Journal of Ambient Intelligence and Humanized Computing
https://doi.org/10.1007/s12652-021-03522-0

ORIGINAL RESEARCH

Distributed authentication framework for Hadoop based bigdata environment

M. Hena[1] · N. Jeyanthi[1]

Received: 22 June 2020 / Accepted: 23 September 2021

Abstract
Big data, the upcoming technology in the field of computing, refers to a large complex dataset. It deals with large complex datasets and yields great valued information when analysed properly. Data Security has become the greatest challenge in the minds of cyber experts and researchers in this scenario. Apache Hadoop frameworks that let distributed processing of these large datasets rely on Kerberos Authentication for mutual authentication and verification. The protocol comes with inherent challenges like Single point of failure, Dictionary Attacks, Replay Attacks, and Time Synchronization problems. This paper puts forward a one-off approach based on recent technologies like Blockchain Networks, Digital Signatures, and Elliptic ElGamal and Threshold Cryptosystem. The proposed scheme aims to mainly deal with the Single Point of Failure problem. Riverbed Modeller (AE) simulation is performed to do the comparative study of the proposed scheme with existing systems that use traditional encryption standards like RSA cryptosystems. Analysis of the simulation results proves that the proposed scheme is more efficient in terms of time and memory without compromising the level of security offered. The response time, network delay and traffic rates of the proposed system are compared with the existing RSA based system and the results strengthen the claims of this work. Lastly, the results from comparative analysis of security features and computational time cost indicate that the proposed method heightens the security level offered for big data systems with a nominal effect on performance.

Keywords Big data · Apache Hadoop · Kerberos · Blockchain · Digital signature · Threshold ElGamal cryptosystem

1 Introduction

Big data and data security are the utmost conversed buzzwords across the globe. Big data analytics deals with those enormous multifaceted datasets that are hard to handle using traditional data handling and management systems. The user authentication and verification strategies vary from organization to organization, whether it's private or public sector, to safeguard the data and avoid unlawful access. Apache Hadoop is one of the most widely used frameworks for big data storage and analytics. Hadoop deploys the capabilities of the Kerberos Protocol for mutual authentication and verification.

✉ M. Hena
henashabeebvit@gmail.com

N. Jeyanthi
njeyanthi@vit.ac.in

[1] School of Information Technology and Engineering, VIT, Vellore, Tamilnadu, India

However, the Kerberos Authentication protocol comes up with many inherent security weaknesses and a variety of solutions have already been put forward by the research community and security experts in this regard (Sutradhar et al. 2018) (Gharib and Belloulata 2014). Single Point of Vulnerability or Failure (SoV / SoF), Password Guessing or Dictionary Attacks, Replay Attacks, and Time Synchronization problems are the hurdles of Kerberos protocol that prevents its widespread acceptance. Single point of Vulnerability or Failure (SoV / SoF) is considered as the most overwhelming of all the above-mentioned challenges because of the dependence of the protocol on a Trusted Third Party Server (TTP) called Key Distribution Center (KDC). Any failure to the Key Distribution Center (KDC) can cause the failure of the entire system. This is not a fortunate scenario for the big data systems, which emphasis quick, trustworthy, and real-time data processing insights and user access. Kerberos Authentication Mechanism deploys symmetric key cryptography to provide mutual authentication and authorization of client-server applications. The Key Distribution Centre (KDC) consists of two

Published online: 11 October 2021

parts—the Authentication Server (AS) and the Ticket Granting Server (TGS). These servers grant the ticket and session keys to the user to access the secured server service. All user credentials are put in a local database at the Key Distribution Centre (KDC). Thus, the Key Distribution Centre (KDC) is always prone to attacks and the entire system collapses if it is compromised. This paper put forward a novel mutual authentication mechanism which is a blend of modern technologies like Blockchain technology, Digital Signatures, Elliptic Curve ElGamal and Threshold Cryptosystem. The scheme is intended to wipe off the threat of Single Point of Failure in Kerberized Hadoop Clusters. The local database at the Key Distribution Centre (KDC) is replaced with a blockchain network which is a distributed database. Upon successful registration of the user, all the associated keys and credentials are posted to the blockchain for storage. The registered user sends a digitally signed access request to the Authentication Server, which verifies the signature of the user retrieving the keys stored in the blockchain. ElGamal Digital Signature Algorithm is used to generate the digital signatures here. After successful initial authentication, the Authentication Server grants the Ticket Granting Server Ticket (TGT) to the user. The user can send the request to access the secured Hadoop server using this ticket to show its authenticity. Here, instead of one Ticket Granting Server (TGS) as in the traditional Kerberos System, multiple Ticket Granting Servers (TGSs) are deployed. (t, n) Threshold ElGamal cryptosystem insists on the cooperation of at least 't' number of these 'n' Ticket Granting Servers (TGSs) to proceed further. Upon successful authentication, the user is allowed to access the Kerberized Hadoop Cluster.

1.1 Related works

Researches have been carried out throughout academia and the industrial world to safeguard big data from security concerns. The techniques include—Monitoring and Auditing Techniques, Anonymization Techniques, Access Control, Encryption, and Mathematical computations. In the paper by Wu and Guo (2013) mentioned that many applications are nowadays collecting data with or without the knowledge of users/owners. The providers of these apps are only charging for the service they provide. The profit they are making out of these huge volumes of data collected daily is mentioned nowhere. Also, the user would not know at any time who all are going to see the confidential data. Dong et al. (2015) proposed Secure Sensitive Data Sharing on a Big Data Platform. The authors tried to handle the security issues that can take place from the insider attacks of CSP's employees. The method splits the le and data is stored disjointedly in the distributed cloud servers. The concept was enhanced in Li et al. (2017b) by an Intelligent Cryptography method for secured distributed big data storage in the Cloud environment by adding an encryption module. Liang et al. (2015) proposed a ciphertext multi-sharing mechanism for privacy preservation. The details of clients who will be using the data are unknown to the data owner. A proxy re-encryption scheme is applied here. Jegadeesan et al. (2019) proposed a mutual authentication scheme for mobile cloud computing environment employing Bilinear Mappings and Hash functions. A distinct private key is used to get the services of different service providers. Although the Trusted Third Party (TTP) is required at the early stages, future authentication processes don't want TTP. Li et al. (2017a) proposed Distributed Authentication and Authorization Scheme (DAAS) to solve the problem of Authentication, Authorization and Auditing (AAA) in Bigdata. It also ensures Bigdata integrity, secure key exchange, and verification of user identity. In order to address the issue of latency Aazam et al. (2018) proposed to deploy Fog Computing between the IoT devices and the cloud. The authors Omoniwa et al. (2019) also proposed to introduce Fog computing. But, both the works has given least priority to security.

1.1.1 Existing studies to enhance Kerberos authentication protocol security

El-Emam et al. (2009) suggested modifying the database at the Key Distribution Centre (KDC) to protect from the dictionary attack. The private keys are generated using the SHA-256 hashing and 3-DES algorithm. A modified version of the Kerberos Authentication Protocol has been presented by Sutradhar et al. (2018). The authors proposed to rely on Threshold Cryptography concept to deploy multiple Ticket Granting Servers (TGSs) at the Keberos Server. Though the system is efficient in terms of cost, computational overhead and security, the response time is high compared to the traditional system. Wang and Feng (2013) applied Dynamic Passwords to prevent password guessing attacks and Diffie–Hellman Algorithm to exchange passwords. The security of Kerberos Key Distribution Server (KDC) and the threat of replay attacks are not dealt with in this work. Hu et al. (2012) put forward the use of the Diffie Hellman DSA mechanism to increase the performance of the Kerberos protocol. The signature generation and verification is performed only once in this system. This reduces the computational overhead compared to the existing version where it is done twice. Moreover, passwords needn't be remembered. Here, the problem of Single Point of Failure is not addressed. Rahul et al. (2014) projected an innovative authentication framework for Hadoop which is a combination of Public-key cryptography, Private Key Cryptography, Hashing, and Random number generation. The application of public-key cryptography makes the system slower by adding the computational overhead and hence degraded the system performance. Jeong et al. (2016) proposed a high dimensional authentication protocol grounded on hash chains. The problem with this method is the storage overhead due to long hash chains or reconstruction of hash chains every time leading to

computational overhead. Dou et al. (2018) put forward the idea to combines hardware and software security solutions. A Trusted Platform Module (TPM) that uses the RSA Encryption method is recommended here. The attack on the TPM is questioned here and no means are adopted to secure the Hadoop Name nodes assuming them to be secure.

1.1.2 Existing schemes to enhance security of Kerberized Hadoop clusters

Apache Hadoop, a universally adopted Big Data platform, depends on Kerberos Authentication Protocol for Mutual Authentication and Verification. Various security loopholes are well-known when Hadoop Clusters are Kerberized. Abidin et al. (2020) proposed a scheme in which multiple devices owned by the user conjoin to deliver authentication services. The approach grounds on the Threshold Secret Sharing mechanism that allows a user device to store only a part of the secret key or the signing key and a prefixed threshold number of parts are needed to rebuild the key to sign any message. Harn and Hsu (2017) proposed bivariate polynomial secret sharing to construct tokens. These tokens can be re-used for membership key establishment, authentication, and cryptographic applications. It doesn't require any extra key formation and authentication as the entire process is based on the same single token. Blockchain technology and secret sharing scheme has been combined in the work by Imine et al. (2018) to guarantee mutual authentication for Fog Computing Architecture edge devices. Once the fog nodes and users are registered at the cloud level, all further verification processes don't require cloud interference.

