A

Project Report

On

Investigation and Finding a DNA Cryptography layer for Securing data in Spark Cluster

A Report submitted in partial fulfillment of

the requirements for the award of the degree

Submitted by

RETALLA NIKHIL GOUD (20EG105318)

KADIYALA VENKAT SATHWIK (20EG105319)

KUNDALA AKSHAYA (20EG105322)

Under the guidance of

DR J BALARAJU (M.Tech.,Ph.D)

Assistant Professor

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

Anurag University

VEKATAPUR– 500088 TELANGANA Year 2023-24

DECLARATION

We hereby declare that the Report entitled "**Investigation and Finding A DNA Cryptography layer for Securing data in Spark Cluster"** submitted for the award of Bachelor of technology Degree is our original work and the Report has not formed the basis for the award of any degree, diploma, associate ship or fellowship of similar other titles. It has not been submitted to any other University or Institution for the award of any degree or diploma.

Place: Anurag University,
Hyderabad

Date : 15/04/2023

20EG105318 RETALLA NIKHIL GOUD
20EG105319 KADIYALA VENKAT SATHWIK
20EG105322 KUNDALA AKSHAYA

CERTIFICATE

This is to certify that the Report entitled "Investigation and Finding A DNA Cryptography layer for Securing data in Spark Cluster" that is being submitted by **RETALLA NIKHIL GOUD** (20EG105318), **KADIYALA VENKAT SATHWIK**(20EG105319), **KUNDALA AKSHAYA**(20EG105322) under the guidance of **DR J BALARAJU** in partial fulfillment for the award of B.Tech in Computer science and engineering to the Anurag University is a record of bonafide work carried out by them under my guidance and supervision.

The results embodied in this Report have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr J Balaraju

Assistant Professor

Head of department **Signature of Supervisor**

ACKNOWLEDGEMENT

It is our privilege and pleasure to express my profound sense of respect, gratitude and indebtedness to our guide **Dr J Balaraju,** Assistant Professor, Department of Computer Science, Anurag University, for his guidance, discussion, encouragement and valuable advice throughout the dissertation work. His motivation in the field of Big Data has made us to overcome all hardships duringthe course of study and successful completion of the project.

We would like acknowledge our sincere gratitude for the support extended by **Dr. G. Vishnu Murthy**, Dean, Dept. of CSE, Anurag University . We also express my Deep sense of gratitude to **Dr. V V S S S Balaram** , Academic coordinator. **Dr. Shyam Prasad** Project Co-coordinator and all other Project review committee members, whose research expertise and commitment to the highest standards continuously motivated me during the crucial stage our project work.

We would like to thank all the faculty of the CSE department, for their support and encouragement.

ABSTRACT

Spark is a very versatile, adaptable platform for storage of data in real time and processing and focusing data analytics, data driven applications, however it was not begun in light of security or authorization for data. Spark is fitting for dealing with efficient in storage and analysis of data and it has certain security issues and also it uses third party security which used huge computations in all the versions of Spark.

This paper proposed a new security mechanism as single security instance gathering metadata from Name node at regular intervals. SEAL (Spark E-Authentication Layer) as a secure layer which positioned above the Spark Cluster. Authentication of users is essential and plays a crucial role in securing data. This developed protocol SEAL is providing security using fewer computations, unlike integrating a third-party authentication protocol like Kerberos. Every new user must send a request to SEAL by sending an email id and it is generating a permanent unique key from the Metadata for each user using DNA sequences as single security instance forwards the created unique key to the user. The user is to be allowed in the spark cluster by verifying the user unique key.

CONTENTS

7. SUMMARY

8. REFERENCES

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATIONS	
NOTATION	**DESCRIPTION**
SEAL	Spark internet authentication layer
DNA	Deoxyribonucleic acid
DHCP	Dynamic host configuration protocol
CORS	Cross-origin resource sharing
DB	Database

1.INTRODUCTION

1.1 Overview

Spark is fitting for dealing with efficient in storage and analysis of data and it has certain security issues with the usage of third party security for authentication which used huge computations in all the versions of Spark. This desired method is a new security mechanism as single security instance, a SEAL (Spark E-Authentication Layer) as a secure layer which positioned above the Spark Cluster.

