Enhancing Cloud Security: Implementing and Evaluating the Zero Trust Architecture with Firebase Services and Advanced Encryption Algorithms

Academic Internship
MSc in Cyber Security

Manohar Babu
x21239631

School of Computing
National College of Ireland

Supervisor: Apurva Vangujar

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name:	Manohar Babu		
Student ID:	X2123963		
Programme:	MSc Cyber Security	**Year:**	2023 - 2024
Module:	Academic Internship		
Supervisor:	Apurva Vangujar		
Submission Due Date:	31/01/2024		
Project Title:	Enhancing cloud security: Implementing and evaluating the Zero Trust Architecture with Firebase Services and advanced encryption algorithms		
Word Count:7120		**Page Count:** 20	

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: Manohar Babu

Date: 31/01/2024

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Enhancing Cloud Security: Implementing and Evaluating the Zero Trust Architecture with Firebase Services and Advanced Encryption Algorithms

Manohar Babu
x21239631

Abstract

To enhance the cloud security developing a web-based user management system integrating authentication while implementing the Zero-trust model. The implementation leveraged tools like Firebase for authentication and data handling, hosted on AWS servers. The project addressed prevalent concerns in modern cybersecurity, emphasizing the need for robust access control and data security in web applications. This model assumes no implicit trust, validating every request explicitly and enforcing the principle of least privilege. The project's primary contribution lay in its holistic approach to user management, incorporating Firebase tools for authentication, custom claims, and real-time database handling. The system effectively validated users through Firebase authentication and secured data handling via Firebase's real-time database with meticulous security rules. The findings align with the current state of the art in cybersecurity by showcasing an application of the Zero-trust model within a web-based user management system.

Cloud storage is becoming a dependable method of storing data, there are still significant security concerns that affect both cloud computing and cloud storage, including maintaining confidentiality integrity as well as confidentiality. The integration of advanced encryption algorithms, namely Double AES (DAES) and Blowfish, within cloud storage systems, particularly focusing on Amazon S3, to bolster data security. The research examines the implementation of these robust encryption methods and evaluates their impact on data transfer efficiency and system performance. The investigation involves encrypting files using DAES and Blowfish before uploading them to an Amazon S3 bucket. Through empirical analysis, this study assesses the encryption time, throughput, and overall performance implications of integrating these encryption algorithms into cloud storage.

Keywords: Zero trust architecture, encryption, multi factor authentication, Blowfish, Double Advance Encryption Standard, AWS, Firebase authenticator

1 Introduction

In the contemporary landscape of cloud computing, where the traditional perimeter-based security model proves increasingly inadequate against sophisticated cyber threats, the paradigm of Zero Trust Architecture (ZTA) has emerged as a compelling approach. Unlike conventional security models that inherently trust users and devices within a defined network perimeter, Zero Trust challenges the assumption of trust advocating for continuous verification and strict access controls. As organizations transition critical operations to cloud environments, the need for robust security measures within this dynamic and distributed landscape becomes paramount. Cloud environments are increasingly vulnerable to security breaches, data leaks, and unauthorized access. With the growing volume of sensitive data being stored and transmitted, there is a critical need for robust security measures to ensure secure data storage. Integrating advanced encryption algorithms mechanisms is a promising approach to addressing these challenges. However, the effective integration of these techniques in cloud and IoT environments requires thorough investigation and analysis.

1.1 Background

Historically, enhancing cloud security involves integrating robust measures like Zero Trust Architecture and advanced authentication mechanisms while understanding the role of encryption in safeguarding data within cloud infrastructures. As suggested by (Stafford, 2020) the Zero Trust Model is significantly relevant to the discussion of increasing user authentication and overall cyber security measure in web applications. The Zero-trust security framework operates on the fundamental principle of ‘Never trust, and always verify’. The key motivation for this research has emerged from the findings of the Verizon Data Breach Investigation Report or DBIR (Verizon, 2021) which significantly shows the persistent threat posed by credentials that are compromised. Combining these measures aligns seamlessly with ZTA's principles, fortifying security against evolving cyber threats. Encryption serves as the cornerstone of digital security, evolving from ancient ciphers to modern algorithms. Symmetric key encryption, like Advanced Encryption Standard (AES) and Data Encryption Standard (DES) uses the same key for encryption and decryption, while asymmetric key encryption, including Rivest-Shamir-Adleman (RSA) and Elliptic Curve Cryptography (ECC), employs a key pair for secure communication. Encryption's strength lies in key length, algorithm complexity, and its ability to adapt to combat evolving threats. When applied to the cloud, encryption ensures sensitive data remains unreadable without proper decryption keys, mitigating risks associated with cloud breaches or unauthorized access. Therefore, enhancing cloud security involves the strategic integration of multifactor authentication, IP address restrictions, and robust encryption measures like AES, RSA, or ECC. These collectively fortify cloud infrastructure, aligning with the core principles of ZTA, and offer comprehensive protection against modern cyber threats.

2 Related Work

2.1 Related work based on Zero Trust Model

The paper (Ometov et al. 2018) provides a comprehensive review of the escalating security concerns in cloud computing, emphasizing the critical need for trust management due to issues like identity theft, data breaches, and data integrity compromises. It rightly highlights the limitations of traditional trust mechanisms in meeting the dynamic demands of cloud services and proposes a conceptual zero-trust strategy tailored for the cloud environment. While the paper effectively contextualizes the importance of trust establishment and explores the challenges faced in cloud computing, it primarily remains theoretical, lacking empirical validation or in-depth exploration of trust challenges. The proposal of a zero-trust model is innovative, but its practical applicability remains untested, calling for empirical studies or real-world case analyses to fortify its theoretical foundations and enhance its credibility.

Research was published by (Basavala et al. 2012), that investigated the integration of different types of user authentication with web and mobile applications. As per the article, authentication is the procedure of establishing an identity for application end users to authorize the application or access system. Authentication comprises different types that may be utilized depending on the level and appropriateness of the authentication or security being utilized in the web application. In general, authentication types may consist of something the user knows, something the user has, and something the user is. The type of authentication is found to depend on its factors, if it is more than one factor then it is significantly more complicated to compromise than methods of a single factor.

According to (Velásquez et al. 2018), single-factor authentication or SFA or conventional authentication is considered a conventional security requirement that demands a username and password to log into a mobile or web application or system. SFA security usually relies specifically on the diligence of the end user. A user needs to take additional precautions and ensure that no one can access it by making a guess. For every mobile and web application that requires sufficient security, it may be suggestible to implement more complex systems, for example, Multifactor Authentication or MFA. On the other hand, two-factor authentication is considered a security approach in which an end user supplies or provides two forms of identification, one of which is usually a physical token for example, a key generator or smart card, and the other of which is usually something memorized, for example, a security code and a PIN.

As suggested by (Reese et al. 2019), a two-factor authentication system does not necessarily provide a strong guarantee for user authentication, nor does it essentially prohibit online fraud. However, a two-factor authentication is considered significantly essential to reduce the prevalence of online attacks and frauds as suggested by the researchers. Both MFA and OTP systems offer increased security compared to basic two-factor authentication, significantly reducing the susceptibility to various online attacks and fraudulent activities. The adoption of these advanced authentication methods reflects a proactive approach in mitigating security risks, although their effectiveness also depends on implementation quality and user adherence to security best practices.

Difference Between Zero Trust Model and Traditional Model Implementation

In the realm of cybersecurity, the Zero Trust model has gained significant attention and adoption in recent years as organizations seek more robust ways to protect their digital assets (Muhammad et al., 2022). This approach differs markedly from the traditional security model that has been prevalent for decades.

