# Security Framework for Distributed Data Processing

**Jagadeesh Sai D[1] , Dr.Krishnaraj P M[2]**
Department of Information Science and Engineering, Ramaiah Institute of Technology, Bengaluru, Visveswaraya Technological University, India
[1]Email: djsai@msrit.edu , [2]Email: krishnaraj@msrit.edu



## Abstract:

The increasing significance of distributed data processing services draws attention to the need to develop security solutions that are tailored to meet the requirements of these types of systems. In light of this, the work discussed here suggests a novel approach to implementing a flexible security framework for data processing services in a dispersed environment. The capacity to provide support for large-scale, medium-sized, and small and medium-sized corporate data analytic services is one of the most critical aspects of the framework that has been proposed. The digital signature and a hybrid encryption system that uses modified attribute-based encryption form the basis of the cryptographic backbone of the proposed architecture. This article offers the results of performance testing and provides an overview of the conceptual design of the framework.

***Keywords*:** Security, Big Data Security, SMART Security Platform.

**DOI Number: 10.14704/nq.2022.20.13.NQ88297 Neuro Quantology 2022; 20(13):2380-2387**

## INTRODUCTION

Distributed data processing possesses certain features that set it apart from other applications, such as volume, variety, velocity, value, and truthfulness [1]. These qualities include volume, variety, velocity, value, and veracity. It calls for targeted solutions that can take into account these various facets. A vast amount of data (volume) coming from a range of sources and arriving in the cloud in a variety of forms (variety) is analyzed in real-time (velocity) with a high degree of accuracy (veracity) in order to identify valuable information (value) for a specific end-user application. Additionally, the data must be standardized and managed logically by the cloud or any additional infrastructures, such as hybrid infrastructures [2]. This infrastructure must be protected to ensure the data's confidentiality and prevent inappropriate use of the services it provides. The following components should be included in a practical distributed data security framework:

- authentication services, access control infrastructure for the supported access policies
- encryption protocols and other standard cryptography services
- fast on-the-fly auditing services

All of the industrial techniques [3] presently in use were developed for the Hadoop architecture and use a secure protocol for resource sharing called Kerberos. A complete security solution is not provided by any of the published works on the subject of extensive data security, even though there are works linked to access control policy [4], hybrid environment machine-learning systems [5], and other topics in the literature. Due to this, it can be deduced that no overarching strategies are utilized in this area.

As a result, the theory of distributed data processing security lacks formal, conceptual models, which results in the lack of verifiable solutions for the security of big data [6] and [7]. This study gives a practical framework to satisfy the security criteria mentioned earlier. It defines the formal, conceptual model of the security infrastructure to fill the void left by the previous work. The technique supports several distinct cryptographic mechanisms, each of which was developed specifically for a different level of limiting access to sensitive data. [8] The plan calls for implementing a Small and Medium-sized Enterprise Data Analytic system in Real Time (SMART).

SMART is a modular framework for Big Data analysis that assists in operating Small and Medium-sized services by utilizing clouds or hybrid infrastructures. The fundamental security architecture is implemented inside the isolated cloud environment [9]. The remaining parts of this work are organized as described below. The pertinent related works are discussed in Section 2. In Section 3, you will find a description and overview that goes over the primary characteristics of the cloud architecture that's being suggested. Section 4 will discuss the authentication and access control strategies proposed for this cloud infrastructure. The performance and security testing results are presented in Section 5, together with the fundamental adversary model. The report is brought to a close in Section 6, which also includes some recommendations for further research.

## RELATED WORK

The paper [10] from the Cloud Security Alliance discusses some of the most significant challenges associated with ensuring the safety of distributed data processing. Mapper security, middleware protection, end-point security, real-time monitoring, user privacy protection, and access control are some of the other available types of security. Modifying standard and conventional procedures before deploying them in dispersed service environments is the task that presents the greatest degree of difficulty. It has not been possible to locate the essential components, such as an access control system.

It turns out that Hadoop-oriented security options that are based on the Kerberos resource-sharing protocol [11] are poor when it comes to support and implementation [12], [13], respectively. Aside from that, setting up and configuring the Kerberos infrastructure is a challenging and quite time-consuming endeavor. When it comes to ensuring users' privacy in a cloud computing system, it is best to utilize public-key infrastructure or attribute-based encryption [14]. However, symmetric encryption should be employed to process large amounts of data [15] quickly.

