodes in the network**. *This condition should be self-executed*.**

***-Consensus*:** Every user or node in the network relies on algorithmically enforced rules to process transactions with no human interaction required to verify in an independent way the correct execution of the protocol, and obtains the same results.

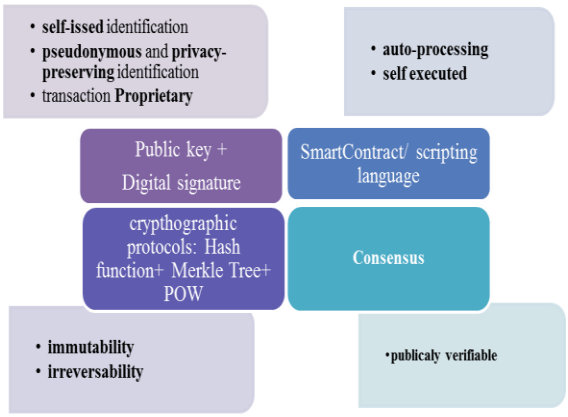

Fig. 1. Blockchain buildings blocs and mechanisms

Each node has exactly the same ledger as all of the other users or nodes in the network. This ensures a complete consensus from all users or nodes in the corresponding currencies blockchain.

3.4 The blockchain mechanisms:

To satisfy the properties cited above, a suit of cryptographic mechanisms has been introduced, as detailed below:

Public key cryptography: enables a ***self-issued, pseudonymous*** and a *privacy-preserving* identification

Digital signature: is used to satisfy the feature of *transaction proprietary and publically verifiable properties*: Each user in the system possesses at least a pair of private and public keys; the public key is published publically to determine the digital identity, while the private key is used to sign the corresponding transaction by its owner (or a trusted party on his behalf). The validity of the signature is tied to the knowledge of the sender's private key, *(proprietary)*

• A signature can be verified by anyone knowing the sender's public key, *(publically verifiable)*

• A signature is invalidated if any parameters of the transaction are changed.

Hash function [26] and Merkle tree [27]: are used to solve the problem of *Immutability*, *irreversibility* and *public verification* of transactions in blockchain systems. Those features are achieved by distributing transactions into time-ordered blocks and time stamping each of these blocks by its cryptographic hash as shown in *Figure 2*.

Transactions are summarized in each block using a Merkle tree, also known as a binary hash tree. It is an efficient data structure that serves to summarize and verify the integrity of large sets of data. Organizing blocks in an ordered chain called (blockchain), where each block refers to the previous one, is a clever technique introduced by Nakamoto to make it impossible to delete or replace the whole blocks. To simplify blockchain verification, key block parameters (such as a Merkle root) are collected in a block header. Thus, providing immutability of transactions is similar to providing immutability of block headers.

The *Immunity* feature of transactions is achieved through the properties of hashing functions. Actually, any alteration in the transaction results in changes in the Merkle root of the block, which is a part of its header. Hence, to change a single block, an attacker must also change all succeeding blocks in the ledger, which is infeasible.