Assumption of Trust: Historically, the traditional security model has operated on the principle of perimeter-based security. It assumes that threats are external, and once inside the network, entities are considered trusted until proven otherwise. This approach relies on firewalls and VPNs to protect the network boundary. Whereas the Zero Trust model challenges the notion of trust and operates on the assumption that threats can originate from both external and internal sources (Talan, 2022). It does not automatically trust any user or device, regardless of their location within or outside the network. Trust is never assumed and must be continuously verified.

Network Architecture: Traditional security models often involve the use of a fixed network perimeter. Access controls are implemented at the network edge, with the belief that anything inside the perimeter is safe. This model often relies on VPNs for remote access. In contrast, Zero Trust embraces a network architecture that is decentralized and micro-segmented (Uwaoma, 2023). It uses the concept of "micro-perimeters" to create isolated zones and enforces strict access controls at every level of the network. Network segmentation is a fundamental principle in Zero Trust.

Identity-Centric Security: In the traditional model, security largely relies on IP addresses and network location. It often focuses on securing the network itself rather than the identities and devices that access it. But Zero Trust is fundamentally identity-centric (Mir and Ram Kumar, 2021). It places a strong emphasis on authenticating and verifying the identity of users and devices before granting access to any resource. It uses techniques such as multi-factor authentication (MFA) and continuous monitoring to validate identities (Alappat, 2023).

Access Control: Traditional security models tend to use a relatively coarse-grained access control approach. Once users are inside the network, they often have broad access to many resources. In contrast, Zero Trust implements fine-grained access control. Access is restricted to the minimum necessary for a user or device to perform their job. Least privilege access is a core principle of the Zero Trust model.

Threat Detection: The traditional model often relies on perimeter-based security appliances like firewalls and intrusion detection systems to detect and respond to threats. Internal threats can be harder to detect (Abwnawar, 2020). However, Zero Trust places a strong emphasis on continuous monitoring and threat detection, both for external and internal threats. Behavior-based analytics and machine learning are commonly used to detect anomalies and potential threats (Singh et al., 2023).

Trust in Devices: Traditional models may implicitly trust devices once they are on the network, assuming that they are secure. This trust is often extended to all devices within the network. Whereas Zero Trust includes device trust assessment as part of its core principles (Li et al., 2022). Devices must meet specific security criteria and be continuously evaluated for compliance to gain and maintain access.

2.2 Related works based on Algorithms

The paper (Thabit, et al., 2023) highlights the escalating need for secure data handling in cloud computing due to the exponential growth of data. It introduces a novel lightweight cryptographic algorithm designed to bolster data security within cloud applications, drawing inspiration from established encryption methods such as Feistel and substitution-permutation architectures. The algorithm's emphasis on flexibility in key length and turns, along with its integration of logical operations to achieve Shannon's diffusion and confusion theory, showcases an innovative approach. However, the paper lacks comprehensive validation and comparative analysis against established encryption systems widely used in cloud computing. Additionally, the absence of real-world application testing diminishes the assessment of its practical effectiveness. To strengthen its credibility, the paper could benefit from rigorous validation against existing systems and real-world scenario testing to showcase its applicability and performance in cloud environments.

This research aims (Ghosh, et al., 2015) a significant task in evaluating six prominent encryption algorithms AES (Rijndael), DES, 3DES, RC2, Blowfish, and RC6 highlighting their role in securing internet and network applications. Recognizing the increasing demand for information security and acknowledging the resource-intensive nature of encryption algorithms, the study conducts a comprehensive comparison across various settings. It assesses different aspects such as data block sizes, data types, key sizes, battery power consumption, and encryption/decryption speeds, presenting simulation results to demonstrate the efficacy of each algorithm. While the paper addresses crucial aspects of resource consumption and performance, its strength lies in providing a broad comparative analysis. However, deeper insights into specific strengths and weaknesses of each algorithm and the potential inclusion of more encryption methods could further enrich its findings and ensure a more comprehensive evaluation of encryption technologies for securing digital applications.

The paper presents (R.Udendhran, 2017) a crucial response to the limitations of single-factor authentication and the vulnerabilities of conventional encryption algorithms by proposing a novel hybrid approach. Combining AES, DES, and RSA within a Feistel structure showcases an innovative amalgamation of symmetric and asymmetric encryption methods, aiming to bolster security for sensitive data transmission. However, the paper lacks depth in explaining the specific implementation details of this hybrid algorithm, which might impede its clarity and reproducibility. While emphasizing parameters like avalanche effect, encryption time, CPU usage, and throughput for evaluation, the absence of a comparative analysis against existing encryption standards raises questions regarding the proposed algorithm's superiority. To enhance the paper's credibility, a more comprehensive methodology description and a comparative analysis against established encryption standards are recommended to validate the effectiveness and distinct advantages of the proposed hybrid encryption approach in fortifying security and performance in data transmission.

The paper (Abdul, et al., 2008) delves into the significant security challenges prevailing in cloud storage and computing, emphasizing the crucial aspects of confidentiality, integrity, and privacy. It aptly recognizes the limitations of existing encryption techniques like AES, highlighting the need for more efficient means to ensure data security. Introducing homomorphic encryption as a solution capable of addressing both confidentiality and integrity concerns aligns with established literature acknowledging its potential in securing sensitive data within cloud environments. The novel proposal of combining homomorphic encryption with location-based decryption presents an innovative approach to bolster data security in cloud storage, leveraging the strengths of both techniques. However, the paper lacks depth in explaining the practical implementation details of this combined framework, which may hinder its clarity and reproducibility. Moreover, empirical validation or experimental results are missing, impeding the demonstration of the proposed framework's effectiveness in real-world cloud storage scenarios. Strengthening the implementation description and validating the framework through empirical studies would enhance the paper's credibility and showcase its practical applicability in fortifying data security within cloud storage systems.

The paper (Olanrewaju, et al., 2018) rightly underscores the criticality of cloud computing in handling diverse user needs and the imperative nature of data security, accessibility, and reliability. Acknowledging the robustness of the AES algorithm in file encryption and the potential threats like brute force and algebraic attacks signifies a comprehensive understanding of evolving cybersecurity challenges. However, the proposal to introduce a hybrid structure of Double Advance Encryption Standard (DAES) and Blowfish algorithms lacks detailed methodology, hindering a clear understanding of its implementation and potential effectiveness. Additionally, the absence of empirical validation or experimental results detracts from substantiating the proposed hybrid encryption method's practical application and effectiveness in real-world cloud storage environments. Strengthening the paper with a detailed implementation description and empirical validation would significantly enhance its credibility and demonstrate the practicality of the proposed solution in fortifying data security within cloud storage systems.

In response to escalating security threats in cloud computing, authentication mechanisms stand as frontline defenses against identity-related risks. This literature review highlights the evolution from conventional single-factor authentication (SFA) to multifactor authentication (MFA) systems, emphasizing the imperative of more robust identity verification in mitigating threats within cloud environments. Also, it's evident that various papers explore enhancing data security within cloud storage through innovative encryption methodologies. Particularly,there's a consistent emphasis on the need for advanced encryption algorithms to fortify security measures. Given the attention to algorithms like Blowfish and DAES, our research question could focus on their integration into cloud storage systems to elevate data security (Olanrewaju, et al., 2018).

Research Questions

1. How does the implementation of Zero Trust Architecture principles, incorporating Firebase services for authentication, access control, and secure data handling, impact the overall security and user experience of a web-based user management system?
2. How can advanced encryption algorithms like DAES and Blowfish be integrated into cloud storage to enhance data security?