It is possible to conclude that there is a demand for building security models and policies that are unrelated to the security infrastructure of Hadoop based on Kerberos [11], [16], and [17]. These models make it possible to determine the fundamental adversary model [16] and analyze the proposed security architecture to determine whether or not it is consistent. Others can be used for distributed data processing infrastructures, like Flink, which has greater flexibility and can provide a variety of data processing functions [2].



Attribute-based Encryption (ABE) [3, 19] and [18] are, by the findings of the CSA report [10], the cryptographic tools that hold the most promise for the implementation of security frameworks. The most significant difficulty associated with the ABE implementation is the need to apply adjustments to support the critical revocation [20] and simplify the key generation [19].

In response to the mentioned challenges, this research proposes a security model that conforms to the attribute-based access model. The required implementation design has been provided to offer support for the concept. This work intends not only to create the model [11], [16], [17] but also to give a realistic ABE-based framework design and implementation.

## CLOUD INFRASTRUCTURE

### Architecture Overview

Complicated cloud services must be automatically configured on several abstraction layers. It may be necessary to specify configuration parameters at multiple levels in cloud infrastructures that comprise diverse hardware environments [21]. These levels include the operational system, service containers, and network capabilities. Figure 1 presents the solution suggested to represent a hybrid system. This solution depicts a Global Dispatcher and a Global Aggregator that will be utilized on the infrastructure. Since the systems are autonomous, data of varying sizes can be manipulated at the same time.[22]

On Map Reduce applications running in hybrid environments, work has been done in the past.

Middleware operations are handled by the Global Dispatcher, hosted outside the cloud. These functions include the management of data provided by users and the assignment of tasks. It is a centralized data storage system that controls

policies for the splitting of data and distribution of that data per the requirements of each system.

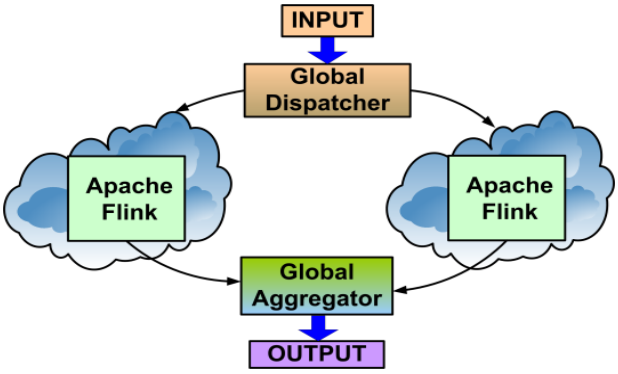


Fig. 1 — SMART architecture

The fundamental idea is comparable to that of a publish/subscribe service (PS-S), in which the system obtains data and publishes computing results [2]. PS-S is a producer/consumer mechanism that operates in a manner analogous to that of a Queue Message. The user defines tasks and local data to be placed in a queue through Apache Kafka in a manner comparable to [23]. The scheduler from the global dispatcher then distributes these jobs in the queue for cloud-hosted computers. In order to produce the whole dataset, the Global Aggregator takes the data that is output by both systems and combines it with its data.

## FORMAL SECURITY MODEL

A cloud-based access control and privacy protection architecture is presently being deployed in protected enterprise cloud storage called Storgrid [9]. The suggested method to secure distributed data processing on the cloud extends that infrastructure. Hybrid attribute-based encryption [14] is at the core of the protected environment in the cloud. This encryption method has additional parameters that enable it to support the protection of heterogeneous devices and attribute access policy while avoiding the immense use of computational resources and optimizing the use of those resources.

As a result, it is appropriate for the SMART cloud-based architecture and the Big Data processing services that correlate with it. When safeguarding user-generated data, the services that demand rapid data processing make use of an approach that is more straightforward. The Storgrid security framework is augmented with the infrastructure shown in Figure 2 below:

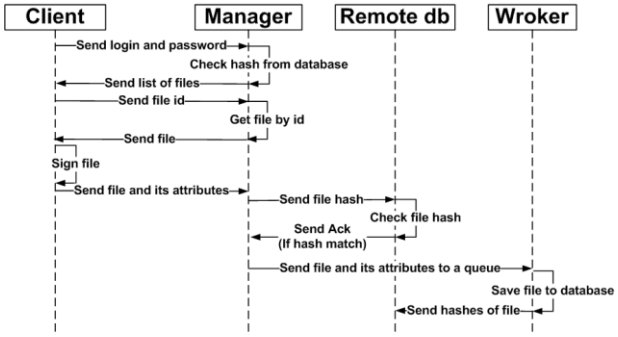


Fig. 2 — SMART, secure method

The two fundamental mechanisms that are used to protect the services provided by such a system are as follows: 1) a digital signature, which ensures that the data entered by a specific user cannot be altered by anyone other than the user himself, and 2) selective cryptography, which ensures that only certain pieces of information are encrypted in order to improve the quality of sort and search service operations. Both of these measures are taken to safeguard the data's integrity.