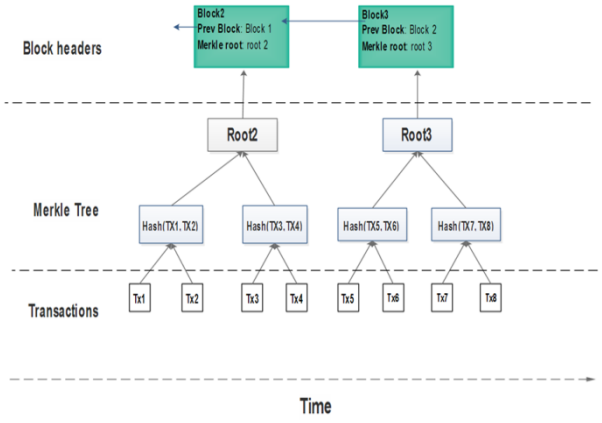

Fig. 2. Blockchain buildings blocs and mechanisms

Proof of X where X ∈ (Work/ stack/Space etc): Once the transactions have been created, it is broadcasted in the network and each node verifies independently the received transaction and includes it in a block. However, each node may have different view of the blockchain leading to a divergence of branches of a blockchain. To mitigate this problem, there is a need of a distributed mechanism to reach consensus among the untrusted participants in the network. This problem is historically known as "Byzantine Generals "(BG) problem, which was raised in [28]. In BG problem, a portion of Byzantine army commanded by group of generals, circle the city. While some generals opt for attacking other generals opt for retreating. Knowing that the attacks would fail if only part of the generals attack the city. Hence, they all need to come to a unanimous agreement to either attack or retreat. Reaching a consensus in such a divided

environment is a challenging task. However, the *consensus* of the network could be achieved in a variety of ways, including proof of work (e.g., as used in Bitcoin), proof of stake (e.g. Nxt), delegated proof of stake (e.g. BitShares) and Practical byzantine fault tolerance (hyper ledger project). For more details, we refer the reader to a comprehensive survey of those four approaches in [9] and [10].

***Smart Contract/ scripting language*:** are used to satisfy the *auto-processing properties*. The scripting language describes the execution of a certain program on a stack machine. Each transaction contains a script which locks, or encumbers, the value transferred by this transaction. A script is a part of each input and each output of a transaction. When we generalize this scripting language computation to arbitrary Turing complete logic, we obtain an expressive smart contract system. The interest in smart contract applications steadily rose since 2014 due to the appearance of Bitcoin-like technologies, such as Etherum [29] and many other works designed specifically to decentralized smart contract systems. Smart Contracts are self-executed cryptographic "boxes" that are stored in the blockchain and contain value, which is only unlocked if certain conditions are met.

3.5 The blockchain taxonomy:

Blockchains can be classified based on several criteria:

Who has access to the network?: based on this criterion, we can classify the blockchain within the following categories:

***Public*:** as defined in [30] "*A public blockchain is a blockchain, in which there are no restrictions on reading blockchain data (which still may be encrypted) and submitting transactions for inclusion into the blockchain*".

Private*:** as defines in [30] ***"*A private blockchain is a blockchain, in which direct access to blockchain data and submitting transactions is limited to a predefined list of entities"*.

***Who can transact or mine*?:** within this criteria we can specify a permissioned, or permissionless blockchain as defined below:

Permissionless: as defines in [30] ***"****A permissionless blockchain is a blockchain, in which there are no restrictions on identities of transaction processors (i.e., users that are eligible to create blocks of transactions)".*

Permissioned: as defines in [30] ***"****A permissioned blockchain is a blockchain, in which transaction processing is performed by a predefined list of subjects with known identities".*

Transaction type: the blockchain can support two types of transactions: a *Bitcoin-style transactions (UTXO model) or smart contracts (account-based model)*:

UTXO model: in the UTXO model, the fundamental building block of a cryptocurrency transaction is an unspent transaction output, or UTXO. UTXO are a value of the currency locked to a specific owner, recorded on the blockchain, and recognized as currency units by the entire network. The UTXO consumed by a transaction are called transaction inputs, and the UTXO created by a transaction are called transaction outputs. The recipient is identified through their public key, so cryptocurrency transactions can be traced throughout the blockchain, to the beginning of the creation of the cryptocurrency. This forms the mechanism for checking the ownership of cryptocurrency. This model is uniquely suited for the transfer of and tracking of digital tokenized assets.

Account-based model: The intent of the account based model is to merge together and improve upon the concepts of scripting, altcoins and on-chain meta-protocols, and allow developers to create arbitrary consensus based applications that have the scalability, standardization, feature-completeness, ease of development and interoperability offered by these different paradigms all at the same time. The blockchain in this model is built-in Turing-complete programming language, allowing anyone to write smart contracts and decentralized applications where they can create their own arbitrary rules for ownership, transaction formats and state transition functions.