3 Research Methodology

The research methodology employed in this study adopts a systematic approach to ensure the validity and reliability of the project. The primary focus is on user authentication within a system, emphasizing the steps taken, the materials and equipment methods, data measurements, and statistical techniques applied. Steps Followed in the Research methodology is as:

1. User Registration:
   - Participants were required to enter their credentials, including email, username, and password during the registration process.
   - Users were prompted to select their user type (e.g., user or admin).

2. Email Verification:
   - After registration, an email verification link was sent to the provided email address.
   - This step ensured the authenticity and security of both admin and user accounts.

3. Admin Dashboard Access:
   - Admins, upon successful registration and authentication, were redirected to the admin dashboard.
   - The dashboard provided specific functionalities tailored to administrative tasks.

4. Phone Number Verification:
   - Upon successful email verification, users were redirected to the phone number verification page.
   - Users entered their phone number, and a One-Time Password (OTP) was generated and sent to their mobile number for additional authentication.

5. System Performance Evaluation:
   - A user-centered evaluation method was integrated to gauge the usability and user satisfaction of the system.
   - Data points were collected during user and admin registration, login, and verification procedures.

6. User-Centered Evaluation Tools:
   - User experience and satisfaction were assessed using relevant tools and methodologies.

7. User Authentication Metrics:
   - Metrics such as successful email verifications, successful phone number verifications, and OTP generation times were measured.

This research methodology ensures an exploration of user authentication processes within a systematic framework, combining qualitative and quantitative data to assess system performance, user satisfaction, and security vulnerabilities. The systematic approach in data collection and analysis contributes to the reliability and validity of the study's findings.

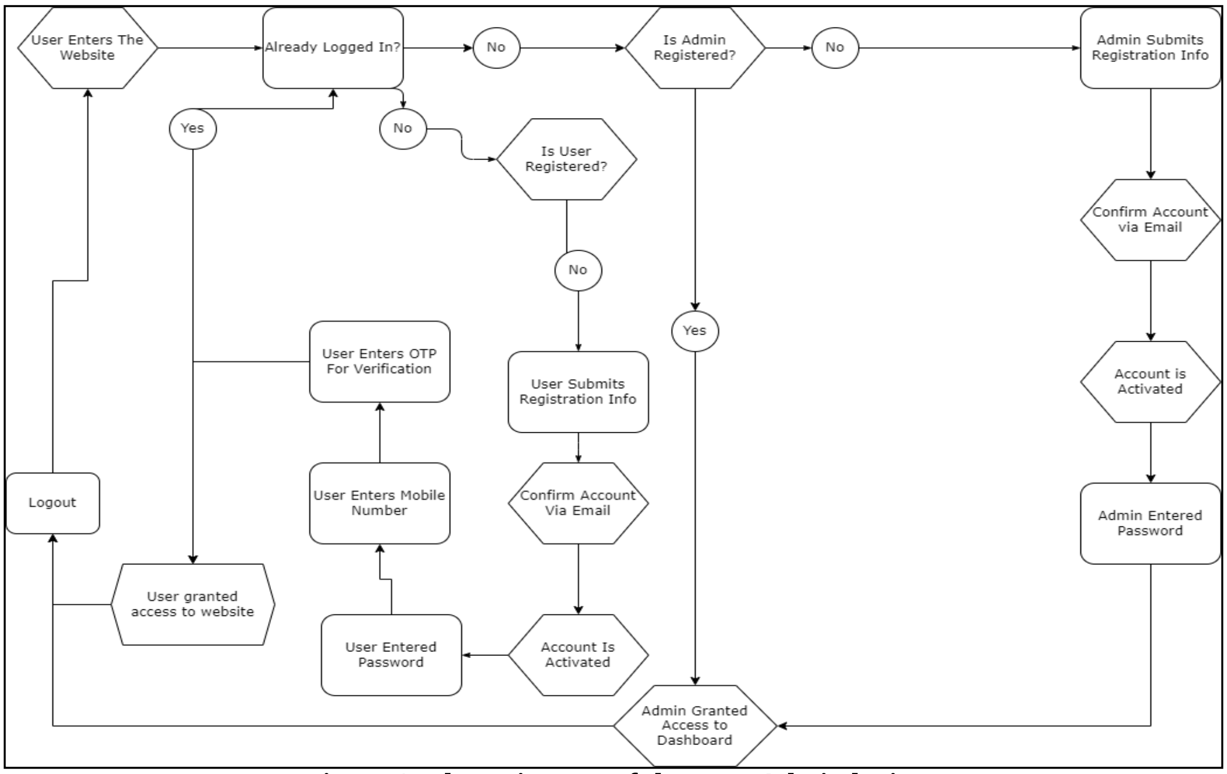

User Enters The Website
Already Logged In?
No
Is Admin Registered?
No
Admin Submits Registration Info
Yes
No
Is User Registered?
Confirm Account via Email
No
Yes
User Enters OTP For Verification
User Submits Registration Info
Account is Activated
Logout
User Enters Mobile Number
Confirm Account Via Email
Admin Entered Password
User granted access to website
User Entered Password
Account Is Activated
Admin Granted Access to Dashboard

Figure 1: Flow Diagram of the User/Admin login

3.2 Methodology for Integrating Hybrid Algorithms

Key Generation:

- Password-based key derivation function (`PBKDF2`) is used to derive keys for encryption.
- Keys are generated separately for Blowfish and DAES algorithms.

Encryption:

- Blowfish and DAES encryption algorithms are employed.
- The `encrypt_blowfish` function encrypts data using Blowfish in Electronic Codebook (ECB) mode.
- The `encrypt_aes` function encrypts data using AES in streaming (ARC4) mode.
- Hybrid encryption combines Blowfish and DAES encrypted outputs.

File Encryption and AWS S3 Upload:

- Data is read from a file.
- Salts are generated for both Blowfish and DAES.
- Encryption time and throughput are measured for each algorithm and the hybrid method.
- Encrypted data is uploaded to an AWS S3 bucket with distinct object keys for Blowfish and DAES.

Time and Throughput Measurement:

- The `measure_time_and_throughput` function captures encryption time and throughput.
- Time is measured using the `time` library, and throughput is calculated based on data size and elapsed time.

Security Considerations:

- Strong and unique passwords are encouraged.
- AWS credentials should be kept secure.
- Error handling and additional security measures should be considered based on specific use cases.

This methodology provides an overview of how the script handles key generation, encryption, measurement, AWS S3 interaction, and visualization. Adjustments can be made based on specific requirements and security considerations.