The original Storgrid hybrid attribute-based AC has been utilized to preserve the privacy of corporate services and provide security for such services. In this particular scenario, the level of performance is decreased in order to increase the level of security.

The approach that has been suggested makes use of several different cryptographic mechanisms. The files, the bulk data, and the multimedia are all secured using 128-bit and 256-bit AES, while the permanent file keys are encrypted using attribute-based encryption. A single AES key about expiring is used to encrypt the entire set of ephemeral ABE keys that correspond to the files that the user can access. This key is referred to as the "KEY SET KEY." The server uses a secret sharing technique that divides this key into four stages; two of these stages are kept on the device, while the other two stages belong to the user (SSS). The process of encryption is described in detail in Figure 3.

The persistent FILE KEY, also known as the unique AES key, is re-encrypted at the beginning of each user session. The collection of FILE KEYs is safeguarded using the appropriate ABE keys.

The one-of-a-kind ABE model provides support for an attribute policy based on user groups and file shares. The model is consistent with the straightforward selected ABE approach [28, 29].

The attribute-based encryption selection scheme is as follows: if at least one attribute in the set to U is equal to at least one attribute in the set to M, then the associated user U can decrypt the text M. If there is no attribute in the set to U that is equal to an attribute in the set to M, the associated user U will be unable to decode the text M. To put it another way, as soon as the user and the share possess at least one characteristic in common, the user will have access to the share.

The following is a list of the components that make up the ABE encryption: Master key (MK), which is stored securely on the server and is only accessible to the domain administrator. It is specified by the equation MK = (t1, t2,..., tn, y), in which the values ti are chosen at random from the group Z p. It should be noted that these are the private keys that correlate to the group attributes. It is crucial to remember that this is not your typical PK encryption because the administrator, not the end users, will have authority over the private keys.

Users can access the information because the public key, also known as the PK, is kept in plain view and is dependent on the values of the master key. PK = (gt 1, gt 2, ..., gt n , .., e(g, g)y) ,

In this case, the bilinear pairing function corresponding to an elliptic curve is denoted as e(g, g).

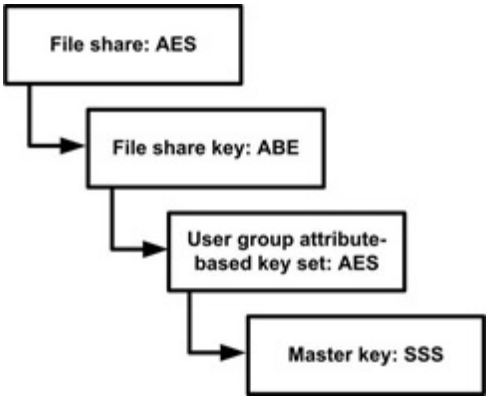


Fig. 3 — Encryption workflow

The attribute set of the user determines the secret user KEY SET. Here, each Di (GROUP KEY) is responsible for the decryption of the information about a specific group of users, such as those involved in a specific project: {ti }U → D = {Di =g yw/ti}.

Encrypted text M, which, in this case, refers to M=FILEKEY, or the permanent AES symmetric key, enables one to avoid re-encrypting the file.



Multiplication is the procedure for encryption. Along with the encrypted text, the following equations are stored: E = M e(g, g)y s, E I = g t I s/w, I ti M The set of public keys E I (PUBLIC SHARE KEY) corresponding to the set of groups that can access the text is stored here.

Decryption is equivalent to division: M = E/Y s

In order for the user to carry out this operation, they are going to require the pair of private key Di and public key Ei that corresponds to the attribute ti: Y s = e(g, g) (g, g) ys = e(Ei, Di) = e(gyw/t I, g t I s/w) = e(g, g)ys.

