

An authenticated key management scheme for securing big data environment

Thoyazan Sultan Algaradi, Boddireddy Rama

Department of Computer Science, Kakatiya University, Warangal, India

Article Info

Article history:

Received Dec 17, 2020

Revised Jan 19, 2022

Accepted Feb 8, 2022

Keywords:

Authenticated
Big data environment
Diffie–Hellman
Key exchange
Key management
Security

ABSTRACT

If data security issues in a big data environment are considered, then the distribution of keys, their management, and the ability to transfer them between server users in a public channel will be one of the most critical issues that must consider on. In which the importance of keys management may outweigh the importance of the encryption algorithm strength. Therefore, this paper raised a new proposed scheme called authenticated key management scheme (AKMS) that works through two levels of security. First, to concerns how the user communicates with the server with preventing any attempt to penetrate senders/receivers. Second, to make the data sent vague by encrypting it, and unreadable by others except for the concerned receiver, thus the server function be limited only as a passageway for communication between the sender and receiver. In the presented work some concepts discussed related to analysis and evaluation as keys security, data security, public channel transmission, and security isolation inquiry which demonstrated the rich value that AKMS scheme carried. As well, AKMS scheme achieved very satisfactory results about computation cost, communication cost, and storage overhead which proved that AKMS scheme is appropriate, secure, and practical to use and protect the user's private data in big data environments.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Thoyazan Sultan Algaradi, Boddireddy Rama
Department of Computer science, Kakatiya University
Warangal City, Telangana State, 506001, India
Email: yaz.sul77@gmail.com, rama.abbidi@gmail.com

1. INTRODUCTION

The remarkable progress in the development of data production has made data processing difficult [1]. However, due to the rapid development in environmental technologies, the transmission of sensitive information through the internet has become easy using the new technologies that come under big data today [2], [3]. Where it became an important and modern topic racing towards researchers as a rich field of research areas and come out with the most prominent researches and thoughts and innovative ways to deal with the problem and cover its gaps [4]. Big data networks today occupy great importance through the offer of usefulness that can be seized by the user in a big data environment [5], [6] where the user can store his data in the cloud service and share it with others [7]. Also, companies and employees can get great convenience by using cloud computing as modern and exclusive technology that is considered as a big data environment. From this standpoint, it becomes clear the great importance of the security of this environment [8] and the preservation of the privacy of its users through full studies of the most prominent problems of this dilemma [9].

That create the need to find appropriate solutions that ensure privacy, credibility, and validation of the security that preserves the rights well [10], [11]. By considering how users are authenticated to ensure a more reliable connection, it encrypts and sends data to the other party to make it difficult for attacks to

penetrate. The power to do so depends heavily on how encryption keys are configured and managed well between users and how they are passed smoothly and securely [12]. From here, the successful management of keys is essential for the management of security keys for any encryption system [13], [14], once the inventory of keys, keys management consists of three steps as the first is key exchange, second is key storage, third is key use [15]. Where before any secured communication, users must set up the details of the cryptography. In some cases, because of the symmetric key system, it may require exchanging identical keys. In others, it may require possessing the other party's public key. While public keys can be openly exchanged. That what is known as key exchange. Then the keys must be stored securely to keep the connection safe from any unauthorized attempts to use these keys, which is known as key storage. Finally, key use refers to the major issue as the length of time a key is to be used, and therefore the frequency of replacement.

The security of the keys used in the encryption system is directly reflected in the security of the system used [16]. where the keys and their security management are one of the most important factors that must be considered in the encryption and decryption processes and no direct relationship to the vulnerability of the system as a result of the key algorithm itself or the device used [17]. Moreover, the cryptographic system can be public if not for threats that could encounter sensitive user information in case if the key is lost or wrong [18], [19]. Some of the systems depend on third parties such as traditional internet style key exchange and key distribution protocols based on infrastructures were considered as unpractical for whopping scale because of the obscure network topology before deployment, communication range palaces, intermittent sensor-node operation, and network dynamics [20]. Thus, the key transfer over the network is the only practical solution and the appropriate option to store the key but the problem in how to protect the key during the transfer and keep it safe from any attempt to steal or lose during the transfer [21], [22].

Fan *et al.* [7] proposed a secure key management scheme deployed in big data environments to protect user data and privacy, where the keys are divided into three layers. In the layered structure, upper keys encrypt lower keys to guarantee the security of keys in their scheme, the server and others can know nothing of the user's key, and he has to remember the login password only. Their proposed reduced the encryption time, improved key exchange, and key distribution. The idea of this research went from this point, hoping to propose a scheme to ensure the safe transfer and management of these keys to protect user's privacy and data integrity. It protects users' data and privacy by managing the keys safely in a big data environment, emphasizing keeping efficiency, convenience, and security. The proposed states that there are two levels of security during the transfer the data and manage the keys among the server's users, one of which concerns how to communicate between the user and the server, as it ensures verification of a user during their communication and prevents any impersonation attempt of the character. The other is how to maintain the data securely from the server itself so that the concerned receiving user is the only one able to decrypt the data and read it.