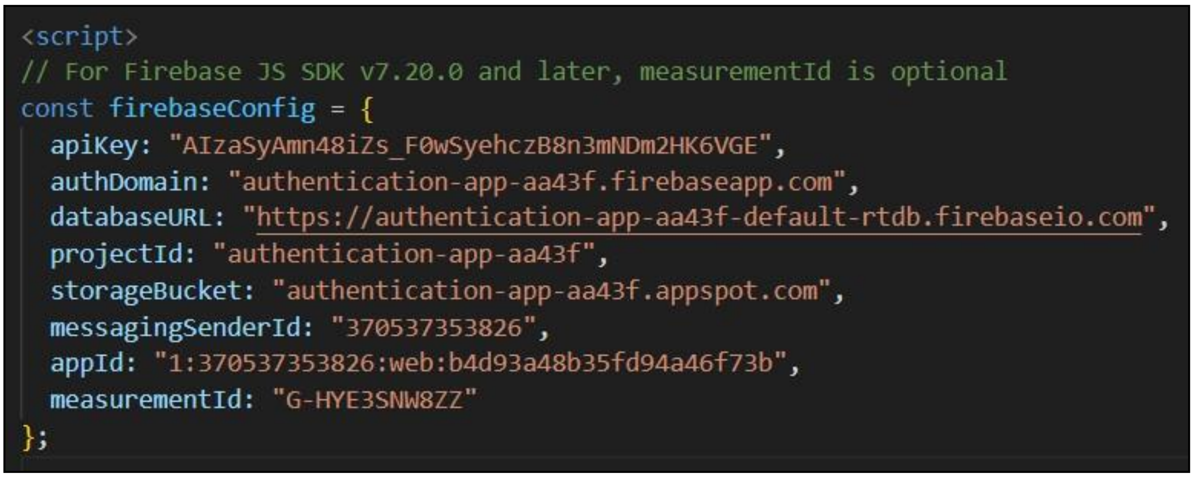
4 Design Specification

4.1 Design

The HTML and JavaScript code represents the web page for a dashboard with user authentication using Firebase. The implementation relies on HTML for structuring the page, CSS for styling, and JavaScript for Firebase authentication and data retrieval. Below is an overview of the design and implementation along with the identified techniques and requirements.

4.2 Structure Overview:

- HTML Structure: The HTML file defines the structure of the dashboard page, including placeholders for the welcome text, a data table, and a logout link.
- CSS Styling: The CSS style rules enhance the visual appeal of the dashboard, providing a clean and responsive design. It includes styles for the body, main content, table, and logout link.
- Firebase Configuration: Firebase is configured in the JavaScript module using the provided API key, authentication, and database settings.
- Firebase Authentication: Firebase authentication functions, such as `onAuthStateChanged`, are used to check if a user is authenticated. If authenticated, the user's data is retrieved from the Firebase Realtime Database.



```
<script>
// For Firebase JS SDK v7.20.0 and later, measurementId is optional
const firebaseConfig = {
  apiKey: "AIzaSyAmn48iZs_F0wSyehczB8n3mNDm2HK6VGE",
  authDomain: "authentication-app-aa43f.firebaseapp.com",
  databaseURL: "https://authentication-app-aa43f-default-rtdb.firebaseio.com",
  projectId: "authentication-app-aa43f",
  storageBucket: "authentication-app-aa43f.appspot.com",
  messagingSenderId: "370537353826",
  appId: "1:370537353826:web:b4d93a48b35fd94a46f73b",
  measurementId: "G-HYE3SNW8ZZ"
};
```

Figure 2: Code snippet of configuration settings required to connect the web application to specific firebase services.

4.3 Identified Techniques and Requirements

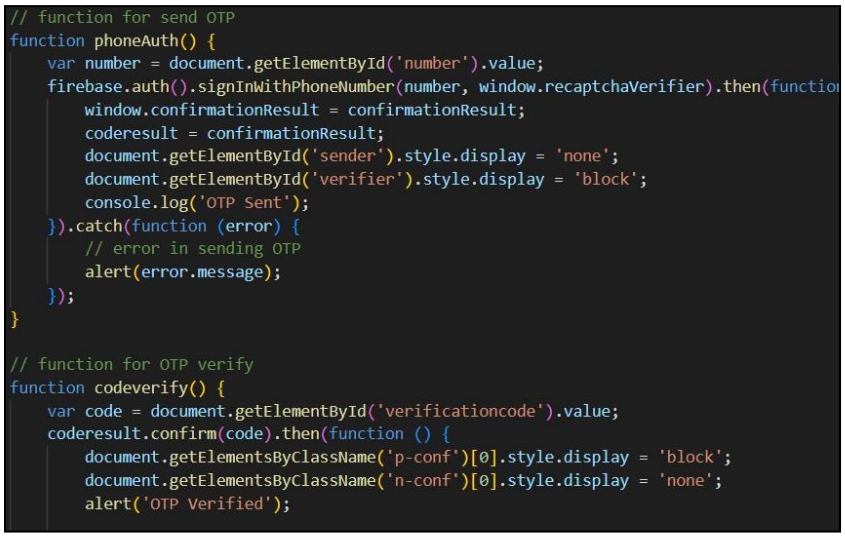
- Firebase Authentication: Utilizes Firebase Authentication to manage user sign-in, sign-out, and user state changes.
- Firebase Realtime Database: Interacts with the Firebase Realtime Database to retrieve user data and determine user type.
- Responsive Design: Implements responsive design principles using CSS for a visually appealing and user-friendly dashboard.
- User Type Verification: Checks the user type (admin or regular user) to customize the dashboard's content based on user roles.
- Security Considerations: Firebase Authentication and Realtime Database interactions are secure methods for user authentication and data storage.

4.4 Proposed Model:

The code focuses on the integration of Firebase for user authentication and data retrieval. The flow revolves around checking user authentication, retrieving user data, and customizing the dashboard based on user roles.

1. Requirements:
    - Firebase Account: Requires a Firebase account and project setup with the provided API key, authentication, and database configurations.
    - Firebase JavaScript SDK: Includes the Firebase JavaScript SDK from the specified CDN links.
    - Web Browser Compatibility: Designed to work on modern web browsers with JavaScript and CSS support.
    - Firebase Realtime Database Rules: Requires appropriate Firebase Realtime Database rules to control access.

The verification.html code is for an OTP (One-Time Password) verification form. Users are required to enter a 4-digit OTP, and upon clicking the "Verify OTP" button, the script redirects them to the "home.html" page.



```
// function for send OTP
function phoneAuth() {
    var number = document.getElementById('number').value;
    firebase.auth().signInWithPhoneNumber(number, window.recaptchaVerifier).then(functio
        window.confirmationResult = confirmationResult;
        coderesult = confirmationResult;
        document.getElementById('sender').style.display = 'none';
        document.getElementById('verifier').style.display = 'block';
        console.log('OTP Sent');
    }).catch(function (error) {
        // error in sending OTP
        alert(error.message);
    });
}

// function for OTP verify
function codeverify() {
    var code = document.getElementById('verificationcode').value;
    coderesult.confirm(code).then(function () {
        document.getElementsByClassName('p-conf')[0].style.display = 'block';
        document.getElementsByClassName('n-conf')[0].style.display = 'none';
        alert('OTP Verified');
```

Figure 3: Code Snippet of OTP verification page

4.5 Hybrid encryption algorithm

The concept of hybrid encryption leveraging both symmetric (like Blowfish) and asymmetric (like Dynamic AES or DAES) encryption algorithms to enhance security while maintaining efficiency in data transmission. This approach (Abdul, et al., 2008) offers the advantages of both types of encryptions: the robust security of symmetric encryption and the convenience of asymmetric encryption. Combining these encryption methods allows for a powerful approach to secure data transmission, ensuring both speed and security. Symmetric encryption provides a strong level of security, while asymmetric encryption adds the convenience of key exchange without requiring direct sharing of secret keys. The utilization of multiple cryptographic approaches addresses potential security flaws, ensuring comprehensive protection for sensitive data. This method aims to fulfill crucial security goals by effectively managing cryptographic keys and ensuring the reliability of the cryptographic technique in handling substantial volumes of transmitted data. By employing a hybrid encryption technique that incorporates DAES and Blowfish algorithms it leverages their respective strengths to create a robust and efficient security mechanism.