Jeong and tae Kim (2014) recommended a token-based authentication scheme for the Hadoop Distributed File System (HDFS) that uses Elliptic Curve Cryptography. Block Access Token and Hash Chain of Keys are used here. The complex algorithm makes the more chances of errors during implementation and hence degrades the security. Chattaraj et al. (2018) suggested an Authentication Protocol for Hadoop-based Big Data Platform to use the digital signature, Advanced Encryption Standard, and Elliptic Curve Cryptography. It relies on the digital signature, Advanced Encryption Standard, and Elliptic Curve Cryptography. Time constraints and computational complexity deters the acceptance of this scheme. Shakil et al. (2020) proposed to use biometric Signature and Resilient Back Propagation to safeguard healthcare data offloaded to the cloud. The cost of hardware and changeability of signature is the deterring factor for the adoption of this scheme.

1.1.3 Recent authentication systems based on elliptic curve cryptography

The authors in He and Wang (2015) proposed a three-factor authentication scheme based on Elliptic Curve Cryptography along with bio-metrics and smart-cards for a multi-server environment. The authors claim that the method provides anonymity, mutual authentication and withstands many known attacks in a multi-server environment. But, it is observed that the scheme achieves this level of security at increased computational and communicational cost. Wang et al. (2016) proposed a biometric-based authentication system for a multi-server environment. The scheme offers more security and functionalities compared to many existing schemes of the time. However, the use of smartcards and biometric sensors adds computational complexity. Moreover, users' privacy is compromised in this scheme as the user needs to share personal information with the application servers during the user registration process and authentication phase. Reddy et al. (2017) proposed a mutual authentication and key agreement protocol using passwords, smartcards, and biometrics. The flexibility of the scheme makes it suitable for a real-time environment. Unfortunately, the scheme enables user traceability and is prone to privileged insider attacks.

1.1.4 Blockchain-based authentication solutions

In Hammi et al. (2018) put forward a decentralized authentication mechanism based on blockchain technology. Certain trusted nodes called CPANs (Personal Area Network Coordinators) are permitted by the smart contracts to write the blockchain. When the device travels from one group to another, the CPAN in the new cluster gets its information from the blockchain which requests the home CPAN for the symmetric keys to connect with the device. A Blockchain-based adaptive authentication scheme for Big Data Hadoop Framework has been suggested by Kankal and Patil (2019). The scheme creates two APIs—a new HDFS/Client gateway and another API based on blockchain. User authentication data is stored on distributed blockchain Databases. Blockchain-based Approach to Enhance Big Data Authentication in Distributed Environment has been proposed Abdullah et al. (2017) to use SIN (Secure Identity Number) instead of session keys to secure the digital signatures. As seen, many methods have been proposed, none of them presented a clear idea. A summary of the above study is tabulated in Table 1.

RSA (Rivest–Shamir–Adleman) and Elliptic Curve ElGamal cryptosystem are two public key cryptosystems that are used in most modern systems. The authors Ertaul and Chavan (2007) performed a comparative study of RSA and Elliptic Curve ElGamal Threshold Cryptosystems for MANETs. Table 2 shows a comparative analysis of RSA and Elliptic Curve ElGamal Schemes.

Keeping in mind the above Literature Survey on various existing authentication mechanism in Hadoop/big data environment and to eradicate the security loopholes, this paper aims to propose an effective authentication protocol for Kerberos enabled Hadoop systems.

Table 1 Summary of Various Existing Authentication Schemes

Paper	Methodology	Pros	Cons	Implementation
Abidin et al. (2020)	Threshold Cryptography - Schnorr signature scheme	No single point of vulnerability	Method fails when the user device is stolen or hacked	Key share generation - mpcToolkit Schnorr Signatures- python implementation
Harn and Hsu (2017)	Threshold Cryptography - Bivariate polynomial secret sharing	No need for any extra key formation and authentication No single point of vulnerability	Additional computational overhead with members for key generation	Theoretical analysis only
Imine et al. (2018)	Blockchain technology and secret sharing	Dynamic & scalable Secure & fully distributed authentication Low latency Adaptive & portable Low overhead	RSA cryptosystem – storage overhead User revocation activities are not much dealt with	Real Wireless Ad-Hoc Network
Jegadeesan et al. (2019)	Bilinear Mappings and Hash functions	Highly time-synchronized Less computation	No efficient key management system for group communications Not dealt with confidentiality of communicating message	Cygwin 1.7.35–15 with the gcc version
Hammi et al. (2018)	Blockchain Elliptic Curve Digital Signature Algorithm	Less energy consumption Storage efficient Fast and robust	Is not very scalable Symmetric key cryptosystem—less secure	Real implementation with C and Ethereum Blockchain *TestRPC*
Kankal and Patil (2019)	Blockchain-based	No Single point of failure Distributed authentication	Demands alteration of prevailing Hadoop Infrastructure - expensive	Interface using existing Hadoop API (java/python) Blockchain—Solidity
Chattaraj et al. (2018)	Digital Signature, Advanced Encryption Standard (AES), Elliptic curve cryptography	Robust and secure	Time-consuming Complex computations	Real-Or-Random (ROR) model AVISPA tool
He and Wang (2015)	Elliptic Curve Cryptography, biometrics, smartcards	Robust and secure	Computational and communicational cost Infrastructure overhead	Burrows-Abadi-Needham (BAN) logic
Wang et al. (2016)	Elliptic Curve Cryptography, biometrics, smartcards	Lower computational cost	Computational complexity User privacy is compromised Infrastructure overhead	Informal Analysis
Reddy et al. (2017)	Elliptic Curve Cryptography, biometrics, smartcards	Robust Efficient	Uer traceability Prone to privileged insider attacks Infrastructure overhead	Burrows-Abadi-Needham (BAN) logic AVISPA

1.2 Contributions

This paper proposes an efficient authentication scheme for Kerberos Enabled Hadoop Clusters. The major scientific contributions of this paper are as follows:

(a) Dictionary attacks or password guessing attacks are prevented by the use of a digital signature mechanism.
(b) Single point of failure or Vulnerability is dealt with the deployment of the multiple Ticket Granting Servers (TGS) at the KDC and the availability of thus guarantee in case of any of the TGSs crashes or compromised.
(c) ElGamal Cryptosystem significantly reduces the memory and computational overhead compared to many existing systems that use traditional mechanisms like RSA.
(d) Blockchain technology promises distributed storage of credentials which is practically infeasible to tamper with.

Organization: The rest of this paper is structured as follows: A discussion on the Mathematical Intuitions and Preliminaries Concepts is presented in Sect. 2. In Sect. 3, the proposed scheme is discussed in detail. The implementation details provided in Sect. 4. Section 5 discusses the security analysis and Sect. 6 presents the performance analysis. Finally, the paper is concluded in with some hints for future works in Sect. 7.

1.3 Highlights of the Proposed System

The major benefits of the proposed scheme are:

(a) the multiple Ticket Granting Servers (TGS) at the KDC safeguards the system from a single point of failure and guarantees availability in the event of any of the TGSs fail
(b) the use of ElGamal Cryptosystem significantly reduces the memory and computational overhead compared to many existing systems that use traditional mechanisms like RSA
(c) the digital signature mechanism relieves the use of password authentication and hence addresses the threat of password guessing attacks
(d) the hottest trend of deploying blockchain technology promises distributed storage of credentials which is practically infeasible to tamper

2 Mathematical intuitions and preliminaries concepts

The (t,n)-threshold Shamir Secret sharing scheme by Shamir (1979) splits a secret into 'n' shares and distributes to 'n' participants, and at least 't' among them are required to contribute their shares to reconstruct the secret. Threshold ElGamal Scheme by Desmedt and Frankel (1989) also relies on the Shamir Secret Scheme.

2.1 Lagrange interpolation and secret sharing scheme

Shamir's Secret Sharing Scheme considers secret as a point in space and shares as points along the secret random curve. Let p be a prime and $f(x)$ be a polynomial of degree $t-1$ over$\mathbb{Z}p$. Then,

$$f(x) = \sum_{j=0}^{t-1} a_j x^j \bmod p \tag{1}$$

Given any set of points (x_i, y_i) where$y_i=f(x_i)$, one can reconstruct the polynomial $f(x)$ by LaGrange's Interpolation function as:

$$L_i(x)=\prod_{j\in,\ i\neq j} \frac{x-x_j}{x_i-x_j} \bmod p \tag{2}$$

where I be any subset of $\{1,n\}$ and its size is$t$. Also,

Table 2 Comparative Analysis of RSA and Elliptic Curve ElGamal Cryptosystem

Parameter	RSA	ElGamal
Key Size required (to achieve same security level)	High	Low
Foundation	Infeasibility in solving the factoring of big primes problem, (i.e., recovering p,q from n=pq)	Difficulty in solving Discrete Logarithmic Problem. It is a bit more complex but performs better.(i.e., recovering x from $h = g^x$)
Computations	Complex	Simpler
Encryption Time	Faster	Slower
Decryption time	Slower	Faster
Semantically secure	No	Yes (Ciphertext reveals only negligible information about plain text)

$$L_i(x) = \begin{cases} 1; x = x_i \\ 0; x = x_j; \forall j \in I; j \neq i \end{cases} \quad (3)$$

Thus, using the construction formulae,

$$f(x) = \sum_{i \in I} L_i(x).f(x_i) mod\ p \quad (4)$$

For the Shamir secret-sharing scheme , one need not reconstruct the whole polynomial. The concern is only on the constant term $f(0)$=a_0 which can be computed using a new set:

$$\Lambda_i = L_i(0) = \prod_{j \in I, i \neq j} \frac{x_j}{x_j - x_i} mod\ p \quad (5)$$

Then,

$$f(0) = \sum_{i \in I} \Lambda_i . f(x_i) \quad (6)$$

2.2 ElGamal cryptosystem over $\mathbb{Z}p^*$

It consists of the following algorithms:

Algorithm 1: GenKey - The Key Generation Algorithm

Step 1 : Choose a large prime p, such that the Discrete Logarithmic Problem over $\mathbb{Z}p$ is hard.
Step 2 : $\mathbb{Z}p^* = \mathbb{Z}p$ -{ 0 }is a cyclic group and randomly choose the generator α.
Step 3 : Then choose a secret, $d = a_0$; $0 \leq \alpha \leq p-2$.
Step 4: Compute $\beta \equiv a^{\alpha} mod\ p$
$pk := < p, \alpha, \beta >$ **is the public key and** $sk := < d >$ **is the private key.**

Algorithm 2: Encrypt - To Encrypt the message 'm'

Step 1 : Choose random $k \in \mathbb{Z}p$ and compute the ciphertext C as

$$C = Encrypt_p k(m,k) = < c1, c2 >$$

where $c1 = \alpha^k mod\ p$ and $c2 = m.\beta^k mod\ p$

Algorithm 3: Decrypt - To decrypt the Ciphertext 'C'

Step 1 : Compute the plain text P as:

$$P = Decrypt(c1, c2) = c2(c1^{d^{-1}}) mod\ p$$

where $c1 = \alpha^k mod\ p$ and $c2 = m.\beta^k mod\ p$

Algorithm 4: Sign: To sign the message m as follows

Step 1: Choose a random integer k relatively prime to $p-1$ from $2 \ldots p-2$
Step 2: Compute $r := \alpha^k mod\ p$
Step 3: Compute $s := H(m) - xr)k^{-1} mod\ p$
Step 4: If $s = 0$, then repeat steps 1-3 with different values of k .
Here (r,s) **is the signature.**

Algorithm 5: Verify: To verify the signature $(r.s)$

if $\alpha^{H(m)} \equiv y^r r^s mod\ p$ **then**
  the signature is valid. $(0 < r < p)$ and $(0 < s < p-1)$
else
  Signature is not valid

The public keys at the user and Hadoop Server, $pk_u := < p_u, \alpha_u, \beta_u >$; $pk_h := < p_h, \alpha_h, \beta_h >$ and the private keys $sk_u := < d_u >$;$sk_h := < d_h >$ are generated using Elliptic Curve ElGamal Cryptosystem. The Ticket Granting Server (TGS) applies threshold cryptography and hence the algorithms are slightly changed as in Algorithm 4.6.

Algorithm 6: Ticket-Granting-Server Key-Gen/Encrypt/Decrypt

KeyGen: Step 1 - 4: Same as computed for user and Hadoop server. Step 5: Compute a polynomial $f(x)$ of degree $t-1$ as :

$$f(x) = a + \sum_{i=1}^{t-1} a_i x^i \ mod\ p$$

Step 6: For each shareholder i , compute n shares of a as:

$$s_i = f(x_i)$$

Encrypt:
Step 7: Choose a random integer $k \in \mathbb{Z}p$ and compute ciphertext:

$$C = < c1, c2 > = < \alpha^k, m.\beta^k >$$

Decrypt:
To decrypt a ciphertext, the t participants must ask the Ticket Granting Server (TGS) for their decryption shares: Step 8: The Ticket Granting Server (TGS) computes the decryption shares for each participant as:

$$d_i = (c_1)^{s_i} = (\alpha^k)^{s_i} mod\ p = (\alpha^k)^{f(x_i)} mod\ p$$

Step 2: Let I be the set of t participants that requested for key shares. The participants after receiving their shares d_i collaborate to compute:

$$d \equiv \prod_{i \in I} d_i^{\Lambda_i} \equiv \prod_{i \in I} (\alpha^k)^{f(x_i)\Lambda_i} \equiv (\alpha^k)^{\sum_{i \in I} f(x_i)\Lambda_i} \equiv \alpha^{kf(0)} \equiv \alpha^{k^a} \left[Here, \lambda_i = L_i(0) = \prod_{j \in I, i \neq j} \frac{x_j}{x_j - x_i} mod\ p \right]$$

Then m is computed as:

$$m = c2d^{-1} \left[= m\beta^k \alpha^{-ka} = m\alpha^{ak} \alpha^{-ka} = m \right]$$

Public key, $pk_t := < p_t, \alpha_t, \beta_t >$ are distributed to all the communicating entities. The secret key share of each participant i, $sk_i = f(x_i)$ is shared among each participant using Shamir's Secret Sharing scheme.

2.3 Blockchain technology

Blockchain is an upcoming technology that has revolutionized the financial industries and crypto-currencies in recent years. The main attraction towards this technology is that there is no central entity to control the nodes in the blockchain network. A consensus is reached between the nodes as a whole, but individually no nodes trust each other. The transactions are immutable and it's difficult to fabricate any data stored in the nodes. The majority of the validation nodes called miners have to verify the transaction before making any changes to the existing blockchain. It involves solving a heavy computation problem as well (Christidis and Devetsikiotis 2016).

The blockchain is thus a distributed tamper-proof ledger that can be used to store information. Being decentralized, it doesn't store data at any single location. They claim to have several fascinating benefits (Sturges 2020)such as security, immutability, and lower price. The Blockchains are stored in computers within the system, also named nodes. Every node maintains a copy of the transactions and is always updated among the nodes after every new entry. Any outage at a single node will not have a devastating result as the other nodes in other locations will carry on the task with the legitimate copies available with them. The technical details of blockchain are beyond the scope of this paper.

3 Proposed system

The proposed system aims to shed off the threat of a single point of failure by amalgamating Elliptic Curve ElGamal Digital Signatures, and Threshold Cryptography with Blockchain technology. The existing authentication mechanism in Hadoop clusters is modified as follows:

(a) The Key Distribution Centre (KDC) authenticates the user at the initial phase using Digital Signatures instead of passwords.
(b) Blockchain network replaces the local database used in the Key Distribution Centre (KDC) which is a widely accepted tamper-proof distributed storage.
(c) A multi-authenticating system is established by deploying multiple Ticket Granting Servers (TGSs). These are modeled to work based on threshold cryptography such that at least t-out-of-n Ticket Granting Servers (TGS) should collaborate to get the decryption key. Each participant Ticket Granting Servers (TGS) holds their respective shares of the secret key to decrypt the Ticket Granting Ticket (TGT).

3.1 Design Goals

The proposed system is designed with the goals to eliminate the following threats in a Kerberized Hadoop Cluster:

(a) Single Point of Failure: The system should withstand even if the Key Distribution Center (KDC) is compromised or failed.
(b) Replay Attacks: The system should be able to secure the Hadoop Cluster from malicious or fraudulent repeat of any messages.
(c) Insider Attacks: The system should secure the Hadoop Cluster from any previously legitimate entity turning hostile later.
(d) Dictionary Attacks: The system should secure the Hadoop Cluster Password Guessing Attack or Brute Force Attack.

3.2 Framework

The overall workflow of the system is illustrated in Fig. 1. The proposed system works in three phases:

(a) Initialization Phase
(b) Registration Phase
(c) Authentication Phase

3.2.1 Initialization phase

This phase sets up the private-public key pairs of all the communicating entities. The proposed scheme deploys the Elliptic Curve ElGamal cryptosystem to generate the public keys. These are posted to the blockchain network as a transaction. Elliptic Curve ElGamal Threshold Cryptography is being used for the key generation and distribution process at the Ticket Granting Server.

3.2.2 Registration phase

The registration request from a user to the trusted third party, Key Distribution Centre (KDC) includes the user ID_u and pk_u. The user credentials are posted to the blockchain network to check if the ID_u– pk_u combination exists before. The entire blockchain nodes are updated accordingly with the new user information. The user is notified about the same if user information already exists.

3.2.3 Authentication phase

A digitally signed authentication request is sent by the user to the Authentication Server in the Key Distribution Centre (KDC). The Authentication Server forwards this request to the blockchain network and the blockchain miners check if the signature is valid and issues the Ticket Granting Ticket (TGT). The user request message includes user ID_u, ID_{tgs} pk_u and σ where:

$$\sigma = sig(sk_u, ID_u) \tag{7}$$

The request is posted to the blockchain as:

ID_u, ID_t, pk_u, σ

The following validation is done by the blockchain miners to check whether:

$$b = ver(ID_u, \sigma, pk_u) \tag{8}$$

such that

$$b = \begin{cases} 1, \text{ if } \sigma \text{ is a valid signature on } ID_u \text{ under the } sk_u \\ 0, \text{ otherwise} \end{cases} \tag{9}$$