3.6 Type of the blockchain adopted in this project:

The type of blockchain we are using to develop our framework in this thesis is a *public* and *permissionless* one. There are no restrictions on reading and submitting transactions for inclusion into the blockchain.

However, we note that, our framework could also be used in private and permissioned blockchain since the way we are leveraging the blockchain to enforce access control policies is independent on the conception layer of the blockchain (consensus protocol, etc).

4 OUR FRAMEWORK

4.1 Introduction:

We present a new distributed access control framework based on Blockchain technology that combines, for the first time, access control models and cryptocurrency blockchain mechanisms. We propose the use of Smart contract [29] to express fine-grained and contextual access control policies to make authorization decisions. We opt for authorization tokens as an access control mechanism, delivered through emergent cryptocurrency solutions. We use blockchain to ensure enforcing access policies in distributed environments where there is no central authority/administrator, and guarantee that policies will be properly implemented by all interacting entities.

The main features of our framework are:

-Pseudonymity and Unlikability: The use of pseudonymous Bitcoin-like addresses that are publicly verifiable yet unlinkable to user identities allows privacy-preserving access control achievement since no real-world name or identifying information are required.

-Lightweight: Actually, enforcing access control policies by the blockchain relieves Big Data constrained nodes from the burden of handling a vast amount of access control-related information and at the same time mitigates the need for outsourcing these functionalities to a trusted powerful entity. In fact, nodes only need to easily verify the validity of the access token securely held on the blockchain. Moreover, using digital keys as identification and authentication at once reduces communication cost, since no further authentication mechanisms are required to get the token. Only signatures are sufficient.

***-User driven & transparency*:** the user is the only one who is responsible to define the granular access control policies over his own data.

-Distributed nature and the lack of a central authority: Each node in the network shares its data with others nodes directly, without the intervention of any third or trusted entity

-Fine-granularity: we argue that our framework is generic and not restricted to the use of any specific access control model. Actually, the use of smart contact enables the framework to implement an expressive and granular access control policies, expressed by any access control model as soon as this model could be transcoded to a script language.

4.2 Preliminaries:

We will denote key pairs using capital letters (e.g.**A**), and refer to the private key and the public key of *A* by: *A.sk* and *A.pk*, respectively (then we can write: *A= (A.sk, A.pk)).* In addition, let $sig_A(m)$ denote a signature on a message m computed with *A.sk*. In addition, let $check_A(m, \sigma)$ denote the result *(true or false)* of the verification of the signature σ on the message m with respect to the public key *A.pk* and finally we note $\mathcal{H}$ as a hash instantiated by a SHA-256 [31] implementation. The acronyms we use to describe the functionalities of our proposed framework and their meanings are summarized in Table 2:

Table 2: Acronyms Used in Our Framework

Acronym	Its meaning
IDx	The index of the current transaction Tx where x = H (Tx) , H is a hash function
rs	The address of requested resource.
rq	The address of the requester who is the receiver of the current transaction
πx	Address of SSmartContract (access control policies written in scripting language)
ψ	unlocking data(considered as input of smart contract
$TKN_{rq,\pi'}$	access token generated by SmartContract πx
ref	point to the previous transaction

4.3 Framework Model Layer:

Our proposed authorization framework is generic and not restricted to the use of any specific access control model. Actually, the access control policy could be expressed with any access control model. But, the only condition the model has to meet in order to be integrated in our framework is to be transcoded to a script language and encapsulated inside the framework's transactions. Hence, in order to get access to a protected resource, an access token is required. The access token is not delivered by this transaction unless the requester meets all access control conditions already described with the model and included in the transaction. However, in order to improve privacy in our framework we will use attribute based access control model (ABAC) to express access control policies.

4.3.1 Authorization Architecture:

Big Data is considered as a collaborative environment. We basically distinguish two levels of the centralization or decentralization of access control: 1). The first level concerns the management of access policies over operations between cooperative organizations: whether a centralized entity is defined as a common authority. Hence, all decisions about whether a user in a cluster "A" is able to access a given resource in a cluster "B", are taken and implemented at a central point which can be the master node of a central cluster, and the decentralized or chain approach where each cluster is responsible for defining its own access control decisions and their implementations. 2) The second level concerns the management of access control within a cluster. But in this case, the management and localization of access control logic could be centralized on a central and dedicated point. Or distributed and located in each protected resource.