This developed protocol SEAL is providing security using fewer computations, unlike integrating a third-party authentication protocol like Kerberos. This authentication step ensures that only users possessing the correct unique DNA key, received through email, are granted access to Apache Spark for distributed data processing.

1.2 Problem Statement

Spark is a very versatile, adaptable platform for storage of data and focusing data analytics, data driven applications, however it was not begun in light of security or authorization for data. Security and controlling data are most significant fragments of any distributed platform that wants to break into the endeavor standard.

Existing method(s) disadvantages:

The primary existing problem in user authentication within spark is the reliance on third-party security measures, such as Kerberos, which introduces computational overhead. The fragmented nature of security capabilities, developed independently, complicates user authentication and access control.

Users face challenges in securing their data, particularly in the absence of a unified security mechanism. The need for a single security instance for user authentication becomes apparent to address these existing authentication issues in spark**.**

1.3 Problem Illustration

Consider a scenario where multiple users attempt to access and interact with data stored in a Spark Cluster. In the current setup, the user authentication process relies on third-party security protocols, such as Kerberos.

1. Fragmented Authentication Mechanisms:

The illustration depicts different users (User A, User B, and User C) attempting to authenticate through various security mechanisms.

User A encounters authentication through Kerberos, while User B goes through a different process.

This fragmentation in authentication mechanisms complicates the overall security and escape, making it challenging to maintain a cohesive and standardized approach.

2. Computational Overhead:

Each user's authentication involves multiple computational steps and interactions with external authentication services.

This process introduces computational overhead, as illustrated by the intricate network of calculations and verifications

Summary: The problem lies in the absence of a security solution that can verify the authenticity of nodes in the Spark Cluster. Without such a solution, malicious nodes can infiltrate the cluster, posing a significant security risk, and potentially leading to data breaches, trust violations, and cluster disruptions.

2.LITERATURE SURVEY

2.1 Existing Methods

Author(s)	Strategies	Advantages	Disadvantages
Nan Zhang, Et al.	Kerberos Authentication	• Strong security: Kerberos is a well-established and widely used authentication protocol. • Centralized authentication: It uses a centralized Key Distribution Center (KDC) for secure authentication.	• Complexity: Setting up and maintaining a Kerberos infrastructure can be complex. • Overhead: Introduces computational overhead due to the need for ticket requests and validations.
Yong-Tae Kim Et al.	Custom Token-based Authentication	•Flexibility: Custom token-based authentication allows for a flexible and tailored solution. • Stateless: Tokens can be designed to be stateless, reducing the need for server-side storage.	•Development effort: Implementing a custom solution requires additional development effort. • Security risks: Custom solutions may have security risks if not designed and implemented correctly.
Paul J. E. Velthuis	Multi-factor Authentication (MFA)	• Enhanced security: Adds an extra layer of security through multiple authentication factors. •Compliance: Meets security compliance requirements in certain environments.	• User experience: MFA may introduce additional steps for users, impacting user experience. •Implementation complexity: Implementing and managing MFA can be complex.

Table 2.1 Existing method literature abstraction

Kerberos Authentication, pioneered by Nan Zhang et al., offers robust security through a well-established and widely used authentication protocol. It employs a centralized Key Distribution Center (KDC) for secure authentication, providing a centralized authentication mechanism. Despite its advantages, such as strong security and

centralized authentication, Kerberos can be challenging to set up and maintain due to its inherent complexity and computational overhead. Custom Token-based Authentication, as proposed by Yong-Tae Kim et al., provides flexibility and scalability by allowing developers to tailor authentication solutions according to specific requirements. This approach offers stateless authentication, reducing the need for server-side storage. However, implementing custom token-based authentication requires additional development effort and may introduce security risks if not carefully designed. Multi-factor Authentication (MFA), advocated by Paul J. E. Velthuis, enhances security by requiring multiple authentication factors. It meets security compliance requirements and adds an extra layer of protection against unauthorized access. Nonetheless, MFA may complicate the user experience with additional authentication steps and can be challenging to implement and manage due to its complexity.