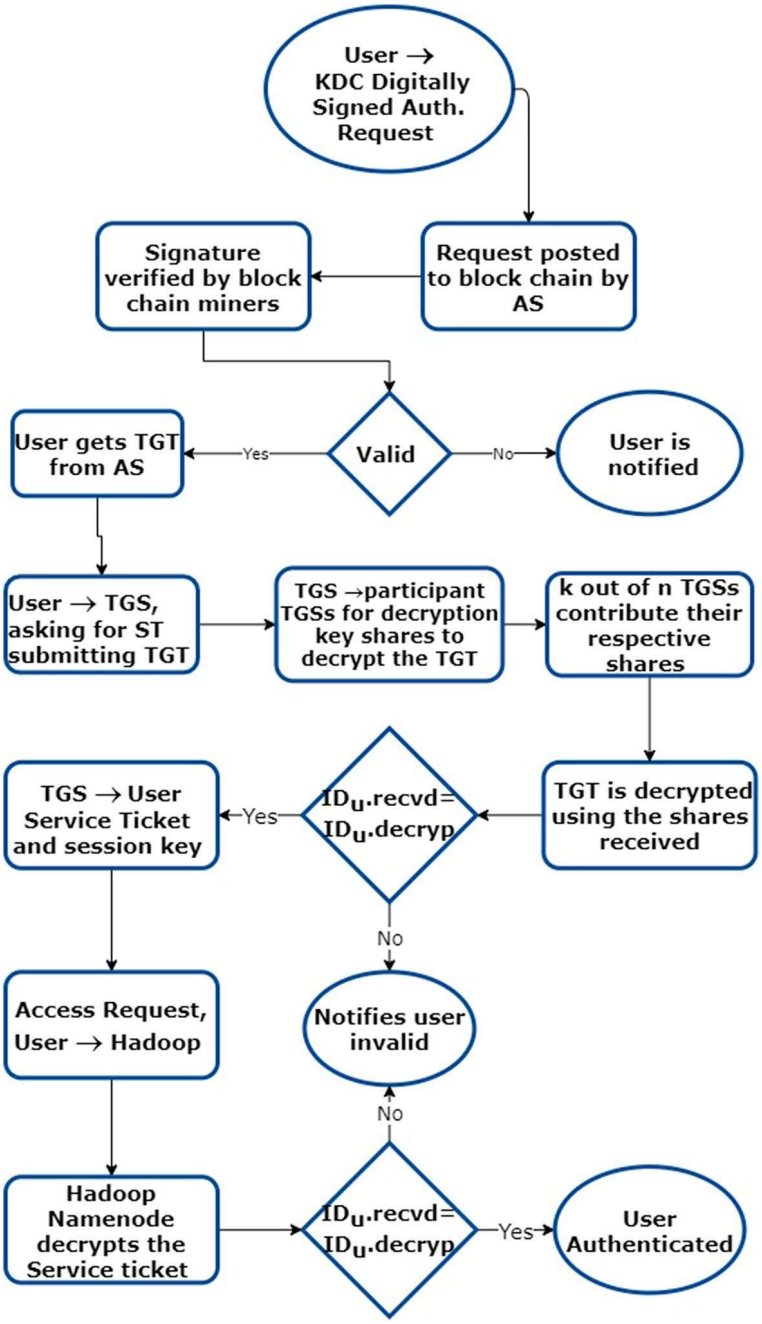

Fig. 1 Overall Workflow of the Proposed Framework

If $b = 1$, the user is granted a Ticket Granting Ticket (TGT) to contact the Ticket Granting Server (TGS). Figure 2 depicts the system architecture. The Ticket Granting Ticket (TGT) is encrypted with the secret key of the Ticket Granting Server (TGS) as:

$Encrypt(T_t) =< c1, c2 >= \alpha^k, T_t.\beta^k >$ where $k \in \mathbb{Z}p$ is a randomly chosen integer by the Ticket Granting Server (TGS). Here, multi-party authentication is enabled at the Ticket Granting Server (TGS) by dividing it into 'n' participants. A service ticket can be issued only if a prefixed threshold 't' number of shares of the secret key held by participant Ticket Granting Server (TGS) is presented. This also upturns the readiness of the Ticket Granting Server (TGS). Any attack or failure in the Ticket Granting Server (TGS) can lead to the stoppage of the entire authentication process in the traditional Kerberos scenario. The Ticket Granting Ticket (TGT) issued to successfully verified user is encrypted with the public key of Ticket Granting Server (TGS). When the user sends this Ticket Granting Ticket (TGT) to TGS for service access, the TGS has to decrypt it first before issuing any service ticket. This can be possible only if the TGS obtains a threshold number of TGS's secret key shares from the participant shareholders. These shares can be interpolated using LaGrange's Interpolation formula to get the decryption. The decryption shares are computed as:

$$d_i = (c1)^{sk_i} \tag{10}$$

The decryption key is computed using the shares as:

$$d = \prod_{i \in I} d_i^{\Lambda_i} \tag{11}$$

where Λ_i is computed using LaGrange's Construction Formula and I is the set of participants. Then, the Ticket Granting Ticket

$$T_t = c2d^{-1} \tag{12}$$

The client's authenticity is tested and the Ticket Granting Servers (TGS) grants the Hadoop Service Ticket (T_h) to the user.

3.3 Algorithm

Table 3 shows the various variables used and the proposed algorithm follows.

Algorithm 7: Distributed Authentication Algorithm for Hadoop

Step 1: **User→ KDC:** $\{ ID_u, ID_{tgs}, pk_u, \sigma \}$ —— (i)
Step 2: AS post (i) to the blockchain and the blockchain miners compute, $b = ver(ID_u, \sigma, pk_u)$
if $b==1$ **then**
  (a) The AS sends encrypted TGT to the client as: $T_t = E_{pk_t}(K_{tu}, ID_u)$
  (b) **AS→ User:** $\{E_{pk_u}(K_{tu}), T_t\}$ —— (ii)
  (c) Go to Step 3
else
  the signature is not valid and the user is notified: **AS→ User:** ***"Access Denied"*** —— (iii)
Step 3: The user decrypts (ii) using her private key sk_u and gets the session key to communicate with the TGS.
Step 4: **User→ TGS:** $\{ID_u, T_t, ID_h\}$ —— (iv)
Step 5:
(a) The Ticket Granting Server (TGS) decrypts T_t received in (iv). It gets k-out-of-n TGSs share of the decryption key, d_i
(b) k participants work together to compute the decryption key, d
(c) TGT is decrypted with the secret key d to obtain $T_t = K_{tu}||ID_u$
(d) **If:** ID_ureceived in (iv) is the same as that in decrypted T_t, **then** go to step 6.
(e) **Else: AS→ User:** ***"Access Denied"*** —— (v)
Step 6: TGS computes Service Ticket T_h as : $T_h = E_{pk_h}(K_{uh}, ID_u)$
Step 7: **TGS→ User :**$\{T_h, E_{pk_u}(K_{uh})\}$ —— (vi)
Step 8: At User: The user decrypts (vi) and obtains the session key, K_{uh} , to communicate with the Hadoop Server.
Step 9: **User→ H:** $\{ID_u, T_h, E_{k_{uh}}(seq\#)\}$ —— (vii)
Step 10: The Hadoop Server decrypts:
(a) the ticket T_h, using its private key sk_h and gets session key K_{uh}
  (i) **If:** ID_ureceived in (vi) is the same as that in decrypted T_h , **then** go to step 10b.
  (ii) **Else: H → User:** ***"Access Denied"***
(b) the $seq\#$ using the session key K_{uh} obtained in the above step (10a).
Step 11: **H → User:** $\{E_{k_{uh}}(seq\#+1)\}$ —— (vii)
Step 12: The user decrypts (vii) to detect the replays.
Step 13: End

4 Implementation details

The proposed method is implemented as a simulated environment using Riverbed Modeller (AE) Simulator Sethi and Hnatyshin (2013) and is illustrated in Fig. 3. The network topology consists of User Workstations, a Key Distribution Center (KDC) with an Authentication Server, and multiple Ticket Granting Servers. Here, the ppp_wkstn_adv node object is deployed as a user workstation and is assumed as the initiator of all the communications. ppp_server_adv node object is deployed as the Authentication Server and for Namenode. The Internet cloud discards 0.0 of arriving traffic and augments 100-millisecond delay to the packets. The authentication request message size is assumed to be 2 KB and the user spends 2 s to digitally sign and initialize this message. The Blockchain network takes 0.5 s to verify the digital signature. The tickets generated in this model are assumed to be of size 1 KB and the ticket encryption process takes 5 s. The size of encrypted messages between the user and the Hadoop cluster is uniformly distributed between 1 KB and 10 KB.

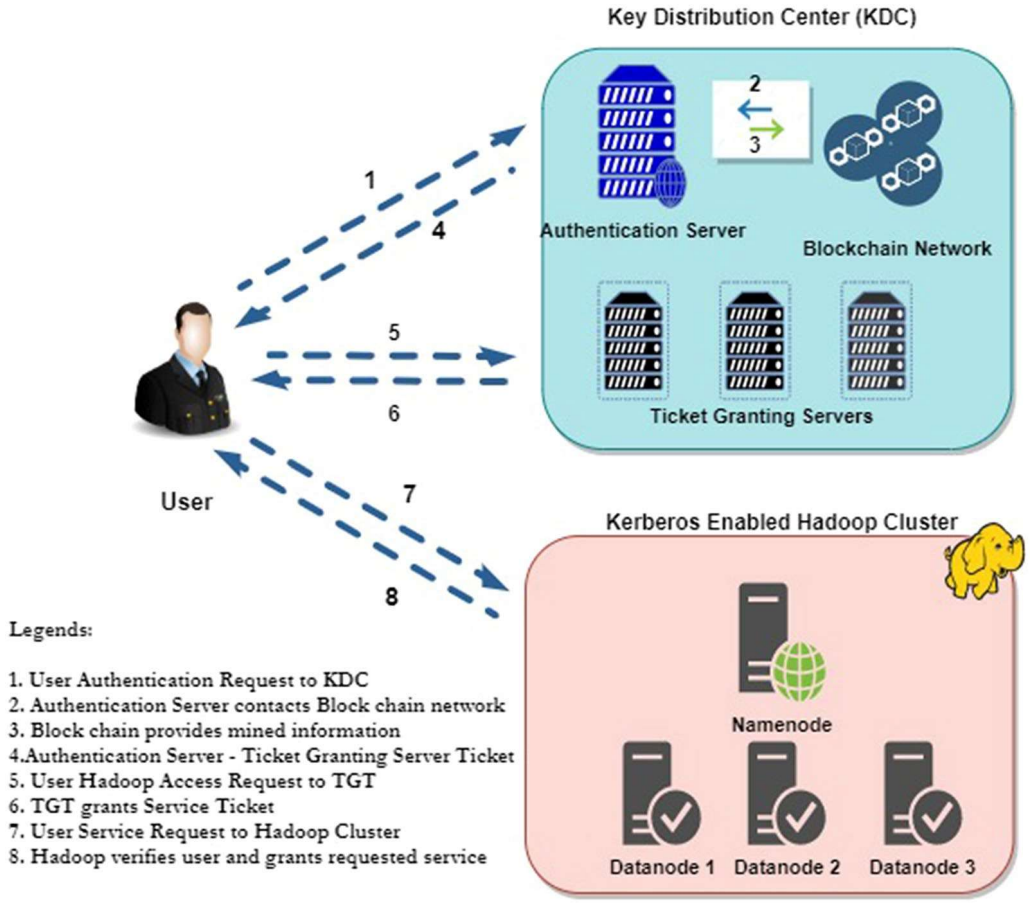

Fig. 2 System Architecture of the Proposed System

Table 3 Symbols and Variables used

Symbols	Definitions	Symbol	Definitions
pk_x	Public key of entity x	AS	Authenticating Server
sk_x	Private Key of entity x	TGS	Ticket Granting Server
ID_x	Identity of entity x	u	User
T_t	Ticket Granting Server Ticket	h	Hadoop Server (master)
T_h	Hadoop Service Ticket	t	Ticket Granting Server
σ	Digital Signature of user	$seq\#$	A Random sequence number
$E(.)$	Encryption function	K_{xy}	Session Key Shared between entities x and y