In our framework, we propose a centralized and decentralized approach to manage access control policies in Big Data. Indeed, we propose a decentralized approach at the first level that concerns the interactions between cooperatives organizations. At this level, we opt for a peer-to-peer approach (fully distributed) where each cluster is responsible of defining and implementing its own security policies. Besides that, at the second level, we opt for a centralized approach where a centralized entity, called Authorization Manager Point (AMP), is defined for each cluster. This AMP manages and approves authentication and authorization data for a resource. It can be responsible for a single or multiple nodes or even for a whole network and a Resource may have multiple AMP.

In order to comply with the privacy preserving objective, aiming to empower and emphasis user involvement in defining and managing the access control process over its own resources, we introduce the Resource Owner (A-RO) and (B-RO) in cluster A and B respectively. Those later decide the security policies and the level of authorization's granularity of their respective endpoints belonging to the same cluster. They are the only one responsible for controlling and managing their corresponding AMP. (A-RO): is the person who is in charge of the requested resource and controls its access permissions. (B-RO): The person who is in charge of requesting the resource. He controls the requests a resource in Organization B (B-R) makes. Our proposed architecture is depicted in *Figure 3*:

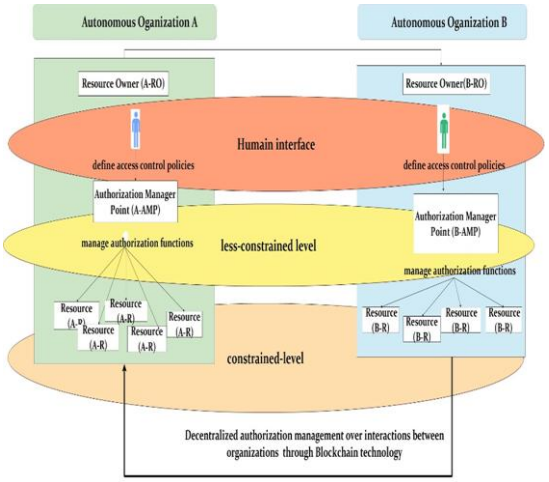

Fig. 3. Architecture Layer

4.3.2 Authorization Mechanism:

A mechanism defines the low-level functions to enforce policies and define how access requests are evaluated against those policies. In our proposed framework, we will use the following mechanisms:

We use, *Bitcoin-like addresses*: to identify all interacting entities. *Smart contract (a.k.a chain code)* to express fine-grained and contextual access control policies enveloped inside transactions. We opt for *authorization tokens* distributed by the blockchain. We use *blockchain* firstly to ensure evaluating and enforcing access policies and secondly to ensure token integrity and detect token double spending. The detail of each mechanism is described below:

Address generation: The wallet generates pairs of keys consisting of a private key *A.sk* and a public key *A.pk*. The private key A.sk is a number, picked randomly from which an elliptic curve multiplication and a one-way cryptographic function is used to generate a public key *A.pk*. Then from the public key *A.pk*, a one-way cryptographic hash function $\mathcal{H}$ is used to generate an address. The private key is then used to create signatures that are required to prove ownership of a resource. We opt for the following hierarchical and deterministic method, as illustrated in *Figure 4*, to generate addresses identifying our framework entities: keys are derived in a tree structure, such that a parent key can derive a sequence of children keys, each of which can derive a sequence of grandchildren keys, and so on, to an infinite depth. Then, the resource's address has to be extracted

from a child public key of the seed corresponding to its Resource owner. Actually, this method gives to our framework the following characteristics:

1) ***Great scalability*****:** from one seed key, the RO is able to identify unlimited number of resources, by extracting address from the generated keys.