2.2 Proposed Method Literature

Hwang et al. focus on maintaining the integrity and functionality of DNA sequences while preventing false start codons and optimizing embedding capacity using efficient coding and recovery techniques. Their method offers advantages such as high embedding capacity and prevention of false start codons, ensuring reliable data hiding. However, the approach introduces complexity and processing overhead, limiting its applicability in certain scenarios.

Balaraju. J et al. propose a sophisticated authentication mechanism based on complex DNA cryptography, providing an innovative approach to data security by converting big data into DNA with reduced computational tasks. Their method offers innovative security solutions and leverages DNA cryptography effectively. Nonetheless, it faces challenges such as limited practical validation and complex implementation, which may hinder its widespread adoption.

A. Vikram et al. utilize DNA symmetric cryptography and DNA computing to convert plain-text into a non-understandable format, enhancing information security and preventing unauthorized access to confidential data. Their approach enhances security and offers resistance to conventional attacks. However, it involves complexity and resource intensity in implementation, and its adoption and standardization may be limited.

Author(s)	**Method**	**Advantages**	**Disadvantages**
Hwang et al	Emphasis is on maintaining the integrity and functionality of the DNA sequence, preventing false start codons, and achieving optimal embedding capacity through efficient coding and recovery techniques.	1.High Embedding Capacity 2. False Start Prevention	1.Complexity and Processing Overhead 2.Limited Applicability
Balaraju. J et al	Proposed a sophisticated authentication mechanism using complex DNA cryptography as an alternative to converting big data into DNA with reduced computational tasks.	1.Innovative Security Approach 2.DNA Cryptography Solution	1.Limited Practical Validation 2.Complex Implementation
A. Vikram et al	The method employs DNA symmetric cryptography, utilizing DNA computing to convert plain-text into a non-understandable format, providing enhanced information security and preventing unauthorized access to confidential data.	1.Enhanced Security 2.Resistance to Conventional Attacks	1.Complexity and Resource Intensity 2.Limited Adoption and Standardization

Table 2.2 Selected Strategy literature abstraction

3.PROPOSED METHOD

3.1 Illustration

A Secure E-Authentication Interface (SEAL) as a secure layer which positioned above the Spark Cluster. Any Single Cluster first registers into the cluster through web interface and the user will get information to authenticate into the cluster. This mechanism provides better authentication for the spark cluster by using limited computations and providing better user security from hackers.

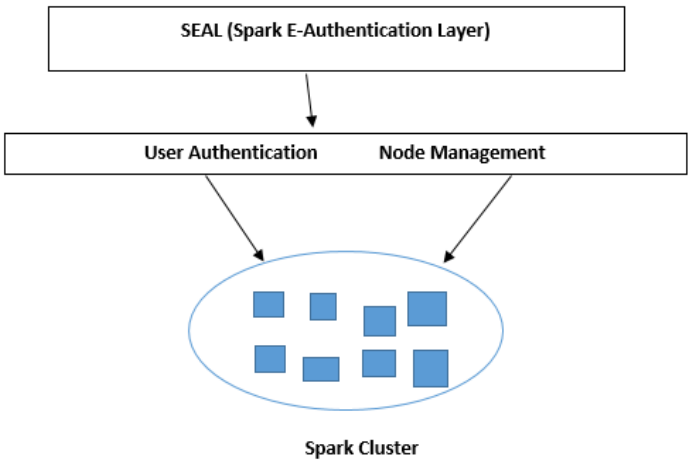

Fig3.1 Proposed Authentication Interface

The proposed project aims to develop a secure user authentication system that leverages a unique DNA sequence generated from user-provided email and username. The process involves a web interface where users input their email and username, triggering the generation of a DNA sequence unique to each user via Gmail. This DNA sequence is sent to the user's email for authentication. Once the user receives the DNA sequence, they can initiate a Spark job, but only if the received DNA sequence matches the required sequence for corresponding user, ensuring that only authorized users with the correct DNA sequence can access and contribute to the Spark cluster.