In the proposed system scenario, the user at workstation1 sends a digitally signed authentication request to the authentication server. The authentication server verifies the signature by contacting the blockchain network. Upon successful verification of the Initial Login Task, the user is notified and issued a Ticket Granting Ticket. Then, the user sends a request for Service Ticket. The multiple Ticket Granting Server deployed at the Key Distribution Center collaborates to decrypt the Ticket Granting Ticket. The user is then issued a service ticket and hence access to the secured Hadoop server. Consider the following scenario:

- Let a client Alice wants to get access to a file stored in the Hadoop Distributed File System (HDFS).
- Alice sends a message to the Key Distribution Center (KDC) requesting a session key for her communication with the Hadoop Cluster. This message contains Alice's ID, the ID of Ticket Granting Server (TGS), the public key of Alice, and her digital signature. It takes 2 s time

for Alice to generate this digitally signed message. The message is of size 2 KB.
- KDC forwards this request to the blockchain network. The miners in the blockchain network evaluates the authenticity of this message by verifying the digital signature against the public key. Blockchain network takes 0.5 s to process this request and generate a response of size 0.5 KB to the KDC indicating the user is valid or not.
- KDC generates a session key and Ticket for Alice to communicate with the Ticket Granting Server. The ticket contains the user ID and session key for Alice to communicate with the Ticket Granting Server and is encrypted with the public key of the Ticket Granting Server. Only those who know the private key of the Ticket Granting Server can decrypt it. The session key is encrypted using Alice's public key and is sent to Alice along with the ticket. This entire process takes 5.2 sec and the message generated here is of size 1 KB.
- Alice decrypts this message with her private key and gets the session key for her to communicate with the Ticket Granting Server and the ticket to access the Ticket Granting Server. Even though she gets the ticket in hand, she doesn't know what is its content. Alice forwards this ticket along with her ID and the ID of the Hadoop Cluster where the HDFS she wants to access resides to the TGS. This entire process takes 0.4 sec and the message generated here is of size 4 KB.
- The TGS contacts the participating TGS(s) to contribute their respective shares of the private key to decrypt the Ticket encrypted with the public key of TGS. When it receives a threshold number of shares (here, it is prefixed as 3), the ticket is decrypted and it gets Alice's ID and session key to communicate with Alice. This ensures that a single point of failure at TGS doesn't occur in the proposed system as if any of the participating TGS(s) fails, the shares from other threshold numbers of TGS(s) is sufficient for the decryption process. This entire process takes 5 s and the message generated here is of size 1 KB.
- The TGS verifies Alice's ID it got when decrypted the ticket and Alice's ID in the message it received from Alice is the same. If Alice's authenticity is confirmed, TGS generates a session key and a ticket (service ticket) for Alice to communicate with the Hadoop Distributed File System. This ticket contains the session key and Alice's ID and is encrypted using the public key of the Hadoop Server. The session key is encrypted using Alice's public key and is sent to Alice along with the ticket. This entire process takes 1 sec and the message generated here is of size 1 KB.
- Alice decrypts the message to get the session key for her to communicate with the Hadoop Distributed File System. She sends an authentication request to the Hadoop Server providing her ID, the service ticket, and a random sequence number encrypted using her session key to communicate with the Hadoop File System. This entire process takes 1.4 s and the message generated here is of size 4 KB.
- The Master node in the Hadoop Cluster decrypts the Service Ticket with its private key and gets Alice's ID and their session key. If Alice's ID it got when decrypted is the same as it received in the request message, Alice is considered as an authenticated user. The Hadoop server then increments the value of the sequence number and increments it with its session key to communicate with Alice and send back to Alice. This entire process takes 0.2 sec and the message generated here is of size 1 KB.
- Alice decrypts the message received and verifies the sequence number. If it's one greater than what she sent to Hadoop Server, she confirms the authenticity of the Hadoop Server as well. Hence, mutual authentication is also established. This entire process takes 0.2 s and no message is generated here.
- Alice can now directly communicate with the Hadoop Server using the session key until it expires.

Table 4 shows the details of the various tasks and phase configuration of the proposed system and the existing system in the simulated environment. A comparison of the results proves the strength of the claims of this paper.

The visual summary of the proposed scheme implementation is shown in Fig. 4. The user first sends digitally signed authentication request to the Authentication Server in the Key Distribution Center(KDC). This includes user's ID, the ID of Ticket Granting Server (TGS), the public key of user, and her digital signature $ID_u, ID_{tgs}, pk_u, \sigma$. This is posted to blockchain for verification. Blockchian miners verify b=ver(ID_u, σ, pk_u) and gives the result 0 or 1 based on the the validity of signature. If valid, a ticket granting server ticket $E_{pk_t}(K_{tu}, ID_u)$ is generated by the AS to share the session key for the TGS to communicate with the user. This ticket along with the session key encrypted using the user's public key is sent to user.

User requests for a Service Ticket to access the Hadoop Server by furnishing the TGT it received and this communication is encrypted using the session key it received $\{ID_u, T_t, ID_h\}$. The TGS obtains the decryption key shares from participating TGSs and decrypt the TGT. The TGS then verifies the user ID and if valid, the user is issued with the Service Ticket ST. With this, user sends access request to Hadoop Server. The Hadoop Server decrypts the Service ticket to obtain the session key for its communication with the user. Also, it verifies the user ID. If valid, it increments the seq# by one for mutual authentication and hence establish secure communication with the user.

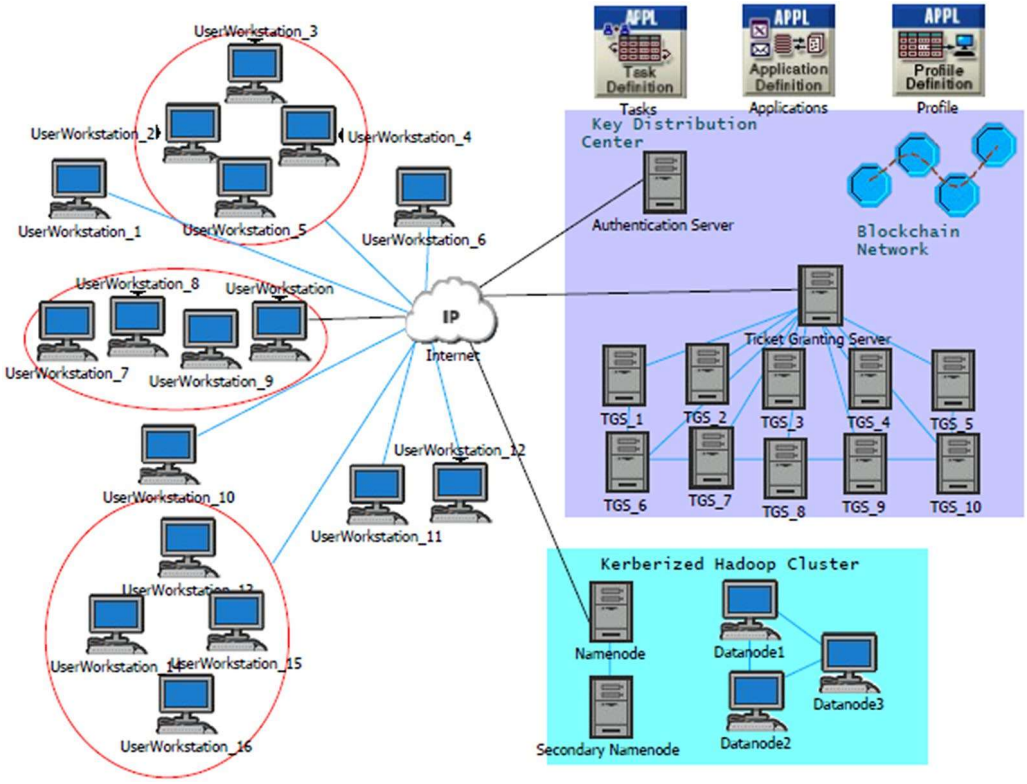

Fig. 3 Network Topology of proposed Kerberized Hadoop Cluster Simulation Model

5 Security analysis

The proposed scheme is resilient against many types of attacks in the existing Kerberos enabled Hadoop Cluster model. The justifications for the same are as follows:

Statement 1: The system is resilient to Replay Attacks. Justification: An adversary A cannot replay the messages at any phase of the system authentication process as the sender always chose a random key called Ephemeral key that changes each time for encryption of the same data at different times. Most of the existing systems use timestamps for the purpose which can result in time synchronization issues. It is observed that there is only a probability of $\frac{1}{(p-1)}$ for collision where p is a large prime chosen such that discrete logarithmic problem over $\mathbb{Z}p$ is hard.