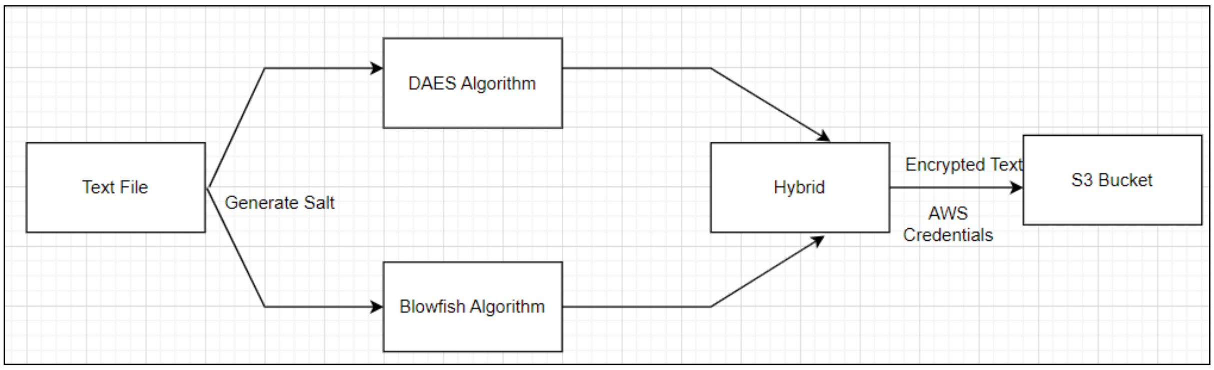


Figure 2: Flow Diagram of file encryption

5 Implementation

5.1 Website User/Admin Registration and Authentication System

1. User Registration: A new user can be created using the signup on the homepage and can be registered as a user or admin. Necessary information like name, email and password are needed and an alert message upon successful registration is generated.
2. User Login: A user/admin can be logged in using login Page (login.html). An error message is displayed for incorrect information. If logged in as an admin, redirect to the "Dashboard.html"; if logged in as a user, redirect to the "Home.html".
3. Two-Step Authentication:
   - Registration Flow: Upon User/Admin provides necessary information. Anemail verification link is sent to the registered email address. User checks emailand clicks on the verification link.
   - Firebase Database: Store registered user data in Firebase Database.
   - Mobile Number Verification: After logging in, the user must enter their mobile number with the country code. An OTP is sent to the provided number. Entering the correct OTP verifies the user.

4. Redirect Pages: Redirected to regular users and admin users after login.
5. Logout Functionality: Included a logout button to redirect the user to the signup/login page.
6. Specific IP Addresses: On Amazon EC2 Dashboard inbound Rules are edited to allow access permissions to specific IP addresses. Only permitted IP addresses will be allowed to connect to the webpage.

5.2 Integrating Hybrid encryption Algorithms

1. The Python script is designed to perform file encryption using the DAES and Blowfish encryption algorithms, as well as a hybrid method combining both algorithms. Required libraries/modules for encryption (cryptography, Blowfish from Crypto.Cipher, base64, boto3, os, time, getpass).
2. The key generation process involves using the password-based key derivation function (`PBKDF2`). generate_key_pbkdf2 creates a key specifically tailored for Blowfish encryption by utilizing PBKDF2HMAC with SHA256. generate_key_aes generates a key suitable for AES encryption through PBKDF2HMAC with SHA256.
3. The `hybrid_encrypt` function combines the outputs of DAES and Blowfish encryption. It generates keys for both algorithms using the provided password and salts, encrypts the data using both algorithms, and combines the results to create hybrid encryption.
4. The script includes a function for uploading the encrypted data to an AWS S3 bucket (`upload_to_s3`). It utilizes the `boto3` library to interact with AWS services, allowing for secure storage and retrieval of encrypted data.
5. For performance evaluation, the script includes a function (`measure_time_and_throughput`) that measures the encryption time and throughput for a given encryption function and its input parameters. This function is used to capture performance metrics for the hybrid encryption method.
6. The `encrypt_file` function orchestrates the entire process. It reads data from a specified file, generates random salts for DAES and Blowfish, measures the encryption time and throughput for each encryption method, uploads the encrypted data to an AWS S3 bucket with unique object keys, and prints relevant information. Additionally, the script configures AWS credentials using `boto3.setup_default_session`.

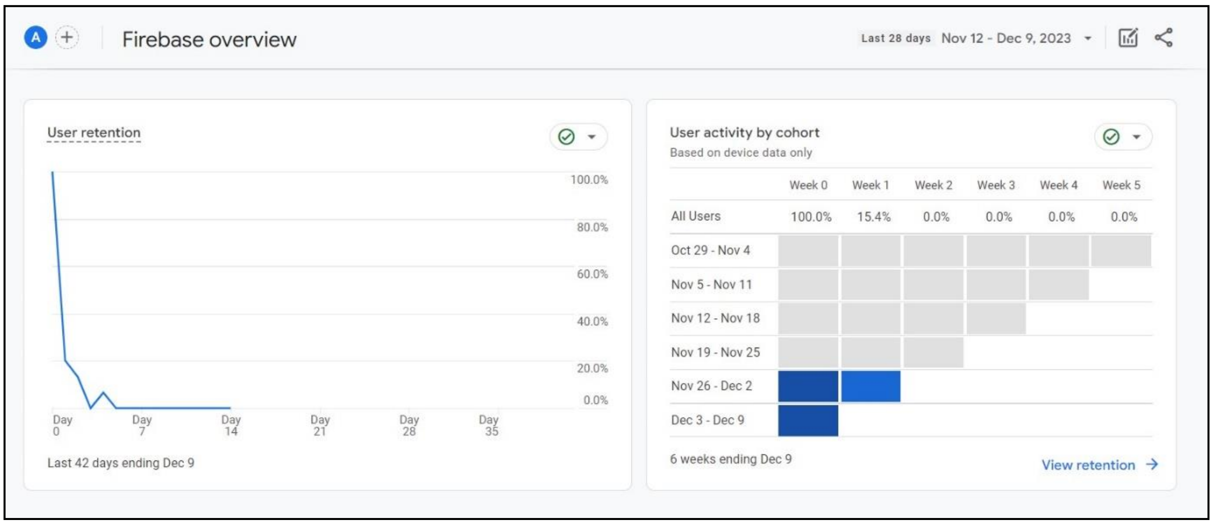
Finally, the script demonstrates an example usage where a file path, password, S3 bucket name, and object key are specified, and the `encrypt_file` function is called with these parameters. We have replaced them with our AWS credentials, file path, password, bucket name, and object key with the actual values to customize the script for the specific use case.

6 Evaluation

Firebase Authentication provides a user-friendly interface for managing user authentication, allowing easy configuration of sign-in methods, user roles, and permissions. The integration between Firebase Authentication and AWS can streamline user management, allowing your application to leverage Firebase's authentication while utilizing AWS services for other backend functionalities. The evaluation phase focused on testing and performance metrics. Testing strategies were employed to assess the robustness of the system, with results influencing iterative changes. Performance metrics, including response time and scalability, were defined and measured against established criteria. The system's limitations were acknowledged, providing a transparent overview of encountered challenges.

As a web application, the project exhibited seamless integration of Firebase authentication and dynamic functionality. The analytics dashboard in Firebase Authentication is a powerful tool providing a consolidated view of user authentication activities within your app. It offers insights into user sign-ups, sign-ins, and authentication methods used, allowing for a deeper understanding of user behavior. This dashboard also tracks performance metrics like authentication latency and error rates, aiding in the optimization of the authentication process. Security monitoring becomes efficient as it detects suspicious activities or unusual login attempts, bolstering app security. Customized reports based on user demographics or geographic data enable targeted marketing strategies or feature enhancements. By integrating with other Firebase services, this dashboard offers a holistic perspective on user behavior, aiding in data-driven decision-making for future updates and improvements. Overall, it's a valuable tool for small businesses, providing actionable insights to refine user experiences and bolster app security.