Steps	Description	Mac address	IP address	Hostname
	Node Data	5c:ea:1d:99:15:67	192.168.1.11	sathwik
1	Combined Data	5c:ea:1d:99:15:67192.168.1.11sathwik		
2	Binary Form	001101010110001101100101011000010011000101100100001110010011100100110001001101010011011000111001000011100100111001000110001001101100011100000011000100100110001011100110110000101110100011010000111011101101010010110101011		
3	DNA Form	ATCCCGATCGCCCGACATACCGCAATGCATGCATACATCCATCGATCTATACATGCATAGATACATCGATGAATACATACATACCTATCGACCTCACGGACTCTCGGCCGGT		
4	Mapping (A=0,C=1,G=2, T=3)	031112031211120103011210032103210301031103120313030103210302030103120320030103010301130312011310122013131221223		
5	Decimal to Hexa(unique key)	8479499BF7DAAA012C0BB09069106B5BDFB839F2BA2A1B08835773A1C685AB4E09EC7076F3BDFAEBEF54D81F9517		

Fig3.2 Generation of unique key through DNA Cryptography

3.2 User flow:

1. User Input and DNA Sequence Generation: The web interface allows users to input their username and email address. Upon clicking the "Generate DNA Sequence" button, the system converts the email address into a DNA sequence using a binary-to-DNA mapping algorithm. This unique DNA sequence serves as the authentication key for accessing the Spark cluster.

2. Sending Email with DNA Sequence: After generating the DNA sequence, the system sends an email to the provided email address containing the DNA sequence. This step ensures that users receive their authentication key for accessing the Spark cluster.

3. Authentication with Spark Cluster: To access the Spark cluster, users must provide their email address and the DNA sequence received in the email. The Spark cluster verifies the provided DNA sequence against the stored DNA sequences in the Mongo DB database. If the DNA sequence matches, the user is granted access to the Spark cluster.

4. Integration with Mongo DB: User information, including the email address, username, and DNA sequence, is stored in a Mongo DB database. This database serves as the backend storage for authentication and allows for efficient retrieval and verification of user information during the authentication process.

5. Execution of Spark Jobs: Once authenticated, users can submit Spark jobs to the cluster for processing. The Spark cluster performs the requested data processing tasks while ensuring that only authenticated users with valid DNA sequences are allowed to execute jobs.

6. Enhanced Security and User Experience: By leveraging DNA sequences as authentication keys, the proposed method enhances security by adding an extra layer of authentication. Moreover, the user experience is improved as users receive their authentication keys via email, simplifying the authentication process and ensuring smooth access to the Spark cluster

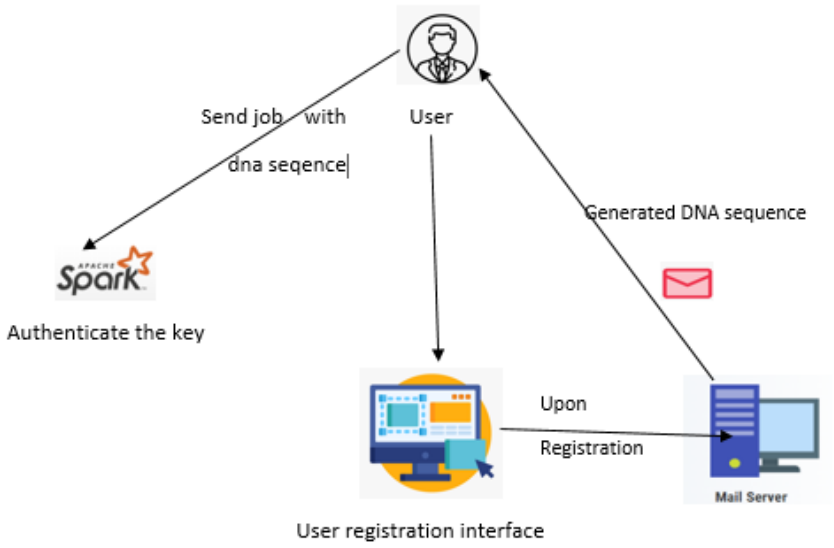

Fig 3.3 User flow through the proposed method

4.IMPLEMENATION

4.1 List of program files

4.1.1 index.js

Attributes:

- express: Express.js framework for building web applications.
- nodemailer: Nodemailer package for sending emails.
- body-parser: Middleware for parsing request bodies.
- cors: Middleware for enabling Cross-Origin Resource Sharing.
- app: Instance of the Express application.
- port: Port number (4000) on which the server listens.