Statement 2: The proposed system is resistant to Dictionary Attack Justification: In the proposed scheme, an adversary A will be able to guess the passwords and log in to the system. The scheme is independent of any stored password and relies on digital signatures and public keys stored in the tamper-free distributed blockchain network. The usage of the ephemeral key for the encryption process prevents the adversary A from determining any patterns in messages and its timings by matching the cipher texts.

Statement 3: There is no Single Point of Failure or attacks in the proposed system. Justification: The system is free from denial of service attacks as there isn't any single point of failure in the scheme. A distributed tamper-proof blockchain network is used as storage at the Key Distribution Centre and the contributions from multiple number of Ticket Granting Servers are required for the authentication process to proceed further. This ensures that the process is not denied at any phase due to any failures or compromised attacks.

Statement 4: The system defends Insider Attacks. Justification: The adversary $\mathcal{A}$ cannot even be a previously legitimate entity. The threshold shares contributed by $t-1$ participant Ticket Granting Systems ensures that at least $t-1$ shares are required for Ticket decryption and anyone who contributes wrong input doesn't make any sense. Hence insider attack is also defended.

Table 5 exhibits a comparative analysis of security features offered by the proposed system with some of the related schemes He and Wang (2015), Wang et al. (2016),and Reddy et al. (2017) . As seen from the analysis, the aforesaid schemes lack many of the required security schemes in a Bigdata environment and the proposed scheme in this paper fulfills all the security needs.

Table 4 Various Task and Phase Configuration of Simulated Environment

Task Name	Phase Name	Start After	Source	Destination	Request	Response
1. Login Initial to get TGT (Elgamal TC - Proposed)	1. Client Request to Authentication Server AS with Digital Signature	App Starts	User Workstation	AS	Init. Time = 2 Sec #Requests=1 Req. Size = 2 Kb	N/A
	2. Authentication Server Contacts Blockchain	Previous Phase Ends	AS	Blockchain	Init. Time = 0 Sec #Requests=1 Req. Size = 2 Kb	Processing Time = 0.5 Sec #Response = 1 Response Size = 0.5 Kb
	3. Signature Verification and User Notify	Previous Phase Ends	AS	User Workstation	Init. Time = 0.2 Sec #Requests= 0	Processing Time = 5 Sec #Response = 1 Response Size = 1 Kb
2. Request to Get ST (Elgamal TC – Proposed	1. Client Request to TGS with TGT	App Starts	User Workstation	TGS	Init. Time= 0.4Sec #Requests=1 Req. Size=4 Kb	N/A
	2. TGS asks shares from Participant TGSs	Previous Phase Ends	TGS	Participant TGSs	Init. Time = 0 Sec #Requests=1 Req. Size = 4 Kb	Processing Time = 5 Sec # Response = 3 Response Size = 1 Kb
	TGS issues Service Ticket to Client	Previous Phase Ends	TGS	User Workstation	Init. Time = 1 Sec #Requests=1 Req. Size = 1 Kb	N/A
3.Request for Service (Common)	1. Client Request to Hadoop Namenode with ST	App Starts	User Workstation	Hadoop Namenode	Init. Time=0.4 Sec #Requests=1 Req. Size = 4 Kb	N/A
	2. Verifies Service Ticket	Previous Phase Ends	Hadoop Namenode	None	Init. Time=0.2 Sec #Requests= 0	Processing Time =1 Sec # Response = 1 Response Size = 1 Kb
	3. Verifies Hadoop Namenode	Previous Phase Ends	User Workstation	None	Init. Time=0.2 Sec #Requests= 0	
4. Service access (Common)	1. Client to Hadoop Namenode	App Starts	User Workstation	Hadoop Namenode	Init. Time=0.2 Sec #Requests=1000 Inter-request Time = 1.2 Sec	N/A
	2. Hadoop Namenode to Client	App Starts	Hadoop Namenode	User Workstation	Init. Time=0.2 Sec #Requests=1000 Inter-Request Time = 1.2 Sec	N/A
5. Login Initial with Password to get TGT (RSA - Traditional)	1. Client Request to AS with Password	App Starts	User Workstation	AS	Init. Time = 0 Sec #Requests=1 Req. Size = 2 Kb	Processing Time = 10 sec #Response=1 Resp. Size = 5 Kb
6. Request For Service Ticket (RSA – Traditional)	1. Client Request to TGS with TGT	App Starts	User Workstation	TGS	Init. Time = 0.4 Sec #Requests=1 Req. Size = 4 Kb	Processing Time = 10 sec #Response=1 Resp. Size = 5 Kb

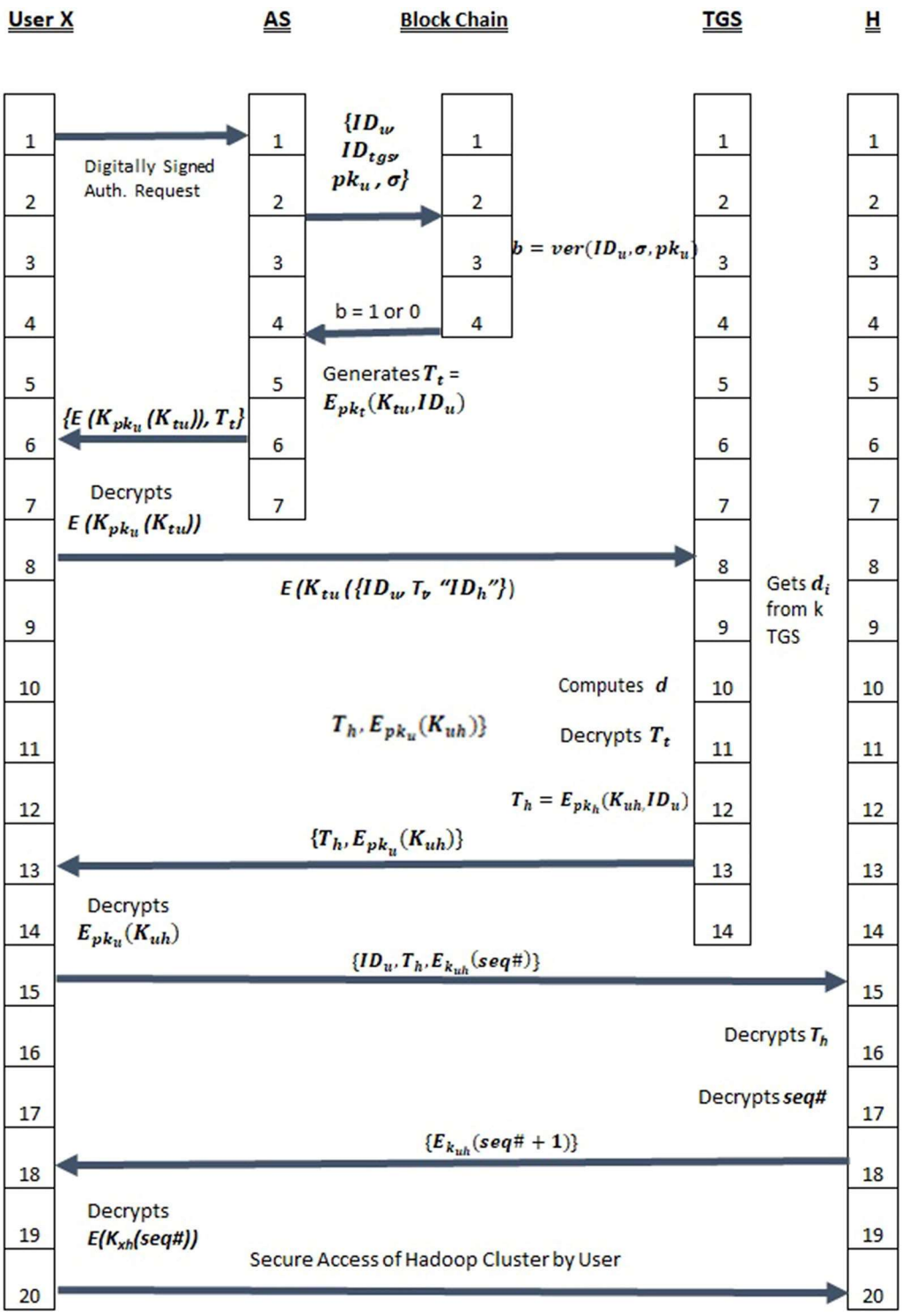

Fig. 4 Visual Summary of the Proposed Algorithm

6 Performance evaluation

To evaluate the performance of the proposed model that employs the Elliptic curve ElGamal Threshold Cryptosystem, a comparative study of the existing model that uses RSA (Rivest–Shamir–Adleman) cryptosystem is conducted in this experiment.

The experimental evaluation proves that the time taken for all phases in various tasks to complete is significantly lesser in the proposed system than in the existing system.

For example, Table 6 illustrates the response time for Login Task in the compared systems. As seen, the existing system doesn't start to respond even at 360 s (later at 390 sec, it responds. Not shown in table) whereas the proposed system started the same at 108 s. This is true with other tasks as well. Hence, the proposed system can be considered as more time-efficient when compared with the RSA based existing system.

Figure 5 depicts the response time for the Initial Login Task in the existing system with RSA encryption of ticket that uses the password to log in and proposed system with Elliptic Curve ElGamal System that uses the digital signature. In 1 h execution of the simulation, the proposed system is found to respond faster than the existing system. This is because of complex computation and large key size used in RSA.

Table 7 shows a comparative analysis of the RSA-based System with a password and the ElGamal-TC-based system with a Digital Signature. It is seen that ElGamal based systems take more time for encryption and lesser for decryption. This makes it suitable for Kerberos based Authentication Systems where the session keys are to be encrypted. As the decryption is faster, it suits well for time constraints big data analytics. Also, the increase in key size enhances the capability of defense for the ElGamal cryptosystem. Unlike RSA, the increase in time is gradual which is exponential in Existing systems that use RSA and other schemes. Also, the Elliptic curve ElGamal Threshold Cryptosystem provides equivalent security as of RSA schemes at much smaller key sizes. Hence, it is very clear and evident that the proposed system consumes lesser time to compare to the existing systems.