- User Authentication Effectiveness: Evaluate the success rates of user authentication methods implemented (email verification, phone number verification, OTP generation). Assess how often these methods prevent unauthorized access and maintain system security.
- System Performance Metrics: Quantitatively assess system performance by analyzing metrics like verification times, error rates during authentication, and system downtime. This provides insights into the efficiency and reliability of the authentication processes.


Firebase overview

Last 28 days Nov 12 - Dec 9, 2023

User retention

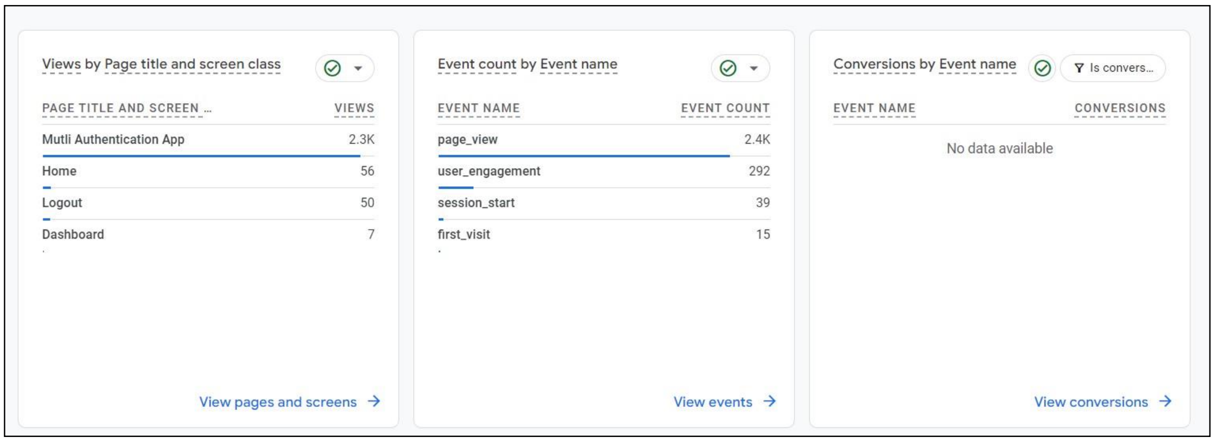
Last 42 days ending Dec 9

User activity by cohort
Based on device data only

	Week 0	Week 1	Week 2	Week 3	Week 4	Week 5
All Users	100.0%	15.4%	0.0%	0.0%	0.0%	0.0%
Oct 29 - Nov 4	▨	▨	▨	▨	▨	▨
Nov 5 - Nov 11	▨	▨	▨	▨	▨	
Nov 12 - Nov 18	▨	▨	▨	▨		
Nov 19 - Nov 25	▨	▨	▨			
Nov 26 - Dec 2	🟦	🟦				
Dec 3 - Dec 9	🟦					

6 weeks ending Dec 9

View retention →


Views by Page title and screen class

PAGE TITLE AND SCREEN ...	VIEWS
Mutli Authentication App	2.3K
Home	56
Logout	50
Dashboard	7

View pages and screens →

Event count by Event name

EVENT NAME	EVENT COUNT
page_view	2.4K
user_engagement	292
session_start	39
first_visit	15

View events →

Conversions by Event name

Is convers...

EVENT NAME	CONVERSIONS

No data available

View conversions →

Figure 3: Firebase Analytics Dashboard

Limitations:

It is essential to acknowledge the limitations of this study. The research focuses on multifactor authentication and IP address restrictions within the broader context of Zero Trust Architecture, and as such, it may not comprehensively address all aspects of cloud security. Additionally, the rapidly evolving nature of cybersecurity and technology introduces inherent limitations in capturing real-time characteristics. The two major limitations are discussed below:

- Integrating Firebase Authentication with AWS IAM introduces challenges due to the differing nature of role definitions and access control mechanisms. Firebase roles, designed for Firebase services, often lack a direct correspondence to AWS IAM policies governing access to AWS resources. Aligning these roles demands manual mapping, susceptible to inconsistencies and complexities arising from varied granularities and functionalities between the platforms. This mismatch requires custom authorization logic, regular audits, and potential automation to bridge the gap effectively.
- Although Firebase roles offer basic differences between user and admin roles, complex access logic implementation requires customized modifications on top of these roles. Aligning Firebase roles with the complex access restrictions required by AWS services which sometimes use more detailed IAM policies presents a difficulty. To bridge this gap, one must have an in-depth understanding of both platforms and be able to

transform Firebase role descriptions into fine-grained AWS permissions through complex mappings or extra logic layers.

- Integration Challenges: Integrating RBAC models from different cloud providers, each with its own identity and access management systems, can be challenging. Ensuring compatibility and synchronization of RBAC policies and user attributes across these systems is crucial.

Combining AES (DAES) and Blowfish in a hybrid encryption scheme forms a robust, multi-layered security strategy. This approach, when implemented effectively, capitalizes on the individual strengths of both algorithms, enhancing overall security measures. Leveraging AES and Blowfish, established encryption standards, significantly raises the complexity for potential attackers, heightening the difficulty of breaking the encryption, particularly when configured appropriately. Encrypting files before they reach the cloud provides an additional layer of security, safeguarding the data even if unauthorized access to the cloud storage occurs. This approach ensures that the encrypted files remain protected, underscoring the bolstered security posture offered by this hybrid encryption method in cloud environments.

Firstly, we examined the integration process involves assessing how these algorithms seamlessly interoperate within the cloud storage infrastructure. Understanding how Blowfish and DAES can be implemented to encrypt data at rest and in transit within cloud servers or during file transfers is critical. Additionally, ensuring compatibility and optimization with the storage systems' architecture is key to maintaining efficiency without compromising security. Secondly, we focused onto the impact of this integration on performance and data transfer efficiency, especially when encrypting and uploading files to a service like Amazon S3. Analyzing factors such as computational overhead, processing time, and resource utilization during encryption and transfer processes is crucial.

6.1 Experiment 1: Encrypting the files using DAES algorithm.

We have considered text files with different sizes like 64kb, 128kb, 256kb, 512kb, 1024kb, 2048kb and 4096kb. The encrypted files are then uploaded to the Amazon s3 bucket.

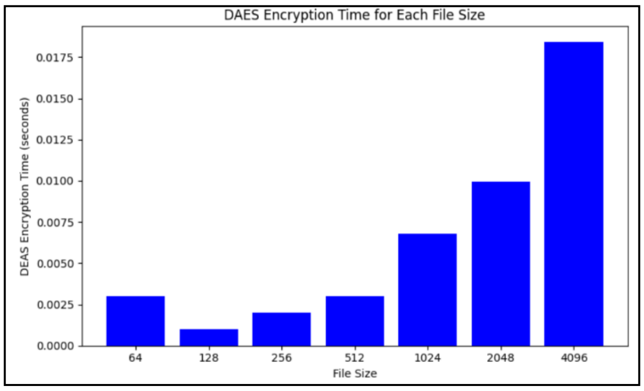
Once the algorithm is performed in the Jupyter notebook, after running the code the encryption password needs to be entered by us. Then Encryption time and Throughput are shown as output.
Code Evaluation:

- Password-Based Key Derivation: Utilizes PBKDF2HMAC with SHA256 for key derivation, enhancing key strength from the provided password.
- Double AES Encryption: Implements a two-stage AES encryption process for heightened security.
- S3 Upload Integration: Successfully uploads the doubly encrypted data to an S3 bucket, showcasing integration with AWS services.