2) ***High involvement and transparency of RO*****:** In fact, with the ability to derive public child keys from public parent keys, without having the private keys. We can produce an infinite number of public keys and addresses, but cannot give the possibility to produce valid transaction without the permission of the RO who has generated the addresses. That means: the RO who has generated the seed is the only person who can define and manage access control policies for all his registered resources. By deriving the corresponding private keys to sign transactions.

3) ***Pseudonymity and unlinkability:*** The Resource Owner can generate different addresses to the same resources from another seed key, for each transaction which enhance unlikability and pseudonymity.

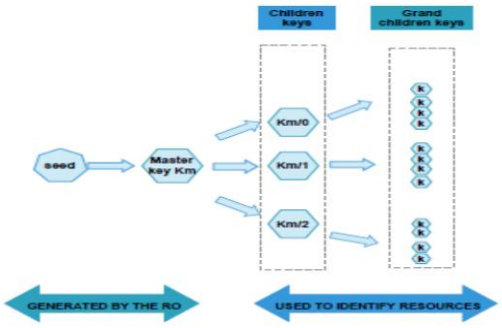

Fig. 4. Hierarchical and deterministic key generation: a tree of keys generated from a seed

Authorization token**:** In our framework, we define an Authorization Token as a data structure that represents the access right or the entitlement defined by the creator of the SmartContract, generating the authorization token, to entity interacting with this SmartContract in order to access a specific resource identified by its address. If the authorization token presented by the device was delivered by a SmartContract that matches with the one that is associated with the other node or service that manages that node, then the access is granted. Actually, the token-based access control enforced by the blockchain technology provides many advantages in Big Data context. In fact, having the token securely held on the blockchain means nodes can easily verify the validity of the access token relieving Big Data nodes from the burden of handling a vast amount of access control-related information and at the same time mitigates the need for outsourcing these functionalities to a trusted powerful entity that prevent end-to-end security to be achieved. Moreover, it reduces communication cost, since no further authentication mechanisms are required to get the token. Since only signature is sufficient. Similarly, the resource owner can manage and updates access to his numerous and heterogeneous devices by placing a resource hash of the requested data on the blockchain. Then the device can read the hash and check the update access control configurations. This hash can be updated with every new connection to the blockchain. In addition, the token could be used for many access control operations such as getting, delegating, revoking and even updating access in an easier and flexible way.

Blockchain: Our Framework provides several useful mechanisms using the blockchain. In fact, the blockchain is considered as a database or a policy retrieval point, where all access control policies are stored in the form of transactions, it serves also as logging databases that ensures auditing functions. Furthermore, it prevents forgery of tokens through transactions integrity checks and detects token reuse through the double spending detection mechanism.

SmartContract: Nick Szabo introduced this concept in 1994 and defined a smart contract as ``a computerized transaction protocol that executes the terms of a contract''. Szabo suggested translating contractual clauses (collateral, bonding, etc.) into code, and embedding them into property (hardware, or software) that can self-enforce them, so as to minimize the need for trusted intermediaries between transacting parties, and the occurrence of malicious or accidental exceptions. In the blockchain context, SmartContracts are scripts stored on the blockchain. Since they reside on the chain, they have a unique address. We trigger a smart contract by addressing a transaction to it. It then executes independently and automatically in a prescribed manner on every node in the network, according to the data that was included in the triggering transaction. (This implies that every node in a smart contract enabled blockchain is running a virtual machine (VM), and that the blockchain network acts as a distributed VM.)

The SmartContract has its own account on the blockchain, and the blockchain supports an *account-based model* [32]. It allows us to express logic functions in code. It operates as *autonomous actors*; whose behavior is completely predictable. As such they can be trusted to drive forward any

on-chain logic that can be expressed as a function of on-chain data inputs, provided that the data they need to manage is within their own reach (in the example above, the contract wouldn't be able to trade assets that it did not own). In addition, the code can be inspected by every network participant since all the interactions with a contract occur via *signed* messages on the blockchain. All the network participants get a cryptographically verifiable trace of the contract's operations.