The remaining parts of this paper organized as follows: section two introduces the material and method that explain the proposed approach with a detailed explanation of the conception and its phases and give a detail of the system model with explaining the way for ciphertext sharing. Section 3 give an analysis and evaluation by discussing some of the important terms included as key security, data security, public channel transmission, and Security Isolation inquiry, and give a comparison with some other related work on the area of big data security, which concerns the term of computation cost, communication cost, and storage cost. Finally, in section 4, the conclusion and references are provided.

2. MATERIAL AND METHOD

In this section, authenticated key management scheme (AKMS) scheme formulated in four parts, first: explain the idea of AKMS scheme in a way that facilitates the continent to understand and know how it works, then go further to the conception of AKMS approach by introducing its three phases as registration phase, login and verification phase, and communication phase. Moreover, explain the detail of the system model in deep. Finally, present the ciphertext sharing to explain the procedures that AKMS take to work.

2.1. Term and definition

AKMS scheme depends on a several of phases which each have its own steps that include new symbols and parameters to perform the purpose, ensure the efficiency and work at the best manner. This section clarified that to facilitate the understanding on the readers. Table 1 illustrates a brief definition of AKMS scheme critical parameters that will be used.

2.2. Work of AKMS solution

The purpose of this research is to find an approach to manage the keys and transfer them between users in a secure way and ensures the confidentiality of data to solve the problem of data protection and privacy assurance. AKMS scheme provided for mimicking and managing keys between users in a big data

network in the cloud to ensure the security using Diffie–Hellman key exchange. In a way that ensures the safe transmission of data between the users involved in the transmission and the server and then receives data through the user concerned. Communicate between users and the server is by distinct keys that vary from user to user, which created in advance during the registration phase. That adds a double security power because of the already encrypted data that even the server itself does not understand, and no one can decrypt it except the user concerned with the reception that was be predefined by the sender. AKMS scheme consists of three main parts, cloud server, user, and client (browser). After the user registers, he encrypts the data before sending it using a random unique key and then use the agreed key between the user and server to send the encrypted data with the random key ciphertext. What distinguishes the work is that, if the attacker can break the message, it will not get more than the data encrypted vague and will not benefit from it. The power of the user registration phase integrated with the power of Diffie–Hellman key exchange and the random key generation is the core of AKMS scheme. Thus, provide complete protection of users' data during transferring, including the preservation of their privacy. In other words, an essential feature of AKMS scheme is the encryption process that the client-side already do. The file first will be encrypted by a randomly generated key related to the algorithm that will use for the encryption, then encrypt this randomly generated key using a secret shared key agreed between the sender and receiver, created by using Diffie–Hellman key exchange. The encrypted file with the encrypted random generated key created between the sender and receiver sends it to the server using the agreed key between the user and the server that predefined between them. Moreover, the cloud service provider knows nothing of the user's information and key.

Table 1. Term and definition used by the AKMS scheme

Term	Definition
ID	User identity, the username of the registered user.
Pw	User's Password, which keeps safely on the user side.
B	Random private key created for each user on the server-side.
MustKey	Master key of the registered user, created on the server side.
PrivA	Private key of user A and it is required to establish a shared key between the sender user A and receiver.
PubA	Public key of user A and it is required to establish a shared key between the sender user A and receiver.
g, P	g is A primitive root modulo of P (offend called generator), and those are a well-known value agreed to beforehand.
A,B	Corresponding public keys.
H()	One-way hash function.
$K_{A,S}$	The agreed key between user A and the server.
$K_{A,B}$	The agreed key between the user A and user B.
File	The file that will be uploaded to the server to be available for the receiver to download it.
<i>fileKey</i>	A randomly generated key on the user side is used to encrypt the file before transferring.

2.3. The conception of the AKMS solution

2.3.1. Registration phase

The registration phase is one of the most important stages that most systems rely on to protect the privacy of users and keep them safe. Through this phase, the user will be able to transfer his personal information as a username and password by performing some mathematical operations that will contribute to creating more robust security, and that will prevent any penetration during the registration process. It is worth mentioning that each user must choose a valid password (pw) and a unique username (ID). Figure 1 shows the dialogue of this phase:

Algorithm:

1. User-A side - enter the user ID (ID_A) and Password (pw).
2. Send ID_A from user-A to server-side.
3. server-side- Generate random b as a random number and calculate Mustkey=SHA2($ID_A||b$).
4. Send Mustkey from server to user-A side.
5. User-A side- calculate $PrivA=f(pw)$, $pubA=pubA = g^{privA} \bmod P$, and $A = mustkey^{pubA}$.
6. Send A from user-A to the server side.
7. Server-side- Calculate $B = mustkey + mustkey^b$, $pubA = \frac{\log A}{\log mustkey}$.
8. Server-side- store (ID_A , B, b, pubA).
9. Do the same process to register all users.