The symmetric AES key that enables the decryption of the contents of a protected file is produced as a byproduct of the decryption process. This is the FILE KEY.

The infrastructure depicted in Figure 4 serves as the foundation for implementing the recommended security methods.

The encryption server is responsible for managing all of the procedures involving certified authentication (CA) and encryption and granting users access to the data storage. This server can store the encryption keys or link to a second server that serves as the Key Storage. This server is responsible for generating the user keys, connecting to the client UI, running the essential renewal routines, storing the user public keys and attributes in addition to the auditing data, and running the key renewal routines.

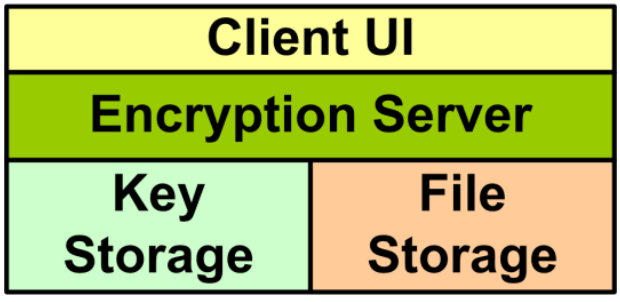


Fig.4 — Security Infrastructure Components

The domain administrator chooses which files to store, and those files are stored, encrypted, and with restricted access. This provides a level of security for the file storage system. It is strongly

suggested that this external component of the file storage be encrypted in its entirety.

The Client UI can connect to the Encryption server and request authorization to access the file storage to view, edit, or upload particular files or folders. Client UI is responsible for storing the user keys used for ABE encryption and the one-of-a-kind symmetric session keys used to restrict access to the downloaded files. Encryption of the symmetric keys is performed using the ABE keys. The client supports multiple computer platforms and operating systems.

The hybrid encryption technology makes it possible to monitor and manage the users' privacy without negatively impacting the total encryption speed. The fundamental ABE [18], [19] technique was adjusted to support the attributes related to both file shares and user groups and set up the validation period based on the user key and support the user key.

The suggested security architecture will be integrated into the Big Data environment in the next section.

4.1 Security Framework for SMART Cloud-Based Processing Service

The encryption server's authentication, certificate authority, and encryption operations need to be kept separate. These protection services come into play once the data have been uploaded.

Figure 5 depicts six different modules: the Global Collector, the Global Dispatcher, the Storage, the Core Engine, and the Global Aggregator and Central Monitoring.

The Core Engine needs to be able to support hybrid systems, which means it must perform both streaming and batch computations simultaneously. As a result, the Flink framework is an essential system that should be considered. Another system designed to boost computational performance, MR-BitDew [24], takes advantage of volunteer computing within hybrid infrastructures. An Application Programming Interface (API) for Clients makes it simple for users to submit their applications and specify the data sources. A single user can be identified with the help of an encrypted key, which is made possible by the Client UI, a security interface. The Encryption-Decryption Engine requires a key in order to function correctly.

The Global Collector layer is responsible for the management and coordination of the sensing modules. It is responsible for obtaining data from a variety of sources and for maintaining the systems that ensure the data's integrity. The data is gathered and serialized using a conventional TCP/IP protocol, which makes up the communication stack sent to the Global Dispatcher.



The data in the message queue mechanism of the Global Dispatcher is separated from the lower layers where it usually resides. In Cloud/Multi-Cloud and Grid/Multi-Grid systems, the data is placed in a FIFO queue before it is sent out to be distributed to servers by the resources available on those servers. The Optimization layer analyses the quantity of the input data. It decides, with the help of the Decision Engine, regarding scheduling jobs and data across various environments.

A simulation process implements an execution time prevision, and the Decision Engine will use this information to improve the accuracy of the scheduling mechanism. The user deploying module allows it to save unsecured data encrypted on an external server.

Before definitively attaching the data, the user must supply a key storage localization and the data path that will travel through the network. Within this layer, the pass-through and storage-and-forward protocols are implemented in a manner that is dependent on the data source.

After being processed on Core Engine, the intermediate results are serialized and sent to Global Aggregator, which is in charge of the data consolidation mission. Data integrity and data integration are supported by the module known as Data Integration. The final stage of the data processing procedure is meant to produce an iterative execution and supply the outcome consolidation.