Output Evaluation:

- Encryption Time: Encryption Time refers to the duration it takes to encrypt data, typically measured in seconds. It signifies the time taken by an encryption algorithm to process and convert plaintext into ciphertext, ensuring data security.

- Throughput: Throughput represents the rate at which data is processed during encryption, usually measured in bytes per second (B/s). It indicates the efficiency or speed of the encryption process, depicting how quickly a system can encrypt data of a certain size. Higher throughput values imply faster encryption speeds.



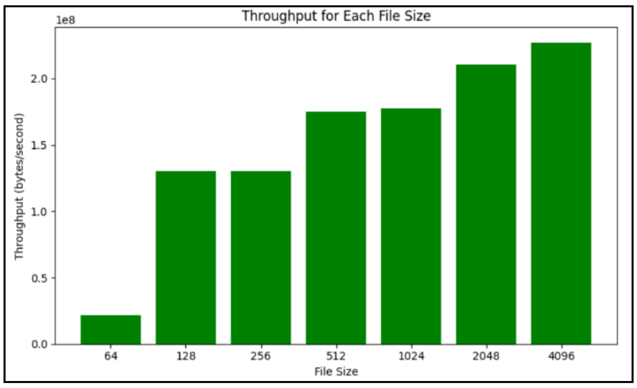


Figure 4: Plot of encryption time and throughput for DEAS algorithm

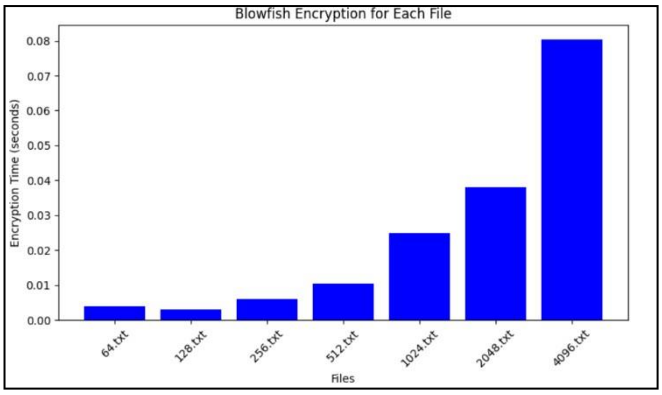
Tabel 1: Encryption time and throughput results using DEAS algorithm.

File Size (bytes)	Encryption Time (seconds)	Throughput (bytes/second)
64	0.002995	21,402,270.11
128	0.000998	130,443,455.88
256	0.001996	130,398,711.74
512	0.002993	174,923,493.43
1024	0.006766	177,530,638.30
2048	0.009973	210,268,718.17
4096	0.01845	227,302,258.61

6.2 Experiment 2: Encrypting the files using Blowfish algorithm.

Code Evaluation:

- Key Derivation: The code utilizes PBKDF2HMAC with SHA256 for key derivation, enhancing the strength of the generated key from the password.
- Salt Generation: Salt is generated for the Blowfish encryption, ensuring randomness for key derivation.
- AWS Integration: The script successfully uploads the encrypted data to an S3 bucket, integrating well with AWS services.



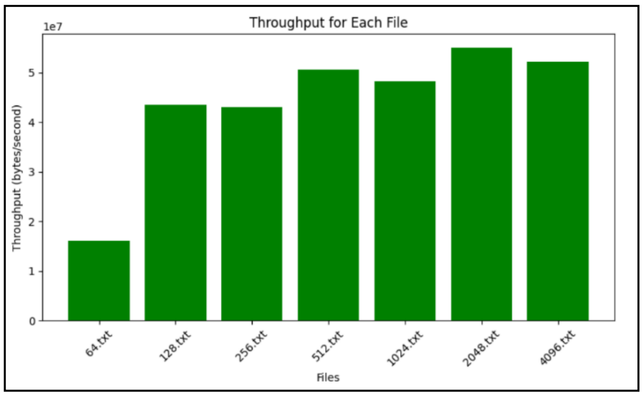


Figure 5: Plot of encryption time and throughput for Blowfish algorithm

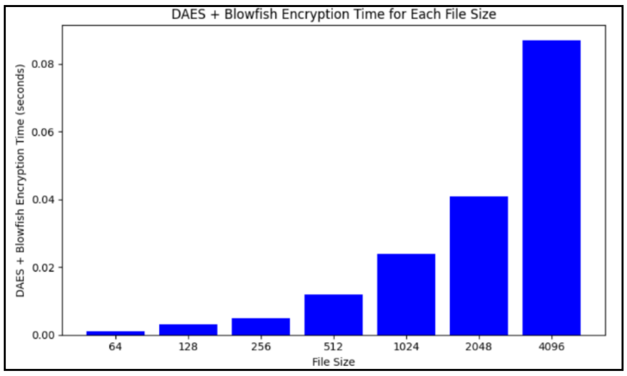
Tabel 2: Encryption time and throughput results using DEAS algorithm

File Size (bytes)	Encryption Time (seconds)	Throughput (bytes/second)
64	0.001994	32,137,945.59
128	0.000998	130,443,455.88
256	0.002025	128,525,564.11
512	0.003023	173,185,157.58
1024	0.007978	150,552,475.14
2048	0.013963	150,177,067.25
4096	0.028915	145,041,845.42

6.3 Experiment 3

Encryption Process: Utilizes DEAS and Blowfish algorithms for encryption.
Key Derivation: Implements PBKDF2 with HMAC-SHA256 for key derivation, which enhances security.
Encryption Time: This time represents the combined duration for encrypting the data using both DAES and Blowfish algorithms.
Throughput: This figure indicates the rate at which data was encrypted.
Analysis

- Speed: The encryption process seems to be quite fast, considering the short duration.
- High Throughput: The high throughput value compared to encryption using single algorithm indicates that the encryption speed for both algorithms was extremely efficient, resulting in a rapid encryption rate.



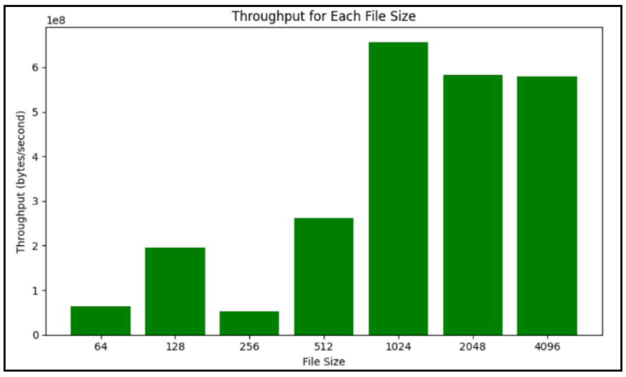
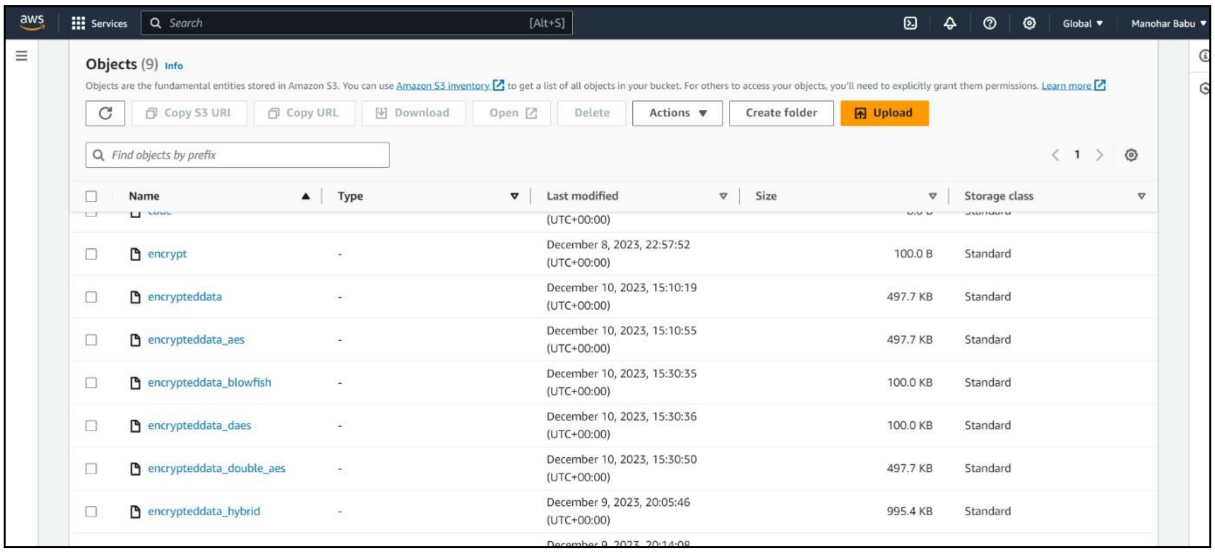