Functionality:

- Sets up an Express server.
- Defines a route ('/send-email') for handling POST requests to send emails.
- Parses the request body to extract email, username, and DNA sequence.
- Sends an email containing the DNA sequence to the provided email address.
- Handles errors and returns appropriate responses to the client.

4.1.2 index.html

Attributes:

- Contains HTML structure for a web page.
- Uses JavaScript to interact with the server and update the UI dynamically.

Functionality:

- Displays the title "DNA Unique Generator."
- Provides input fields for entering username and email.
- Allows users to generate a DNA sequence by clicking the "Generate DNA Sequence" button.
- Calls the generateDNASequence() function to handle the generation and sending of DNA sequences.
- Dynamically updates the result message to inform the user about the status of the email sending process.

4.2 Interface depiction

The next picture shows the user interface where a user enters username and email to generate DNA sequence.

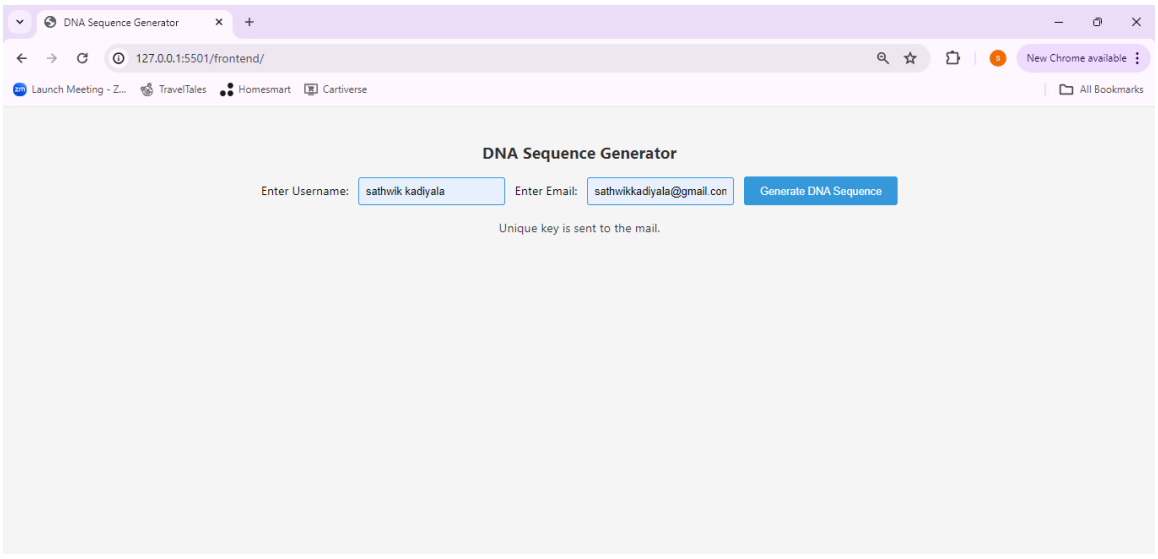

Fig 4.1 User Interface to register user in cluster

After user registration, a unique DNA sequence is generated for each user, serving as a distinctive identifier. This DNA sequence is then securely sent to the user's email address for authentication purposes. When a user attempts to submit a job to the Spark cluster, they are prompted to provide their email address along with the DNA sequence received in the email. The Spark submission process incorporates this authentication step.