It is required to have a Communication API in order to integrate employees into a virtual network for data computation. The Aggregator Application Programming Interface (API) is a module that orchestrates the aggregation of results and maintains the safety data mechanism for end users. A centralized monitoring system presents the information through the End-User Interface with an appealing visual characteristic.

This integration architecture enables the required level of security and user privacy to be implemented in all governmental and corporate organizations and services. The flexible implementation architecture protects user privacy and controls access to sensitive data in a heterogeneous environment via the Client UI created for many operating systems and platforms.

This design is extendable to use more advanced data and key protection techniques, such as secret sharing or complex attribute regulations.

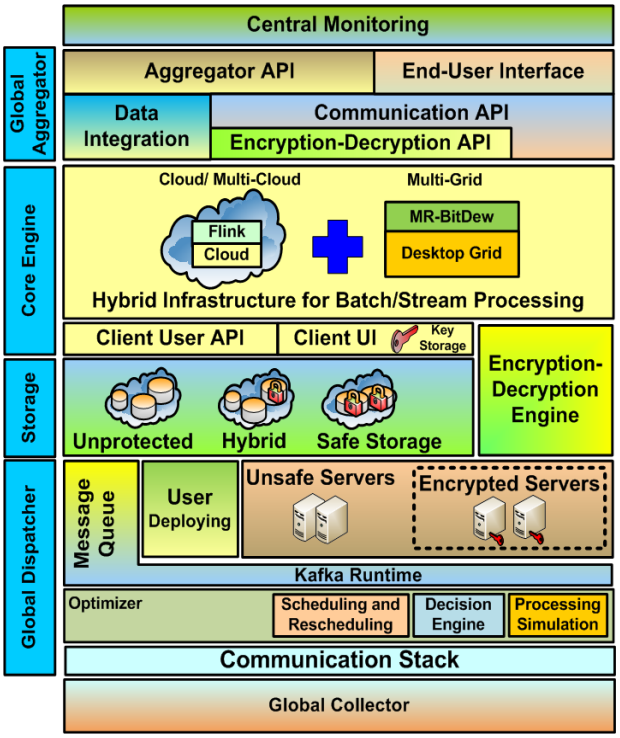


Fig.5 —Components of the security infrastructure and their interactions

## 5. IMPLEMENTATION AND PRELIMINARY RESULTS

As part of the Storgrid project, the suggested security architecture is now being developed. Storgrid is a safe and reliable cloud storage management system. This section gives some findings and analysis of a security framework similar to the model described. The administrator (or domain manager) oversees server operations and has access to the server via a secret password. They must determine the level of protection for protected/public files and set an authorization policy through a web interface.

The standard encryption technique on the server is 128-bit AES. The file storage secret key has an expiry date to boost security. The method of key renewal and re-encryption occurs daily and does not affect performance because it runs when the system is inactive or has little activity. Auditing and log analysis performed on the server side does not permit a key attack.

Table 1 displays the preliminary results of the re-encryption testing retrieved from the real-time server log. It indicates that, even for large amounts of data, the average encryption time is relatively short compared to the extent of the data acquired by the SMART services. The server load determines the encryption time at the moment of encryption. The results show that the protected file re-encryption in the local file system has a low overhead.

As a result, this re-encryption process is compatible with the existing SMART infrastructure. Another reason to use re-encryption is that the model described does not employ the concept of complete data encryption. To save overhead, this allows selective encryption of sensitive user data. This notion ensures that the internal network is not overburdened and that file security is maintained. The domain manager access control utility manages this operation.

| Server Log | Files Number | Duration (ms) |
|---|---|---|
| srv.log_2015-09-18 18:06:28; [id = 1003;] | 16,304 | 324.345 |
| srv.log_2015-09-18 18:06:28; [id = 1004;] | 35,834 | 162.629 |
| srv.log_2015-09-18 18:07:23; [id = 1005;] | 21,501 | 124.399 |
| srv.log_2015-09-18 18:12:32; [id = 1006;] | 23,651 | 149.948 |
| srv.log_2015-09-18 18:40:00; [id = 1007;] | 28,614 | 523.232 |
| srv.log_2015-09-18 18:40:00; [id = 1008;] | 49,494 | 397.334 |
| srv.log_2015-09-18 18:55:02; [id = 1008;] | 41,751 | 533.838 |
| srv.log_2015-09-18 19:01:38; [id = 1010;] | 45,360 | 368.900 |

Table 1. — Re-encryption time results

The client is in charge of security on both the user and the cloud. It is available on a variety of platforms. The proposal supports the concept of a light-weighted client, which does not do much encryption or other critical/heavy processes. The server handles all encryption, and the client only supports key usage. The exciting feature is that if the keys remain the same, the encrypted data are not downloaded again. After authenticating with his password and email, the user can access his downloaded file, which is safely stored (encrypted with a unique AES key) on his storage device.