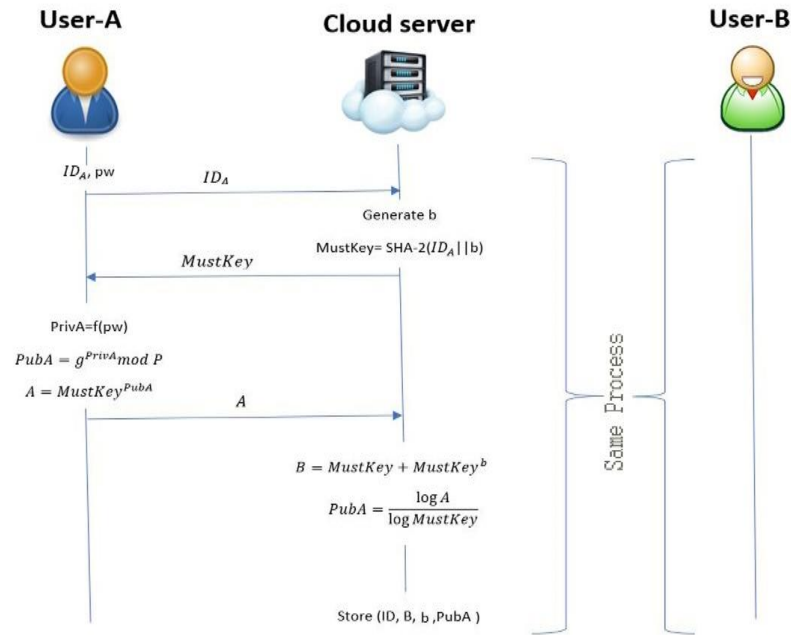


Figure 1. Registration phase of AKMS scheme

2.3.2. Login and verification phase

Here, before starting transfer data, the user must log in to the server by entering the username-ID while keeping the password (pw) on the user side and send any of the other previously registered users-ID he wants to deal with him. The server checks the name of the user sent in his database and gets its random b and B. It also checks the name of the other user that wants to deal with as a receiver to get its public key. Finally, get the MustKey from the random b and send it along with B and pubB to the sender. The continue to get A value, secret key, and get its hash as a key between the sender and server. The user also performs some mathematical operations, such as uses the MustKey to extract the secret key that connects between him and the server. Here, the verification is successful only when this secret key and be equal to that got in the server, or they will not be able to communicate. As well the sender also uses the public key of the receiving user to extract the private secret key between them in complete secrecy, where even the server itself does not know. Figure 2 shows the dialogue of this phase:

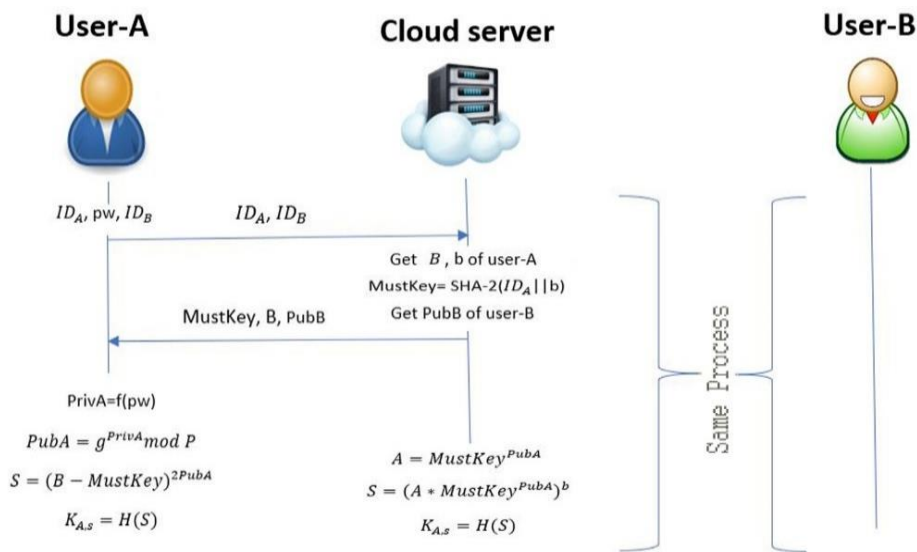


Figure 2. Login phase of the AKMS scheme

Algorithm:

1. User-A side- inter user ID (ID_A) password (pw), and receiver user ID (ID_B).
2. Send ID_A and ID_B from user-A to the server side.
3. Server-side- get b and B value of ID_A , as well as pubB of ID_B .
4. Calculate Mustkey=SHA-2($ID_A||b$).
5. Send Mustkey, B, PubB from server to user-A side.
6. User-A side – calculate $privA = f(pw)$, $pubA = g^{privA} \bmod P$, $S = (B - \text{mustkey})^{2^{pubA}}$, and $K_{A,S} = H(S)$.
7. Server-side – calculate $A = \text{mustkey}^{pubA}$, $S = (A * \text{mustkey}^{pubA})^b$, and $K_{A,S} = H(S)$.
8. Do the same process to authenticate the receiver user ID (ID_B) and get its Key for the server.