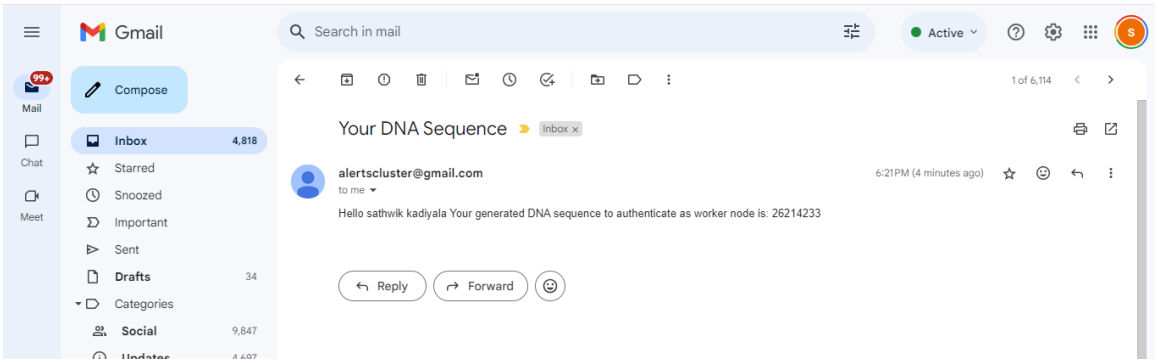

Fig 4.2 Unique Key send via mail server

The provided Python job script offers a streamlined approach to authenticate users based on their DNA sequences stored in a MongoDB database. Initially, it extracts the email and received DNA sequence from the command-line arguments. Subsequently, it establishes a connection to the MongoDB database hosted on MongoDB Atlas and queries the 'users' collection to retrieve the user document corresponding to the provided email. Upon finding the document and ensuring it contains the necessary 'dnaSequence' field, the script proceeds to authenticate the user by comparing the received DNA sequence with the stored one. If the sequences match, indicating successful authentication, the script executes the Spark job logic.

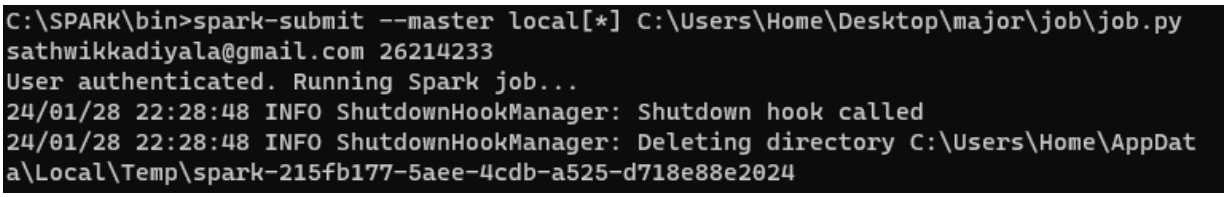

Fig 4.3 User authentication successful

Conversely, if there's a mismatch in the sequences or if the user document is not found, authentication fails, and the script exits gracefully with an appropriate message. This approach ensures a secure and efficient user authentication process, leveraging DNA sequences as unique identifiers while minimizing computational overhead.

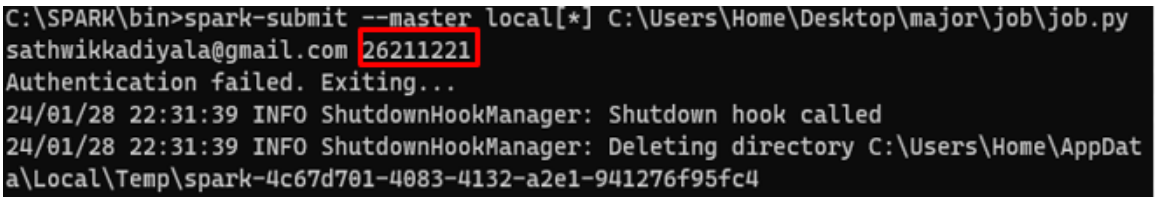

Fig 4.4 User authentication failed

5. OBSERVATIONS

5.1 Experiment setup

5.1.1 Software Requirements

- Operating System **:** Windows, Linux
- Latest version of VSCode 1.60.0.
- Node.js (latest version 18.18.0) installed. You can download it from the official website.
- "Cors" (version 2.8.5) library has been integrated to facilitate Cross-Origin Resource Sharing, enabling our experimental setup to efficiently manage and share resources between different origins.
- "ExpressJS" (version 4.18.2) library has been included to harness its capabilities for building a robust and responsive web server.
- MongoDB to act as No-SQL database to store and retrieve data.
- Latest version of Nodemailer package for sending emails. You can install it using npm.