The server, ABE encryption, and key expiry period govern access to the AES key. Because of the implementation architecture, regenerating public keys and resending them to users is not time-consuming. Table 2 shows the results of testing the user key generation technique for 1000 user logins and 1000 key samples over 20 parallel threads. These findings indicate that the suggested security framework is intended to reduce network burden while maintaining user security. These characteristics fully complement the SMART architecture's security requirements.

| User numbers | Average | Min | Max | KB/s | Avg. Bytes |
|---|---|---|---|---|---|
| 30 | 0.82 | 38.4 | 35.7 | 154 | 320 |
| 60 | 0.67 | 42.1 | 34.7 | 138 | 340 |
| 120 | 0.52 | 45.1 | 34.0 | 124 | 370 |

Table 2. — User Key Generation Procedure Results

## 6. CONCLUSION

This study presents a security architecture for various distributed data processing systems scales. This concept, in particular, is aimed at cloud-based systems that manage vast amounts of data in real-time analysis from highly various data sources, which is the rationale for the SMART architecture. The suggested model includes authentication and access control techniques that are adequate for dealing with the high demands of the applications that SMART will support.

The framework testing shows a low overhead without sacrificing the security level required for SMART application requirements.

## REFERENCES

[1] M. Stonebraker, S. Madden, and P. Dubey, “Intel “Big Data” Science and Technology Center Vision and Execution Plan,” SIGMOD Rec., vol. 42, no. 1, pp. 44–49, May 2013. [Online]. Available: http://doi.acm.org/10.1145 /2481528.2481537.

[2] H. Jin, Z. Li, D. Zou and B. Yuan, "DSEOM: A Framework for Dynamic Security Evaluation and Optimization of MTD in Container-Based Cloud," in IEEE Transactions on Dependable and Secure Computing, vol. 18, no. 3, pp. 1125-1136, 1 May-June 2021, doi: 10.1109/TDSC.2019.2916666.

[3] J. C. S. Anjos, G. Fedak, and C. F. R. Geyer, “BIG hybrid: a simulator for Map Reduce applications in hybrid distributed infrastructures validated with theGrid5000 experimental platform,” Concurrency and Computation: Practice and Experience, vol. 1, no. Special Issue Paper, pp. 1–24, September 2015, cpe.3665.

[4] D. B. Rawat, R. Doku and M. Garuba, "Cybersecurity in Big Data Era: From Securing Big Data to Data-Driven Security," in IEEE Transactions on Services Computing, vol. 14, no. 6, pp. 2055-2072, 1 Nov.-Dec. 2021, doi: 10.1109/TSC.2019.2907247.

[5] Z. Qiao, S. Liang, S. Davis, and H. Jiang, “Survey of attribute based encryption,” in Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing (SNPD), 2014 15th IEEE/ACIS International Conference on, June 2014, pp. 1–6.

[6] I. Hababeh, A. Gharaibeh, S. Nofal and I. Khalil, "An Integrated Methodology for Big Data Classification and Security for Improving Cloud Systems Data Mobility," in IEEE Access, vol. 7, pp. 9153-9163, 2019, doi: 10.1109/ACCESS.2018.2890099.

[7] V. C. Hu, D. Ferraiolo, R. Kuhn, A. Schnitzer, K.Sandlin, R. Miller, and K. Scarfone, “Computer Security, ”Tech. Rep. 800-162, Jan. 2014, accessed in September 2015. [Online]. Available: http://dx.doi.org/10.6028/NIST.SP.800-162

[8] C. Bakir, "New Blockchain Based Special Keys Security Model With Path Compression Algorithm for Big Data," in IEEE Access, vol. 10, pp. 94738-94753, 2022, doi: 10.1109/ACCESS.2022.3204289.

[9] D. Chen and H. Zhao, “Data Security and Privacy Protection Issues in Cloud Computing,” in Computer Science and Electronics Engineering (ICCSEE), 2012.International Conference on, vol. 1, March 2012, pp. 647–651.