The packet network delay is the network delay experienced by response and request packets. Table 8 shows network delay at various instants during the entire simulation in both scenarios and the graphical representation of the same for the proposed and existing system is depicted in Fig. 6.

The proposed system has significantly reduced the network delay from 0. 496 sec in the existing system to 0.163 s. This again proves the efficiency of the proposed model concerning performance considering a real-time requirement in a scenario like big data Hadoop environment. For time constraint real-time analytics, there shouldn't be any delay or minimal delay (if any).

The rate at which traffic is received by all nodes in the simulated scenarios at certain instants is illustrated in Table 9. The proposed system generates lesser traffic than the existing system. For instance, at 540 s, the proposed system with ElGamal Cryptography generates traffic at the rate of 16 bytes/sec whereas the existing system with RSA generates 37 bytes/sec. The average traffic arrival rate at all nodes in the simulated environment. The same is depicted in Fig. 7.

The proposed system generates lesser traffic than the existing system which can improve the overall performance of the system. This enables the saving of bandwidth in real-time situations.

The computational complexity of the proposed scheme is compared with some of the related works by authors in He and Wang (2015), Wang et al. (2016), and Reddy et al. (2017) as in Table 10. The computations in all these systems involve elliptic scalar multiplication and hashing. As per the study, the total computational time costs for initial login task are approximately 16.961 ms, 0.105 ms and 8.505 ms for the schemes He and Wang (2015), Wang et al. (2016) and Reddy et al. (2017) respectively. At the same time, for the proposed system, it is 9.136 ms.

Table 6 Comparative Analysis of Security features of various Existing Systems with the Proposed System

Time (sec)	RSA—Existing system	ElGamal TC–Proposed system
0	0	0
36	0	0
108	0	4.873816842
144	0	4.873816842
180	0	4.873816842
216	0	4.873816842
252	0	4.873816842
288	0	4.873816842
324	0	4.873816842
360	0	4.873816842

Table 5 Comparative Analysis of Security features of various Existing Systems with the Proposed System

Features	He and Wang (2015)	Wang et al. (2016)	Reddy et al. (2017)	Proposed System
No Single point of failure	×	×	×	✓
Resist the privileged insider attack	×	×	×	✓
Resist dictionary attack	×	×	×	✓
Resist reply attack	×	×	✓	✓

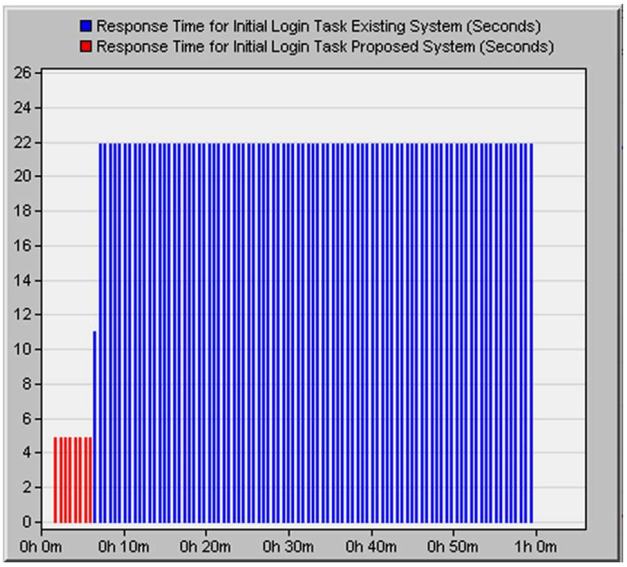

Fig. 5 Response Time for Initial Login Task

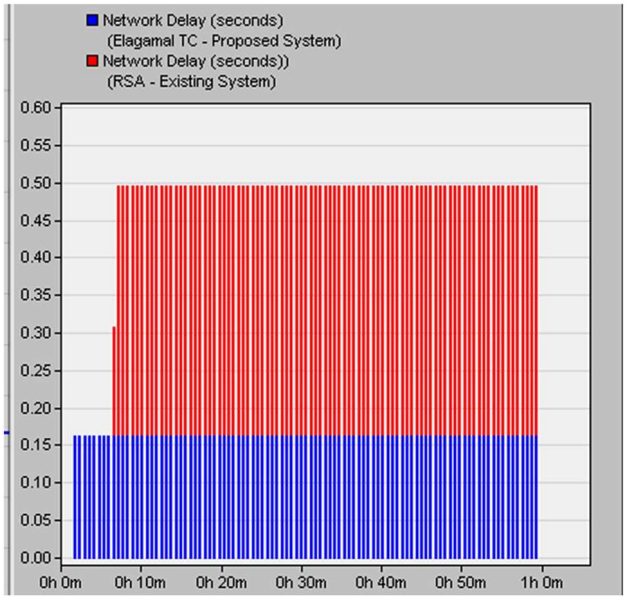

Fig. 6 Network Delay for request and response messages in proposed and existing system

Table 7 Comparative Analysis—RSA based System with password Vs ElGamal-TC based system with Digital Signature

Time (sec)	RSA—Existing system	ElGamal TC—Proposed system
Key generation	1 s	1 s
Encryption	0.003 s	1.2 s
Decryption	0.032 s	0.02 s

Table 8 Network Delay—RSA based System with password Vs ElGamal-TC based system with Digital Signature

Time (sec)	RSA—Existing system	ElGamal TC—Proposed system
0	0	0
36	0	0
72	0	0
108	0.162746639	0
144	0.162746639	0
180	0.162746639	0
216	0.162746639	0
252	0.162746639	0
288	0.162746639	0
324	0.162746639	0
360	0.162746639	0
396	0.162746639	0.30731367
432	0.162746639	0.496298959
468	0.162746639	0.496298959
504	0.162746639	0.496298959

Table 9 Traffic Rate at all nodes—RSA based System with password Vs ElGamal-TC based system with Digital Signature

Time (sec)	RSA—Existing system	ElGamal TC—Proposed system
468	18.28571429	42.66666667
504	17.06666667	39.82222222
540	16	37.33333333
576	15.05882353	35.1372549
612	14.22222222	33.18518519
648	13.47368421	31.43859649
684	12.8	29.86666667
720	12.19047619	28.44444444
756	11.63636364	27.15151515
792	11.13043478	25.97101449
828	10.66666667	24.88888889
864	10.24	23.89333333
900	9.846153846	22.97435897
936	9.481481481	22.12345679
972	9.142857143	21.33333333
1008	8.827586207	20.59770115
1044	8.533333333	19.91111111

Even though the proposed system costs more than schemes in Wang et al. (2016) and Reddy et al. (2017), it offers a significant level of security when compared to these systems as in Table 5. Hence, the proposed is a highly efficient and secure authentication system for Bigdata environment like Hadoop Clusters.

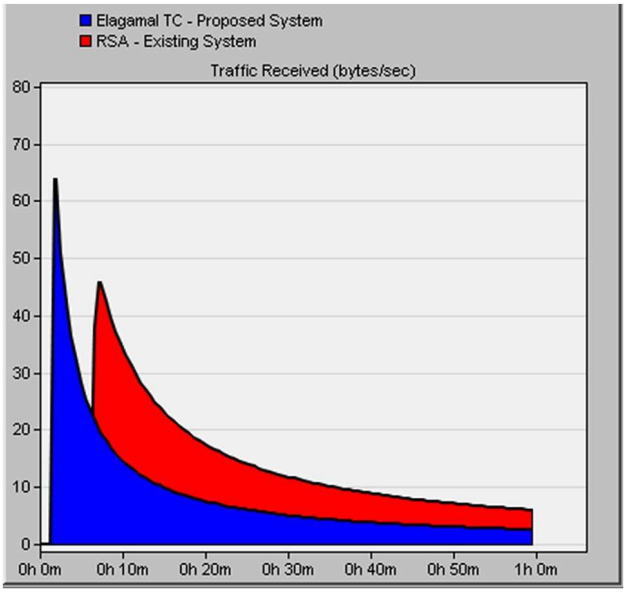

Fig. 7 Traffic Received Rate by all nodes

Table 10 Comparative Analysis of Computational time Cost

Entity	At User (ms)	At Auth. System (ms)	At Secured System (ms)	Total (ms)
He and Wang (2015)	6.349	6.342	4.27	16.961
Wang et al. (2016)	0.056	0.042	Null	0.105
Reddy et al. (2017)	4.263	4.242	Null	8.505
Proposed system	3.0	12.2	Null	9.136

7 Conclusion and future works

Apache Hadoop depends on third party security providers like Kerberos Protocol which suffers from Single point of failure and insider attacks for satisfying the security requirements. This paper put forward an effective authentication system for Kerberos enabled Hadoop clusters. The improved protocol relies on the power of most hot technologies like blockchain technology and Threshold ElGamal cryptosystems. The local database at the Key Distribution Center (KDC) is changed to a Blockchain network that stops the adversaries from pilfering the user's relevant information. The digitally signed access request deters the dictionary attacks. ElGamal and Threshold cryptosystems reduces the processing and storage overhead when compared to other existing systems that use traditional methods like RSA. Furthermore, the arrangement of multiple Ticket Granting Server (TGS) at the Key Distribution Center (KDC) guarantees the system availability and avoids the problem of a Single Point of Failure to a great extent. Thus, the proposed scheme is resilient against many types of attacks including Single Point of Failure, privileged Insider Attack, Dictionary Attack and Replay Attack compared to many existing Kerberos enabled Hadoop Cluster models.