5.1.2 Hardware Requirements

- Processor**:** Intel Core i3/i5-2350M CPU @2.30GHz.
- RAM Capacity**:** 4 GB RAM or more.
- System Type: 64-bit Operating System.

5.2 Parameter Formula

The total Spark Cluster security in existing model takes:

$$\mathbf{T(KA)=U[(6t)+x+y]}$$

T(KA) is the total time spent on the authentication process in the existing model.

The term U[(6t)+x+y] represents the computational factors involved in the authentication process, where 'U' is a function of various parameters.

So, the total Spark Cluster security time in the existing model is given by:

$$T(SC)=U[(6t)+x+y] +O(N)$$

O(N) denotes additional overhead, possibly associated with the Spark Cluster security mechanism.

So, the total Spark Cluster security time in the developed model is given by:

$$T(SEAL)=U(5) +[O(N)+D]$$

'D' represent additional factors introduced in the developed model.

6. DISCUSSION OF RESULTS

	Number of times Updation Required(Existing)	**Number of times Updation Required (Proposed)**
User1	**8**	**1**
User2	**16**	**2**
User3	**24**	**3**

Table 6.1 Metadata Security existing and proposed Mechanisms.

The above table is showing for 3 Spark salve nodes metadata information existing which is updating every 8 seconds, but the proposed security mechanism can update every 64 seconds for reducing the computational burdens.

Findings:

- Reduces update frequency to every 64 seconds.
- The proposed mechanisms demonstrate significant improvements in computational efficiency and metadata security.

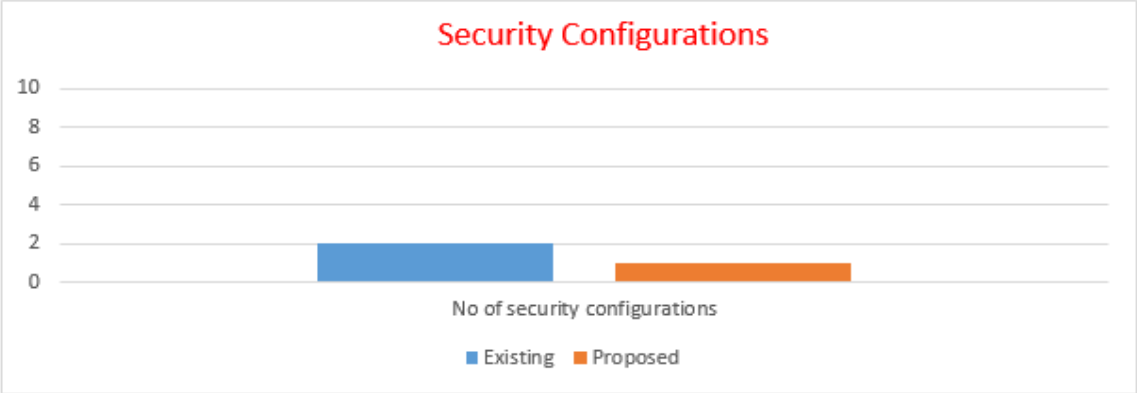

Fig 6.2 Security Configurations existing and proposed

Findings:

- Consolidates security into 1 configuration.
- Eliminates metadata security computations.

The following table showing the 10 users wants login single node cluster and the required to authentication by using Kerberos before entering the cluster. Every time they request KDC for each session which is taking 10 computation per session, each users may request authentication KDC depending on their usability of cluster.