[10] R. Jiang, M. Shi and W. Zhou, "A Privacy Security Risk Analysis Method for Medical Big Data in Urban Computing," in IEEE Access, vol. 7, pp. 143841-143854, 2019, doi: 10.1109/ACCESS.2019.2943547.

[11] B. D. S. BV, "Storgrid EFSS: Secure Enterprise File Sharing Software," Jan. 2016, available from Internet:http://www.storgrid.com/.

[12] J. Moreno, E. B. Fernandez, M. A. Serrano and E. Fernández-Medina, "Secure Development of Big Data Ecosystems," in IEEE Access, vol. 7, pp. 96604-96619, 2019, doi: 10.1109/ACCESS.2019.2929330.

[13] S. Rajan, W. V. Ginkel, N. Sundaresan et al." Expanded Top Ten Big Data Security and Privacy Challenges," Tech. Rep., Apr. 2013. [Online]. Available: https://downloads.cloudsecurityalliance.org/initiatives/bdwg/Expanded Top Ten Big Data Security and PrivacyChallenges.pdf

[14] D. Dharminder, M. S. Obaidat, D. Mishra and A. K. Das, "SFEEC: Provably Secure Signcryption-Based Big Data Security Framework for Energy-Efficient Computing Environment," in IEEE Systems Journal, vol. 15, no. 1, pp. 598-606, March 2021, doi: 10.1109/JSYST.2020.2978015.

[15] B. Lublinsky, K. T. Smith, and A. Yakubovich, Professional Hadoop Solutions, 1st ed. 10475 Crosspoint Boulevard: John Wiley & Sons, Inc., Sep. 2013

[16] P. Yang, N. Xiong and J. Ren, "Data Security and Privacy Protection for Cloud Storage: A Survey," in IEEE Access, vol. 8, pp. 131723-131740, 2020, doi: 0.1109/ACCESS.2020.3009876.

[17] T. Galibus and H. Vissia, "Cloud storage security,"in Network Security and Communication Engineering,K. Chan, Ed. CRC Press, Jun. 2015, pp. 123–126.

[18] F. M. Awaysheh, M. N. Aladwan, M. Alazab, S. Alawadi, J. C. Cabaleiro and T. F. Pena, "Security by Design for Big Data Frameworks Over Cloud Computing," in IEEE Transactions on Engineering Management, vol. 69, no. 6, pp. 3676-3693, Dec. 2022, doi: 0.1109/TEM.2020.3045661.

[19] Z. Liu and D. S. Wong, "Practical Attribute-Based Encryption: Traitor Tracing, Revocation, and LargeUniverse," Cryptology ePrint Archive, Report 2014/616,2014, http://eprint.iacr.org.

[20] D.-H. Le, H.-L. Truong, G. Copil, S. Nastic, and S. Dustdar, "SALSA: A Framework for Dynamic Configuration of Cloud Services," in Cloud Computing Technology and Science (CloudCom), 2014 IEEE 6thInternational Conference on, Dec 2014, pp. 146–153.

[21] E. Hossain, I. Khan, F. Un-Noor, S. S. Sikander and M. S. H. Sunny, "Application of Big Data and Machine Learning in Smart Grid, and Associated Security Concerns: A Review," in IEEE Access, vol. 7, pp. 13960-13988, 2019, doi: 10.1109/ACCESS.2019.2894819.

[22] T. Zhang, "Reliable Event Messaging in Big Data Enterprises: Looking for the Balance Between Producers and Consumers," in Proceedings of the 9th ACM International Conference on Distributed Event-Based Systems, ser. DEBS '15. New York, NY, USA: ACM, 2015, pp. 226–233. [Online]. Available: http: //doi.acm.org/10.1145/2675743.2771878

[23] T. Y. Win, H. Tianfield and Q. Mair, "Big Data Based Security Analytics for Protecting Virtualized Infrastructures in Cloud Computing," in IEEE Transactions on Big Data, vol. 4, no. 1, pp. 11-25, 1 March 2018, doi: 10.1109/TBDATA.2017.2715335.

[24] J. Shen, D. Liu, Q. Liu, X. Sun and Y. Zhang, "Secure Authentication in Cloud Big Data with Hierarchical Attribute Authorization Structure," in IEEE Transactions on Big Data, vol. 7, no. 4, pp. 668-677, 1 Oct. 2021, doi: 10.1109/TBDATA.2017.2705048.