This enables our framework to express fine grained access control policies. A policy is a set of rules and conditions (based on a specific context or attribute, etc) that a requester entity has to fulfill in order to obtain the Access Token and get access to the specific resource. These rules could be expressed by any access control model but must be transformed to a script language considered as locking script placed on the output of a transaction. Fortunately, new blockchain protocols are being developed including full Turing completeness capability, allowing anyone to write smart contracts and decentralized applications with their own arbitrary rules for ownership, transaction formats and state transition functions [29]. The use of those advanced language such as Etherum, will certainly enable our framework to express fine-grained context-aware access control policies. We believe that Smart contracts are a promising emergent field to express, granular, contextual and contractual access control model in general and in Big Data in particular.

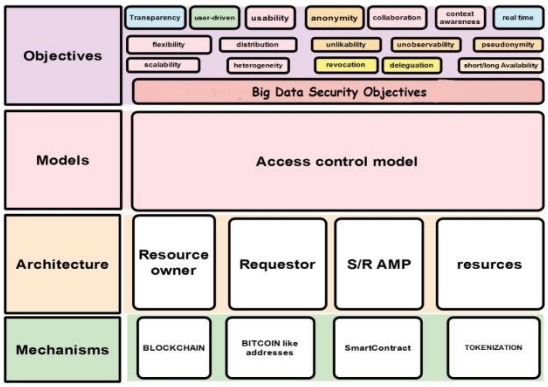

Fig. 5. Framework's Reference Model

Hence, the vision of our decentralized access control framework is a system of autonomous clusters hinged around one or many Resource Owners in possession of one or many resources identified with addresses and interacting between each other through transactions (requesting, granting, delegating and revoking access) under the control of their RO. The blockchain is a ledger keeping track and ensures the validity of access transaction among interacting organization. Each manages its own access policy, under only the control of his Resource Owner.

5 CONCLUSION AND FUTURE WORKS

At this step, we demonstrated the feasibility of using blockchain technology to manage access control process for Big Data through the description of our proposed framework. It leverages the salient features of Blockchain that are, distribution, full-fledged and append-only ledger to make a promising solution for addressing the aforementioned access control challenges in Big Data. It has successfully achieved the following objectives:

1) Distributed nature and the lack of a central authority. User-driven and transparency.
2) Lightweightness.
3) Fine-granularity.
4) Pseudonymity and Unlinkability.

However, adopting the blockchain technology to handle access control functions is not straightforward and additional critical issues emerge that are:

***-The public aspect of the blockchain*:** The transparent nature of blockchain may initially appear to be at odds with privacy. In truth, the transparent aspect of the blockchain makes all of the transaction history public, which means that in our framework all the authorization operations history and access control policies will be public. While the policy could be a commercial secret and knowledge of user's policy would compromise his strategy and invite unwelcome imitators. This is indeed a valid concern However; there are some solutions to reconcile the two aspect that are: transparency and privacy. We cite below some of many possible examples.

Secure MultyParty Computation (sMPC) [33]: allows data to be used while its privacy is still guaranteed. Therefore, a program could be evaluated while the inputs are kept secret MIT's enigma project instantiate this idea. Authors propose to split data into shares (e.g., using Shamir's Secret Sharing [31]), rather than encrypting them, then they could use secure Multiparty Computation (MPC) to securely evaluate any function including SmartContract.

Succinct Non-interactive ARguments of Knowledge (SNARKs) [34], [35], [36]: Zero-knowledge proofs allow a user to construct a mathematical proof such that: given a program, when executed on some (possibly hidden) input known by the user, has a particular (publicly

known) output, without revealing any other information. A recent work called 'HAWK' introduced in [37] proposes a framework for building privacy-preserving smart contracts using SNARKs

***Private Blockchain solution*:** In a private chain, we can control all nodes as well as their access. Then, we can encrypt the data in a way that makes it unreadable to the outside world. Mijin[1] and IBM-Hyperledger project [38] are examples of how private blockchain will work

***Traceability*:** Users are recommended to employ many identities to enhance their privacy. Meanwhile, an increasing body of research shows that anyone can de-anonymize transactions by using information in the blockchain [39] [40], such as the structure of the transaction graph as well as the value and dates of transactions. As a result, this version fails to offer a robust anonymous exchange.