The feasibility of the proposed scheme is studied using Riverbed Modeler (AE) simulation platform. The results prove that the proposed method is efficient in terms of communication and computational overheads. The network delay is comparatively very less in the proposed system than the existing system. The study proved that the traffic rate and response time for the proposed system is much better than RSA based existing systems. Even though the proposed system costs more than some of the existing schemes, it offers a significant level of security when compared to these systems. In nutshell, the scheme ensures the protection of Hadoop Systems from the Single Point of Failure problem along with additional security compared to most of the existing systems. In the future, experiments will be conducted in real-world settings and results will be studied to prove the efficiency and strength of the proposed method. Also, time synchronization problems of Kerberos enabled Hadoop Clusters will be addressed in our future work.

Funding No funds, grants, or other support was received.

Data availability Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

Declarations

Conflict of interest: The authors have no conflicts of interest to declare that are relevant to the content of this article.

References

Aazam M, Zeadally S, Harras KA (2018) Deploying fog computing in industrial internet of things and industry 4.0. IEEE Trans Ind Inform, 14: 4674–4682

Abdullah N, Hakansson A, Moradian E (2017) Blockchain based approach to enhance big data authentication in distributed environment. In: 2017 Ninth international conference on ubiquitous and future networks (ICUFN), pages 887–892, 2017. 10.1109/ICUFN.2017.7993927

Abidin Aysajan, Aly Abdelrahaman, Mustafa Mustafa A (2020) Collaborative authentication using threshold cryptography. Lecture Notes Comput Sci Emerging Technol Authorization Authentication 11967:122–137. https://doi.org/10.1007/978-3-030-39749-4_8

Adarshpal S (2013) Sethi and Vasil Y Hnatyshin. CRC Press, Practical opnet user guide for computer network simulation

Adi Shamir. How to share a secret. *Commun. ACM*, 22 (11): 612–613, November 1979. ISSN 0001-0782. https://doi.org/10.1145/359168.359176. URL https://doi.org/10.1145/359168.359176

Cathy Sturges. How can blockchain improve data storage?, Jan 2020. URL https://cointelegraph.com/news/how-can-blockchain-improve-data-storage

Chao Wu and Yike Guo. Enhanced user data privacy with pay-by-data model. In *2013 IEEE International Conference on Big Data*, pages 53–57, 2013. 10.1109/BigData.2013.6691688

Chattaraj D, Sarma M, Kumar Das A, Kumar N, Rodrigues JJPC, Park Y (2018) Heap: An efficient and fault-tolerant authentication and key exchange protocol for hadoop-assisted big data platform. IEEE Access, 6: 75342–75382 10.1109/ACCESS.2018.2883105

Chengqi Wang, Xiao Zhang, and Zhiming Zheng. Cryptanalysis and improvement of a biometric-based multi-server authentication and key agreement scheme. *Plos One*, 11 (2), 2016. 10.1371/journal.pone.0149173

Christidis Konstantinos, Devetsikiotis Michael (2016) Blockchains and smart contracts for the internet of things. IEEE Access 4:2292–2303. https://doi.org/10.1109/ACCESS.2016.2566339

Chundong Wang and Chaoran Feng. Security analysis and improvement for kerberos based on dynamic password and diffie-hellman algorithm. In *2013 Fourth International Conference on Emerging Intelligent Data and Web Technologies*, pages 256–260, 2013. 10.1109/EIDWT.2013.49

Desmedt Y, Frankel Y (1989) Threshold cryptosystems. In: Advances in cryptology—CRYPTO '89, 9th annual international cryptology conference, Santa Barbara, California, USA, August 20-24, 1989, Proceedings, volume 435 of *Lecture Notes in Computer Science*, pages 307–315. Springer, 1989. 10.1007/0-387-34805-0_28

Dong Xinhua, Li Ruixuan, He Heng, Zhou Wanwan, Xue Zhengyuan, Hao Wu (2015) Secure sensitive data sharing on a big data platform. Tsinghua Sci Technol 20(1):72–80. https://doi.org/10.1109/TST.2015.7040516

Dou Zuochao, Khalil Issa, Khreishah Abdallah, Al-Fuqaha Ala (2018) Robust insider attacks countermeasure for hadoop: design and implementation. IEEE Syst J 12(2):1874–1885. https://doi.org/10.1109/JSYST.2017.2669908

El-Emam E, Koutb M, Kelash H, Farag Allah OF (2009) An optimized kerberos authentication protocol. In: 2009 International Conference on Computer Engineering & Systems, pages 508–513

Ertaul L, Chavan N (2007) Rsa and elliptic curve- elgamal threshold cryptography (ecceg-tc) implementations for secure data forwarding in manets. In: Proceedings of the 2007 international conference on security and management, SAM'07, pages 142–146, 01

Gharib H, Belloulata K (2014) Authentication architecture using threshold cryptography in kerberos for mobile ad hoc networks. Adv Sci Technol Res J, 8 (22): 12–18, ISSN 2080-4075. https://doi.org/10.12913/22998624.1105141. URL https://doi.org/10.12913/22998624.1105141

GireeshKumar R, Dash S, Panigrahi. *A Novel Authentication Framework for Hadoop*, volume 324, page 333–340. Springer, 2014

Hammi MT, Bellot P, Serhrouchni A (2018) Bctrust: A decentralized authentication blockchain-based mechanism. In *2018 IEEE wireless communications and networking conference (WCNC)*, pages 1–6, 10.1109/WCNC.2018.8376948

He Debiao, Wang Ding (2015) Robust biometrics-based authentication scheme for multiserver environment. IEEE Syst J 9(3):816–823. https://doi.org/10.1109/JSYST.2014.2301517

Imine Y, Kouicem DE, Bouabdallah A, Ahmed L (2018) Masfog: an efficient mutual authentication scheme for fog computing architecture. In *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, pages 608–613, . 10.1109/TrustCom/BigDataSE.2018.00091

Jegadeesan S, Azees M, Malarvizhi Kumar P, Manogaran G, Chilamkurti N, Varatharajan R, Hsu C-H (2019) An efficient anonymous mutual authentication technique for providing secure communication in mobile cloud computing for smart city applications. Sustain Cities Soc, 49: 101522, 2019. ISSN 2210-6707. https://doi.org/10.1016/j.scs.2019.101522. URL https://www.sciencedirect.com/science/article/pii/S2210670718301720

Jeong Y-S, Shin S-S, Han K-H (2016) High-dimentional data authentication protocol based on hash chain for hadoop systems. *Cluster Computing*, 19 (1): 475–484, March 2016. ISSN 1386-7857. https://doi.org/10.1007/s10586-015-0508-y. URL https://doi.org/10.1007/s10586-015-0508-y

Kankal M (2019) Prof.Pramod Patil. An adaptive authentication based on blockchain for bigdata hadoop framework. Int J Eng Tech 5: 89–94

Kashish A. Shakil, Farhana J. Zareen, Alam M, Jabin S (2020) Bamhealthcloud: A biometric authentication and data management system for healthcare data in cloud. J King Saud Univ - Comput Inform Sci, 32 (1): 57–64, . ISSN 1319-1578. https://doi.org/10.1016/j.jksuci.2017.07.001. URL https://www.sciencedirect.com/science/article/pii/S1319157817301143

Lein Harn and Ching-Fang Hsu. A novel threshold cryptography with membership authentication and key establishment. Wirel. Pers. Commun., 97 (3): 3943–3950, December 2017. ISSN 0929-6212. https://doi.org/10.1007/s11277-017-4708-z. URL https://doi.org/10.1007/s11277-017-4708-z

Liang K, Susilo W, Liu JK (2015) Privacy-preserving ciphertext multi-sharing control for big data storage. IEEE Trans Inform Forensics Security 10(8):1578–1589. https://doi.org/10.1109/TIFS.2015.2419186

Li R, Asaeda H, Li J, Fu X (2017a) A distributed authentication and authorization scheme for in-network big data sharing. *Digital Communications and Networks*, 3 (4): 226–235, 2017a. ISSN 2352-8648. https://doi.org/10.1016/j.dcan.2017.06.001. URL https://www.sciencedirect.com/science/article/pii/S2352864817300676. Big Data Security and Privacy

Li Y, Gai K, Qiu L, Qiu M, Zhao H (2017b) Intelligent cryptography approach for secure distributed big data storage in cloud computing. Inform Sci, 387: 103–115, 2017b. ISSN 0020-0255. https://doi.org/10.1016/j.ins.2016.09.005. URL https://www.sciencedirect.com/science/article/pii/S0020025516307319

Mukti Rani Sutradhar, N. Sultana, Himel Dey, and Hossain Arif. A new version of kerberos authentication protocol using ecc and threshold cryptography for cloud security. *2018 Joint 7th International Conference on Informatics, Electronics & Vision (ICIEV) and 2018 2nd International Conference on Imaging, Vision & Pattern Recognition (icIVPR)*, pages 239–244, 2018

Omoniwa B, Hussain R, Javed M, Bouk S, Malik S (2019) Fog/edge computing-based iot (feciot): Architecture, applications, and research issues. IEEE Internet Things J, 6: 4118–4149

Reddy AG, Yoon E-J, Kumar Das A, Odelu V, Yoo K-Y (2017) Design of mutually authenticated key agreement protocol resistant to impersonation attacks for multi-server environment. IEEE Access, 5: 3622–3639, 2017. 10.1109/access.2017.2666258

Yoon-Su Jeong and Yong tae Kim (2014) A token-based authentication security scheme for hadoop distributed file system using elliptic curve cryptography. J Comput Virol Hacking Tech 11:137–142

Zhao Hu, Yuesheng Zhu, and Limin Ma. An improved kerberos protocol based on diffie-hellman-dsa key exchange. 2012 18th IEEE international conference on networks (ICON), pages 400–404, 2012