	No.of sessions Authentications/ Day (Existing)	Total Computations	Number of Time Authenticated/ Day(Proposed)	Total Computations
User1	5	50	2	8
User2	10	100	3	12
User3	8	80	4	16
Total	23	230	9	36

Table 6.3 User Authentication computation

Table shows the user sessions and total computations of existing and proposed authentication.

7. SUMMARY

7.1 Findings

The experimental results indicate a substantial improvement in both computational efficiency and metadata security with the proposed mechanisms compared to the existing ones in the spark cluster environment.

Furthermore, the consolidation of security configurations into a single setup streamlines the user authentication process, significantly reducing the computational load required for each authentication instance.

This consolidation not only simplifies the authentication workflow but also enhances the overall system performance by minimizing redundant computations.

Overall, these findings underscore the efficacy of the proposed mechanisms in addressing key security and computational challenges inherent in spark clusters. By optimizing authentication processes and metadata management, the proposed approach offers a robust and efficient solution for securing big data environments while minimizing computational overhead, ultimately enhancing system performance and user satisfaction.

7.2 Justification

Parameter	**Previous methods**	**Proposed method**
Metadata Security	Requires metadata updates every 8 seconds	Significantly reduces computational burdens.
Security Configurations	Involves 2 security configurations.	Consolidates security into 1 configuration.
User Authentication	Total computations for authentication reach 230 for 3 users.	Total computations for authentication are reduced by 84% for 3 users.

Table 7.1 Parameter Comparison

Parameter Formulas:

The current spark cluster security mechanism consumes

$$T(SC)=U[(6t)+x+y] +O(N)$$

The developed Spark Cluster security mechanism consumes

$$T(SAI)=U(5) +[O(N)+D]$$

Parameters improved:

Improves security by stopping and exiting spark job in the spark cluster thereby enhancing **cluster security**, **data integrity** and **less computational** user authentication.

This is achieved through unique key generation methods, real-time monitoring, alerting mechanisms, and seamless integration with cluster management tools. These enhancements make the system highly effective in preventing node duplication, ensuring a secure and reliable cluster environment.

7.3 Conclusion

A Secure E-Authentication Layer (SEAL) as a secure layer which positioned above the Spark Cluster. This mechanism provides better authentication for the spark cluster by using limited computations and providing better user security from hackers. The main objective is to design layer to implement node-sensitive data hiding using DNA sequences and provide security to the user and its data from hackers.

8. REFERENCES

[1] Nan Zhang, Xiaoyu Wu, Cheng Yang, Yinghua Shen and Yingye Cheng,"A lightweight authentication and authorization solution based on Kerberos," 2016 IEEE Advanced Information Management, Communicates,Electronic and Automation Control Conference (IMCEC), Xi'an, 2016, pp.742-746,DOI:10.1109/IMCEC.2016. 7867308.

[2] Yoon-Su Jeong & Yong-Tae Kim "A token-based authentication security scheme for Hadoop distributed file system using elliptic curve cryptography" August 2015Journal of Computer Virology and Hacking Techniques 11(3) DOI:10.1007/s11416-014-0236-5

[3] Paul j. e. Velthuis, "New authentication mechanism using certificates for big data analytic tools", kth royal institute of technology school of information and communication https://kth.divaportal.org/smash/get/diva2:1149007/FULLTEXT01.pdf

[4] Lee, S., Lee, E., Hwang, W. et al. Reversible DNA data hiding using multiple difference expansions for DNA authentication and storage. Multimed Tools Appl 77, 19499–19526 (2018). DOI:10.1007/s11042-017-5379-1

[5]Balaraju, J., & Prasada Rao, P. V. R. D. (2020b). Investigation and Finding A DNA Cryptography Layer for Securing Data in Hadoop Cluster. Int. J. Advance Soft Compu. Appl, 12, 3.

[6] A. Vikram, S. Kalaivani and G. Gopinath, "A Novel Encryption Algorithm based on DNA Cryptography," 2019 International Conference on Communication and Electronics Systems (ICCES), Coimbatore, India, 2019, pp. 1004-1009, doi: 10.1109/ICCES45898. 2019.9002399.