2.3.3. Communication phase

After the verification process that took place by the sending user to obtain two secret keys, one is related to his communication with the server, and the second is concerned with the sending user's interaction with the receiver user, which was agreed using Diffie–Hellman key exchange technology. The third stage, during which the user prepare the file to be sent and then encrypt it using a random key (*filekey*), where often determined in a way that is appropriate for the encryption algorithm that used, then the scheme will encrypt this key (*filekey*) using the private key which identified between the sender user and the concerned receiver user. Finally, the scheme will send that encrypted data and the encrypted key of (*filekey*) to the server using the private key between the user and the server itself. Figure 3 shows the dialogue of this phase:

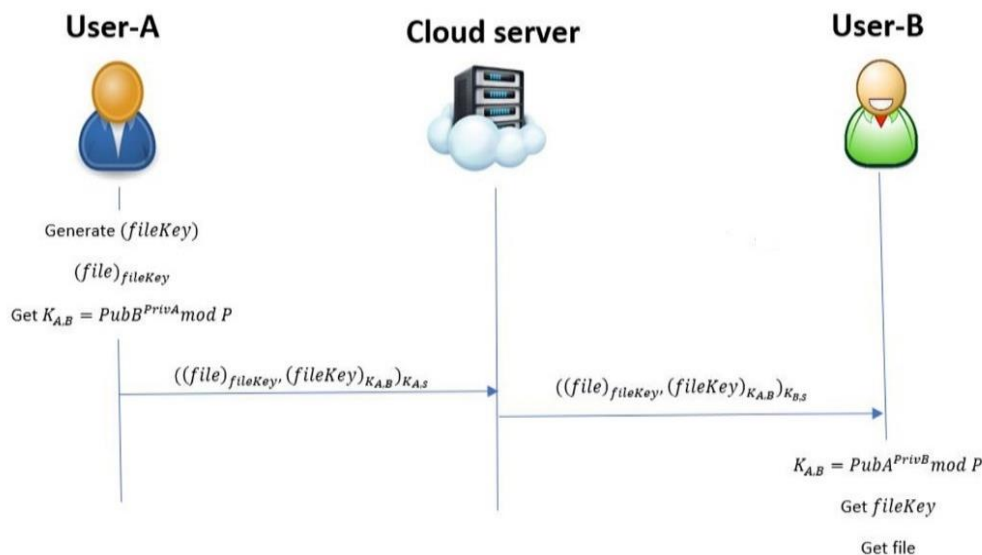


Figure 3. Communication phase of the AKMS scheme

Algorithm

1. User-A side- generate random (*filekey*), encrypt the original file by *filekey* Key, get the secret Key between the sender and receiver $K_{A,B}$, encrypt *filekey* by $K_{A,B}$ and encrypt both by $K_{A,S}$ key.
2. Send $((file)_{filekey}, (filekey)_{K_{A,B}})_{K_{A,S}}$ from user-A to the server side.
3. Server-side- get $(file)_{filekey}$ and $(filekey)_{K_{A,B}}$ then encrypt them again using $K_{B,S}$.
4. Send $((file)_{filekey}, (filekey)_{K_{A,B}})_{K_{B,S}}$ from server to user-B side.
5. User-B side- calculate the secret Key $K_{A,B}$, get the file key, and finally get the original file.

2.4. Detail of system model

2.4.1. Generate *mustkey*

The generated random b in AKMS scheme considered one of the most important things that must generate for each user, where the server stores it for later use to get the master key and then by using the master key identifying the private key ($K_{A,S}$) between it and the concerned user, often at the registration stage for any big data environment application that requires the user to enter a valid password and valid username (ID). In AKMS, the user will keep the password for himself and only send the username (ID). The scheme

generates the master key on the server side itself by obtaining a hash value, where hashing algorithm SHA-2 combines the username (ID) with a random number (b) generated on the server-side. That helps to distinguish users in case if there are users with the same name [23].

2.4.2. Generate *PrivA* and *PubA* and $K_{A,B}$

AKMS scheme guarantees confidentiality in transferring keys between users in a confidential manner, which even the server itself cannot know. Where it generates a random key (*fileKey*) that it uses to encrypt the file before the transfer, the user also encrypts *fileKey* using the unified key created between him and the concerned receiving user, which will rely on Diffie–Hellman key exchange technology, that led by mathematical operations through which both the sender and the receiver can agree on a secret shared key ($K_{A,B}$), which is not sent directly [24]. For that reason, AKMS scheme generate two keys for each concerned user, one private (*PrivA*), and the other is public (*PubA*). The private key entirely dependent on the unshared password used by the sender, while the public key generates using the private key as indicated in (1) and (2), which are finally sent and stored on the server.

$$PrivA = f(pw) \quad (1)$$

$$pubA = g^{privA} \bmod P \quad (2)$$

On the other side, the receiver user takes the same steps using his password to generate his private and public key. By knowing each user, the public key of the other party, and through the use of Diffie–Hellman key exchange technology, the sender and receiver can generate a unified key through which they can communicate separately from the server.