This paper presented the foundation of our Blockchain based access control, a lot more can be done to enhance this framework, the next step is doing a case study on a Big Data platform where the framework is implemented and tested. We will focus in later phases of this project on the authentication and accountability aspects to make a more complete access control system.

6 REFERENCES

[1] IBM, "What is Big Data," 2017. [Online]. Available: https://www-01.ibm.com/software/data/bigdata/what-is-big-data.html.

[2] L. Okman, N. Gal-Oz, Y. Gonen, E. Gudes, J. Abramov, "Security issues in nosql databases," in Trust, Security and Privacy in Computing and Communications (TrustCom), 2011 IEEE 10th International Conference on, 2011, pp. 541–547..

[3] R. S. Sandhu, Edward J. Coyne, Hal L. Feinstein, Charles E. Youman, "Role-based Access Control," IEEE Computer, vol. 29, no. 2, pp. 38-46, 1996.

[4] P. Samarati, S. D. C. Di Vimercati, "Access Control: Policies, Models, and Mechanisms," Foundations of Security Analysis and Design, vol. 2171, pp. 137-196, 2001.

[5] S. Nakamoto, "Bitcoin : A Peer-to-Peer Electronic Cash System," pp. 1-9, 2008.

[6] M. Swan, "Blockchain: Blueprint for a new economy," 2015.

[7] M. Ulieru, "Blockchain 2.0 and Beyond: Adhocracies," Banking Beyond Banks and Money, pp. 297-303, 2016.

[8] M. Crosby, P. Pattanayak, and S. Verma, "Blockchain technology: Beyond bitcoin," 2015.

[9] Z. Zheng, S. Xie, H. Dai, and H. Wang, "Blockchain Challenges and Opportunities: A Survey," 2016.

[10] F. Tschorsch, B. Scheuermann, "Bitcoin and beyond: A technical survey on decentralized digital currencies," IEEE Communications Society, pp. 2084 - 2123, 2015.

[11] C. Fromknecht, D. Velicanu, and S. Yakoubov, "A Decentralized Public Key Infrastructure with Identity Retention," IACR Cryptology ePrint Archive, 2014.

[12] M. Ali, J. Nelson, R. Shea, and M. Freedman, "Blockstack: Design and Implementation of a Global Naming System with Blockchains," Last Visit, 2016.

[13] D. Wilson, G. Ateniese, "From Pretty Good to Great: Enhancing PGP Using Bitcoin and the Blockchain,," Springer International Publishing, p. 368–375, 2015.

[14] G. Zyskind and A. S. Pentland, "Decentralizing Privacy : Using Blockchain to Protect Personal Data," 2015.

[15] A. Schaub, R. Bazin, O. Hasan, and L. Brunie, "A trustless privacy-preserving reputation system," ICT Systems Security and Privacy Protection , pp. 398-411, 2016.

[16] S. Wilkinson, J. Lowry, and T. Boshevski, "Metadisk a blockchain-based decentralized file storage application," 2014.

[17] Y. Guo and C. Liang, "Blockchain application and outlook in the banking industry," Financial Innovation, 2016.

[18] P. Devine, "Blockchain learning: can crypto-currency methods be appropriated to enhance online learning?," 2015.

[19] M. Sharples and J. Domingue, "The Blockchain and Kudos: A Distributed System for Educational Record, Reputation and Reward," European Conference on Technology Enhanced Learning, pp. 490-496, 2016.

[20] A. Ekblaw, A. Azaria, J. Halamka, and A. Lippman, "A Case Study for Blockchain in Healthcare:'MedRec' prototype for electronic health records and medical research data," 2016.

[21] M. Mihaylov, S. Jurado, and N. Avellana, "NRGcoin: Virtual currency for trading of renewable energy in smart grids," European Energy Market (EEM), 2014.