2.4.3. Generate a corresponding public key (A, B), and $K_{A,s}$

It is known that the server would not be able to know any information about the sent data, where the data encrypted on the sender side, and no one able to decrypt it except the concerned receiving user using the secret shared key agreed between them. Moreover, AKMS scheme will add another level of data security to ensure that previously encrypted data will securely reach the server. From here comes the importance of getting a unique secret key agreed between the user and the server to use it during communication, where each user will have its own agreed secret key to deal with the server, which symbolized it for the user-A as $K_{A,s}$. This key is mainly obtained by A and B as a corresponding public key between the user and the server and can be using them calculate the right key, where the following formulas (3) to (6) shows how:

$$A = g^{privID} \bmod P \quad (3)$$

$$B = \text{mustkey} + \text{mustkey}^b, \quad (4)$$

$$S = (A * \text{mustkey}^a)^b = (B - \text{mustkey})^{2pubA} \quad (5)$$

$$K_{ID,s} = H(S) \quad (6)$$

2.4.4. Ciphertext sharing

If user-A wanted to send a file to share with user-B, the scenario would be as follows, user-A logs in by entering his name and password and then entering user-B as the user concerned with receiving the file, user-A chooses the file to send, the server checks for the presence of user-B in its database, then sends the public key (*PubB*) of user-B, and get the master key of user-A himself (*Mustkey* – A), send both to the user-A side, the master key of user-A (*Mustkey* – A) is used to find the shared secret key between the user-A and the server ($K_{A,s}$). Moreover, the scheme generates a random key (*filekey*) complies with the encryption algorithm that used to encrypt the file in the user-A side, and by using the public key of user-B (*PubB*), the scheme will get the shared key that connects the two users ($K_{A,B}$) depending on the Diffie–Hellman key exchange technique. The random key encrypted by the user-A and user-B shared key ($K_{A,B}$), then the ciphertext file along with sharing *filekey* ciphertext will be sent to the server using ($K_{A,s}$) and store it there. On the other side, user-B do the same steps to log in and then get his own ($K_{A,B}$), and ($K_{A,s}$). user-B downloads the storage file and the sharing *filekey* ciphertext from the server using ($K_{B,s}$) that connect between them. Then uses ($K_{A,B}$) to decrypt the sharing *filekey* ciphertext, then by using the obtained (*filekey*), user-B can get the file which user-A shared.

3. RESULT AND DISCUSSION

3.1. Analysis and evaluation

3.1.1. Keys security

AKMS scheme depend on the indirectly transferring keys using mathematical formulas that make it very difficult for attackers to break. As well, it careful not to send any values that might be a reason to reach the main keys, so all the sent values that the attacker may obtain have no meaning or benefit. As observed on the registration and login phases, AKMS provides zero-knowledge security by using the unify key $K = H(S)$, which is the hash result calculated in both sides based on some other mathematical formulae as (7), (8):

$$S = (A * \text{mustkey}^{\text{Pub}A})^b \quad (7)$$

$$S = (B - \text{mustkey})^{2\text{pub}A} \quad (8)$$

where to get both equations, hackers need the value of *PubA*, which is never shared. Moreover, the value *PubID* is dependent on the value of *PrivA* that directly depends on the password of the user himself. That ensures that no user will be able to get the correct key to connect the server unless he shows the correct password used during his registration phase and never shared. Thus, any impersonation attempt also fails. Concerning the communication phase to transfer the file, the AKMS scheme encrypt the file using a generated random encryption key (*filekey*) that is compatible with the encryption algorithm used. Then encrypt this key to get sharing *filekey* ciphertext in a way that only the concerned receiving user be able to break it using Diffie–Hellman key exchange technique, which has proven its security worth in advance. In this way, what will reach the server is the encrypted file along with sharing *filekey* ciphertext. It will be delivered in the same manner to the concerned receiving user without decrypting them.

3.1.2. Data security

In general, AKMS scheme designed to ensure the confidentiality of data and to preserve the security and privacy. It guarantees duple security by using two-level, first is between the user and the server, second between the sending user and his counterpart user who concerned to receive data. Moreover, the scheme encrypts the file before sending it using *filekey* generated random key, then sends and upload that to the server to store. Only the user himself can decrypt the file, and what will send is useless information, even the server cannot understand it. Therefore, the user will encrypt the generated random *filekey* by using a shared key between him and the concerned receiving user, using Diffie–Hellman key exchange technique to transfer the key to the receiver secretly. Thus, maintain and guarantee the preservation of security and protection of data.

3.1.3. Public channel transmission

AKMS scheme provides a double level of security, the first of which is at the user and server, and the other between the sending user and the concerned receiving user. The shared keys on the first level of security are reached in a complex and indirect way, as each user has his unshared password, thus will distinguish him from others and protect him against an impersonation attempt even on his side or on the server-side. Moreover, user data will be sent in its ciphertext format, using a random key (*filekey*), which is generated each time in a way that is compatible with the encryption algorithm used. Finally, get the sharing *filekey* ciphertext, and only the concerned receiving user can break it using the Diffie–Hellman key exchange technique. Thus, even if the big data environment channel attacked, it will become difficult for the attacker to break the ciphertext data transferred between the user and the server, since the user is the only one who can configure the correct key between him and the server, and if any can break it, will only find a ciphertext data that does no benefit.

3.1.4. Security Isolation inquiry

The benefit of a cloud server is in the communication phase to store the ciphertext of users' data and keys in the whole process. The scheme transfers the data in a ciphertext from which no one other than the concerned user can understand it. Also, the server maintains the data security protection through its transferring as well as ensures a strict registration and login stages that difficult to break. Means, the server's function will remain to pass only, and no more than checking the truth of the users and managing the keys between them. Thus, the data stored on the cloud be isolated from the server.

3.1.5. Security features

The security analysis of AKMS proved the strength of the approach to face most of the common attacks through a powerful and strict way to enable the user to register himself in the server in a more robust

secure manner, allow him to get its key to communicating with the server over the untrusted channel. Table 2 shows and summarizes some of the security features of the AKMS scheme.

Table 2. Security features of the AKMS scheme

Attacks	Avoided
Password guessing attack	Yes
Replay attack	Yes
Man in the middle attack	Yes
Brute-force attack	Yes
Dictionary attack	Yes
Insider attack	Yes
Impersonation attack	Yes
User anonymity attack	Yes

3.2. Performance analysis

In this section, the efficiency of AKMS demonstrated with compared to some of the related works concerns to big data security, as this be through a discussion of performance analysis by computing the cost of computation, communication, and storage overhead. The meaning of notations used in the comparison is given in Table 3.

Table 3. The Notation used in computation cost comparison

Symbols	Notation for
T_H	Hash function
T_E	Exponential operation
T_S	Symmetric key encryption/decryption

3.2.1. Computation cost

Table 4 shows the results of the computation cost for AKMS scheme and the works related static knowledge-based authentication mechanism (SKAM) [6], A token-based authentication security (TASS) [25], and A novel authentication framework for Hadoop (NAFH) [23] as well as give a relative comparison regarding the registration phase, Table 5 will do the same for both the login and communication phases.

Table 4. The computation cost of the registration phase

AKMS (The Proposed)	SKAM	TASS	NAFH
$2T_H + T_E$	$2T_S + 3T_H + T_E$	$2T_H + T_E$	$4T_S + T_H$

Table 5. The computation cost of login and communication

AKMS (The Proposed)	SKAM	TASS	NAFH
$3T_S + T_H + T_E$	$8T_S + 2T_H$	$5T_S + 6T_H + 5T_E$	$21T_S$

The total computation cost of the registration phase that represents the user and server is $2T_H + T_E$. It is considered a somewhat good value because this phase will run only once for a single user during his registration to the server. In contrast, the total computation cost of login and communication phases for the user and server is $3T_S + T_H + T_E$. It is considered more efficient compared to the related works compared with due to the symbols.

3.2.2. Communication cost

The communication cost is usually calculated for measuring the efficiency of login and verification phases, as it is frequently executing than other phases. The following values suggested as the basis from which the values of the concerned terms calculated. Where the identity ID, password pw, salt, secret one-way hash function, visualization function, ticket, related message as answers all recommended imposed to being 128-bits long, while the random number, sentences, and the encrypted session key all 1024-bits long, other value as the timestamp is of 32-bits long.

In AKMS scheme, the communication overhead is $3872=(5*160+3*1024)$. As noted, it is better as compared to other related works. Table 6 shows the communication cost for AKMS scheme and other related works such SKAM [6], TASS [25], and NAFH [23], calculated in the same way and assuming that the same

variables in each of them have the same values. Table 6 are graphically drawn in Figure 4 to clearly show the different communication cost of each work measured in bits.

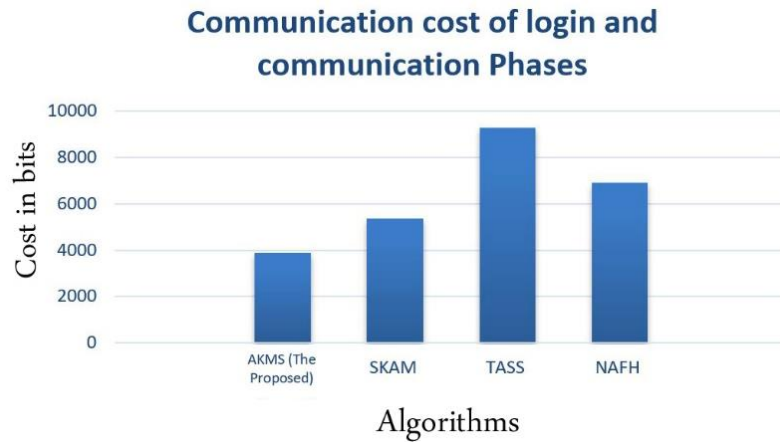


Figure 4. Communication cost of login and communication phase

Table 6. Communication cost of login and communication

AKMS (The Proposed)	SKAM	TASS	NAFH
3872	5376	9280	6912

3.2.3. Storage overhead:

AKMS scheme works by registering some of the user values that the server maintains in its database, then using that in the future to verify the user in the logging phases. It stores (ID, B, b, Pub_{ID}) in the server database, thus the maximum storage cost is 2368 ($2 \times 160 + 2 \times 1024$) bits. Table 7 provides the comparison results of AKMS scheme over the related works SKAM [6], TASS [25], and NAFH [23], and shows that the storage cost of AKMS is relatively reasonable and may become less if the keys used are reduced and made smaller. Table 7 are graphically drawn in Figure 5 to clearly show the different storage cost of each work measured in bits.