[1] http://mijin.io/en/about-mijin

[22] L. Johnson, A. Isam, N. Gogerty, and J. Zitoli, "Connecting the Blockchain to the Sun to Save the Planet," 2015.

[23] J. Herbert and A. Litchfield, "A Novel Method for Decentralised Peer-to-Peer Software License Validation Using Cryptocurrency Blockchain Technology," Australian Computer Society, pp. 27-35, 2015.

[24] C.Noyes, "BitAV: Fast Anti-Malware by Distributed Blockchain Consensus and Feedforward Scanning," arXiv:1601.01405, 2016.

[25] J. Herrera-Joancomartí and C. Pérez-Solà, "Privacy in Bitcoin Transactions: New Challenges from Blockchain Scalability Solutions," Modeling Decisions for Artificial Intelligence, pp. 26-44, 2016.

[26] J. Carter and M. Wegman, "Universal classes of hash functions," Journal of Computer and System Sciences, 1979.

[27] R.Merkle, "A digital signature based on a conventional encryption function," Conference on the Theory and Application of Cryptographic Techniques, pp. 369-378, 1987.

[28] L. Lamport, R. Shostak, and M. Pease, "The Byzantine generals problem," ACM Transactions on Programming Languages and Systems , vol. 4/3 , pp. 382-401 , 1982.

[29] V.Buterin, "Ethereum : A next-generation smart contract and decentralized application platform," 2014.

[30] BitFury Group and J. Garzik, "Public versus Private Blockchains. Part 1: Permissioned Blockchains,," pp. 1-23, 2015.

[31] A.Shamir, "How to share a secret," Communications of the ACM, vol. 22, no. 11, pp. 612-613, 1979.

[32] V.Buterin, "Ethereum DEV plan," p. 35, 2014.

[33] A. Ben-David, N. Nisan, and B. Pinkas, "FairplayMP," Proceedings of the 15th ACM conference on Computer and communications security - CCS '08, p. 257, 2008.

[34] E. Ben-Sasson, A. Chiesa, D. Genkin, E. Tromer, and M. Virza, "SNARKs for C: Verifying Program Executions Succinctly and in Zero Knowledge,," Springer Berlin Heidelberg, pp. 90-108, 2013.

[35] R. Gennaro, C. Gentry, B. Parno, and M. Raykova, "Quadratic Span Programs and Succinct NIZKs without PCPs,," Springer Berlin Heidelberg, pp. 626-645, 2013.

[36] B. Parno, J. Howell, and C. Gentry, "Pinocchio: Nearly practical verifiable computation," in Proceedings of the IEEE Symposium on Security and Privacy , 2013.

[37] A. Kosba, A. Miller, E. Shi, Z. Wen, and C. Papamanthou, "Hawk: The Blockchain Model of Cryptography and Privacy-Preserving Smart Contracts," IEEE Symposium on Security and Privacy, 2016.

[38] C.Cachin, "Architecture of the Hyperledger Blockchain Fabric," 2016.

[39] F. Reid and M. Harrigan, "An Analysis of Anonymity in the Bitcoin System," Security and Privacy in Social Networks, p. 197–223, 2013.

[40] D. Ron and A. Shamir, "Quantitative Analysis of the Full Bitcoin Transaction Graph," Springer Berlin Heidelberg, pp. 6-24, 2013.

[41] in Communication, Control, Signal Processing and Computing Applications (C2SPCA), Bangalore, 2013, pp. 1-4.

[42] Nouredine Seddiki, Bassou Abedsalem, "Study of the Performance of Multi-hop Routing Protocols in Wireless Sensor Networks," International Journal of Advanced Computer Science and Applications (IJACSA), Vol. 8, No. 2, 2017.

[43] A. K. Singh, S. Rajoriya, S. Nikhil and T. K. Jain, "Design constraint in single-hop and multi-hop wireless sensor network using different network model architecture," International Conference on Computing, Communication & Automation, Noida, 2015, pp. 436-441.