Table 7. Storage cost of registration phases

AKMS (The Proposed)	SKAM	TASS	NAFH
2368	1504	6112	3392

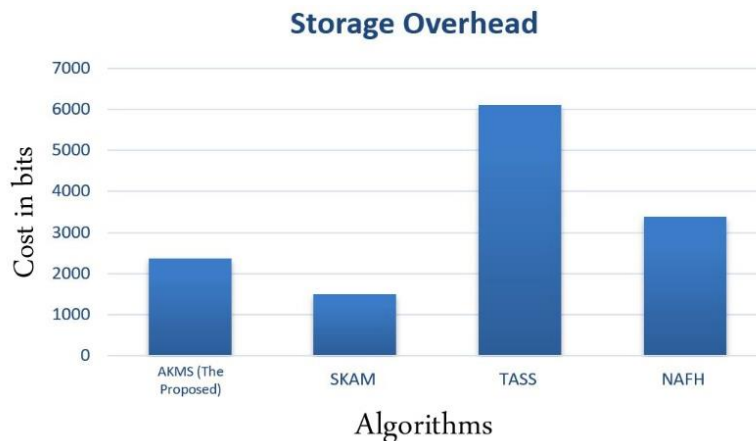


Figure 5. Storage overhead

4. CONCLUSION

This paper covered an authenticated key management scheme for securing big data environment (AKMS) to protect users' data and privacy by managing the keys safely in a big data environment with an emphasis on keeping efficient, convenient, and secure. AKMS scheme works through two levels of security. First, concerns to how the user communicates with the server in a way that prevents any attempt to penetrate the user who sent/receive the data. Second, to make the data sent vague that no one can read except for the concerned user. It starts from the users' registration phase on the server where a common unique key for each user is reached to communicate with the server that entirely depends on the password, in which any communication between them takes place depending only on this key. Second, AKMS encrypts the data to be sent before the transfer process using a chosen random encryption key (*filekey*) in proportion to the used encryption algorithm. Then encrypts this key also to get the sharing *filekey* ciphertext using the shared key between it and the concerned receiving user, which will be accessed by both parties using Diffie–Hellman key exchange technology. AKMS ensured that any data sent will not be penetrated by any of the users, including the server. Thus, provides complete protection of users' data during transferring, including the preservation of their privacy. Moreover, this paper discussed some concepts related to analysis and evaluation as keys security, data security, public channel transmission, and security Isolation inquiry in which demonstrated the rich value that AKMS scheme carried. As well, AKMS achieved very satisfactory results about computation cost, communication cost, and storage overhead.





REFERENCES

- [1] A. Castiglione, A. De Santis, A. Castiglione, and F. Palmieri, "An efficient and transparent one-time authentication protocol with non-interactive key scheduling and update," in *2014 IEEE 28th International Conference on Advanced Information Networking and Applications*, May 2014, pp. 351–358, doi: 10.1109/AINA.2014.45.
- [2] M. E. Hodeish, L. Bukauskas, and V. T. Humbe, "A new efficient TKHC-based image sharing scheme over unsecured channel," *Journal of King Saud University - Computer and Information Sciences*, Aug. 2019, doi: 10.1016/j.jksuci.2019.08.004.
- [3] M. E. Hodeish and V. T. Humbe, "An optimized halftone visual cryptography scheme using error diffusion," *Multimedia Tools and Applications*, vol. 77, no. 19, pp. 24937–24953, Oct. 2018, doi: 10.1007/s11042-018-5724-z.
- [4] U. Narayanan, V. Paul, and S. Joseph, "A novel approach to big data analysis using deep belief network for the detection of android malware," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 16, no. 3, pp. 1447-1454, Dec. 2019, doi: 10.11591/ijeecs.v16.i3.pp1447-1454.
- [5] T. S. Algaradi and B. Rama, "A novel blowfish based-algorithm to improve encryption performance in hadoop using mapreduce," *International Journal of Scientific & Technology Research*, vol. 8, no. 11, pp. 2074–2081, 2019.
- [6] T. S. Algaradi and R. B, "Static knowledge-based authentication mechanism for hadoop distributed platform using kerberos," *International Journal on Advanced Science, Engineering and Information Technology*, vol. 9, no. 3, p. 772, Jun. 2019, doi: 10.18517/ijaseit.9.3.5721.
- [7] K. Fan, S. Lou, R. Su, H. Li, and Y. Yang, "Secure and private key management scheme in big data networking," *Peer-to-Peer Networking and Applications*, vol. 11, no. 5, pp. 992–999, Sep. 2018, doi: 10.1007/s12083-017-0579-z.
- [8] Michael Fire, D. Kagan, A. Elyashar, and Y. Elovici, "Social privacy protector-protecting users' privacy in social networks," In *Proc. of the Second International Conference on Social Eco-Informatics (SOTICS)*, Venice, Italy, 2012.
- [9] T. S. Algaradi and B. Rama, "A new encryption scheme for performance improvement in big data environment using mapreduce," *Journal of Engineering Science and Technology*, vol. 16, no. 5, pp. 3772–3791, 2021.
- [10] N. IrshadHussain, B. Choudhury, and S. Rakshit, "A novel method for preserving privacy in big-data mining," *International Journal of Computer Applications*, vol. 103, no. 16, pp. 21–25, Oct. 2014, doi: 10.5120/18159-9378.
- [11] T. S. Algaradi and B. Rama, "Big data security: a progress study of current user authentication schemes," in *2018 4th International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT)*, Sep. 2018, pp. 68–75, doi: 10.1109/iCATccT44854.2018.9001946.
- [12] A. M. I. Alkhlani and S. B. Thorat, "Lightweight anonymity-preserving authentication and key agreement protocol for the internet of things environment," 2018, pp. 108–125.
- [13] S. H. A. Refish and S. W. Shneen, "E-PAC: efficient password authentication code based RMPN method and diffie-hellman algorithm," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 19, no. 1, p. 485, Jul. 2020, doi: 10.11591/ijeecs.v19.i1.pp485-491.
- [14] F. I. Kandah, O. Nichols, and Li Yang, "Efficient key management for big data gathering in dynamic sensor networks," in *2017 International Conference on Computing, Networking and Communications (ICNC)*, Jan. 2017, pp. 667–671, doi: 10.1109/ICNC.2017.7876209.
- [15] S. Xu, M. Matthews, and C.-T. Huang, "Security issues in privacy and key management protocols of IEEE 802.16," in *Proceedings of the 44th annual southeast regional conference on - ACM-SE 44*, 2006, p. 113, doi: 10.1145/1185448.1185474.
- [16] G. A. Al-Rummana, A. H. A. Al Ahdal, and G. N. Shinde, "A robust user authentication framework for bigdata," in *2021 Third International Conference on Intelligent Communication Technologies and Virtual Mobile Networks (ICICV)*, Feb. 2021, pp. 1256–1261, doi: 10.1109/ICICV50876.2021.9388505.
- [17] M. Ajtai, "Public key cryptosystem and associated method utilizing a hard lattice with $O(n \log n)$ random bits for security," *US 8462940 B2[P]*, 2006.
- [18] D. Denning, "Is quantum computing a cybersecurity threat?," *American Scientist*, vol. 107, no. 2, 2019, Art. no. 83, doi: 10.1511/2019.107.2.83.
- [19] E. Hamida, H. Noura, and W. Znaidi, "Security of cooperative intelligent transport systems: standards, threats analysis and cryptographic countermeasures," *Electronics*, vol. 4, no. 3, pp. 380–423, Jul. 2015, doi: 10.3390/electronics4030380.
- [20] L. Eschenauer and V. D. Gligor, "A key-management scheme for distributed sensor networks," in *Proceedings of the 9th ACM conference on Computer and communications security - CCS '02*, 2002, p. 41, doi: 10.1145/586110.586117.
- [21] R. Zhou, Y. Lai, Z. Liu, and J. Liu, "Study on authentication protocol of SDN trusted domain," in *2015 IEEE Twelfth*





- International Symposium on Autonomous Decentralized Systems*, Mar. 2015, pp. 281–284, doi: 10.1109/ISADS.2015.29.
- [22] D. Dolev and A. Yao, “On the security of public key protocols,” *IEEE Transactions on Information Theory*, vol. 29, no. 2, pp. 198–208, Mar. 1983, doi: 10.1109/TIT.1983.1056650.
- [23] P. K. Rahul and T. GireeshKumar, “A novel authentication framework for hadoop,” In *Artificial Intelligence and Evolutionary Algorithms in Engineering Systems*, Springer, New Delhi, 2015, pp. 333–340, doi: 10.1007/978-81-322-2126-5_37.
- [24] E. Bresson, O. Chevassut, D. Pointcheval, and J.-J. Quisquater, “Provably authenticated group Diffie-Hellman key exchange,” in *Proceedings of the 8th ACM conference on Computer and Communications Security - CCS '01*, 2001, Art. no. 255, doi: 10.1145/501983.502018.
- [25] Y.-S. Jeong and Y.-T. Kim, “A token-based authentication security scheme for hadoop distributed file system using elliptic curve cryptography,” *Journal of Computer Virology and Hacking Techniques*, vol. 11, no. 3, pp. 137–142, Aug. 2015, doi: 10.1007/s11416-014-0236-5.

BIOGRAPHIES OF AUTHORS



Thoyazan Sultan Algaradi     received the Bachelor’s degree in Computer Science and Engineering majoring in Computer Engineering from AlHodeidah University, Yemen, in 2014 and received the M.Sc. degree in Computer Science majoring in Computer Network from School of Computational Science at S.R.T.M. University, Nanded, India, in 2017 and he has received the Ph.D. degrees in Big Data Security from the department of Computer Science, Kakatiya University, Warangal, India, in September 2021. He has published more than 5 referred research papers. His research interests include Information and Network Security, specially on Cryptography, Authentication, and Key Management, Security of Big data, IOT, and Cloud Computing. He can be contacted at email: yas.sul77@gmail.com.



Boddireddy Rama     received her Ph.D. degree in Computer Science from the Sri Padmavathi Mahila Viswa Vidhyalayam (India) in 2008. She has been an Assistant Professor of Computer Science at the Kakatiya University since from 2010. She has published more than 80 referred research papers, 1 book, and has 10 patents filled and pre published. She is the recipient of a Best Paper Award at in International Conference for the paper “A Data mining perspective for forest fire prediction”. Her interests include Artificial Intelligence, Datamining. Previously she was head of the department of computer science and presently chairman board of studies in computer science, network director, and incharge website of the kakatiya university. In her credit 5 phds are awarded and 3 more are submitted under her guidance. She can be contacted at email: rama.abbidi@gmail.com.