Figure 6: Plot of encryption time and throughput for hybrid (DAES+Blowfish) algorithm

Tabel 3: Encryption time and throughput results combing DEAS and Blowfish algorithm

File size (bytes)	DEAS + Blowfish Encryption Time (seconds)	Throughput (bytes/second)
64	0.000999	64,185,166.77
128	0.002993	195,435,906.40
256	0.004987	52,185,980.32
512	0.011969	262,028,096.39
1024	0.023961	656,782,309.24
2048	0.04083	582,443,548.01
4096	0.087021	578,924,814.92

The code effectively encrypts data using AES and Blowfish algorithms, employing PBKDF2 for key derivation and securely uploading the encrypted content to AWS S3. This suggests the efficiency of both algorithms in swiftly processing substantial data volumes. The code's ability to consistently deliver swift encryption with high throughput signifies its effectiveness in securing files for storage in cloud environment like Amazon S3.



Objects (9) Info

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

Name	Type	Last modified	Size	Storage class
		(UTC+00:00)		
encrypt	-	December 8, 2023, 22:57:52 (UTC+00:00)	100.0 B	Standard
encrypteddata	-	December 10, 2023, 15:10:19 (UTC+00:00)	497.7 KB	Standard
encrypteddata_aes	-	December 10, 2023, 15:10:55 (UTC+00:00)	497.7 KB	Standard
encrypteddata_blowfish	-	December 10, 2023, 15:30:35 (UTC+00:00)	100.0 KB	Standard
encrypteddata_daes	-	December 10, 2023, 15:30:36 (UTC+00:00)	100.0 KB	Standard
encrypteddata_double_aes	-	December 10, 2023, 15:30:50 (UTC+00:00)	497.7 KB	Standard
encrypteddata_hybrid	-	December 9, 2023, 20:05:46 (UTC+00:00)	995.4 KB	Standard

Figure 5: The encrypted files in the Amazon s3 bucket

6.4 Discussion

This system emphasizes robust security measures, including email verification, OTP-based mobile number verification, and access controls based on specific IP addresses. It provides a seamless user experience through registration, login, role-based redirection, and secure logout functionalities. The utilization of Firebase Database ensures secure data storage and management. Additionally, the implementation of specific IP address restrictions on Amazon EC2 enhances the system's security posture by limiting access to trusted sources.

The implementation of a hybrid encryption structure utilizing AES and Blowfish algorithms for file encryption before uploading to Amazon S3 represents a robust security measure. This approach not only introduces added complexity but also diversifies the encryption techniques, mitigating the risks posed by brute force and algebraic attacks. The proposed structure, coupled with an authentication mechanism within Amazon S3, fortifies data access controls. The implementation within Amazon S3 demonstrates a proactive step towards securing sensitive data in the cloud, but continuous monitoring and potential enhancements to the encryption strategy can further bolster its resilience against evolving threats, ensuring sustained data integrity and confidentiality. This encryption script employs PBKDF2 for key derivation, employing AES or Blowfish algorithms to encrypt files based on the user's choice. It measures encryption time and throughput, providing insights into performance. Upon encrypting the data, it securely uploads it to an AWS S3 bucket. The code integrates various security measures like salt generation using a secure random number generator and handles sensitive user input using get pass to safeguard the encryption password.

7 Conclusion and Future Work

The Zero Trust model represents a paradigm shift in cybersecurity compared to traditional security models. It challenges the assumption of trust, enforces identity-centric security, implements fine-grained access control, and places a strong emphasis on continuous monitoring and threat detection. Zero Trust provides significant advantages over traditional security models, particularly in the context of a cloud-first, mobile-first world. With its focus on enhanced security, safeguarding sensitive data, greater visibility and control, increased flexibility, and simplified compliance and governance, Zero Trust is the recommended approach to address evolving cyber threats. This implementation demonstrates a robust user management system adhering to the principles of Zero Trust, employing Firebase tools effectively for authentication, access control, secure data handling, and logging. The combination of Firebase services and MFA creates a strong foundation for a secure web application, though regular assessments and updates may be required to adapt to evolving security threats and user requirements.

Combining Dynamic AES (DAES) and Blowfish algorithms in a hybrid encryption model presents a formidable advantage in data security within cloud environments and also provides a dual approach that strengthens the overall security posture. The synergy of these algorithms enhances the robustness of data protection by leveraging Blowfish's efficiency in bulk encryption and DAES's proficiency in key exchange. This hybridization streamlines key management processes, optimizing efficiency without compromising on security. Furthermore, it strikes a balance between speed and robust encryption, ensuring swift data transmission while maintaining a high level of protection. The adaptability of this hybrid approach caters to diverse security requirements within cloud computing, offering comprehensive and versatile protection against a spectrum of potential security threats.

The DAES and Blowfish configuration excels in throughput but sacrifices some encryption time. The Double AES encryption with a specific key demonstrates a balance between encryption time and throughput, while the single encryption with a unique salt shows lower throughput but comparable encryption time to the combined encryption.

As a part of future work, enhancing RBAC policies involves a major approach toward refining role definitions, access controls, and permissions to fortify security in intricate hybrid cloud environments. By developing adaptive and scalable RBAC policies, organizations can tailor access rights with precision, minimizing security gaps and ensuring granular control over user privileges. Simultaneously, automated RBAC management solutions, such as hybrid-specific identity and access management (IAM) platforms, streamline the complexities of role assignments, access reviews, and compliance checks. Automation not only simplifies these processes but also ensures consistency and accuracy in applying RBAC policies across diverse cloud infrastructures, bolstering overall security while optimizing administrative efforts in managing access controls. Ensuring the scalability and compatibility of the hybrid encryption approach across diverse cloud architectures, including emerging paradigms like edge computing and Internet of Things (IoT) is pivotal for its widespread applicability. An GUI model can be built were user can enter bucket name and upload the file. Adapting this model to accommodate various infrastructures and evolving technological landscapes enhances its versatility and effectiveness in safeguarding data within dynamic environments.

References

Basavala, S.R., Kumar, N. and Agarrwal, A., 2012, December. Authentication: An overview, its types and integration with web and mobile applications. In 2012 2nd IEEE International Conference on Parallel, Distributed and Grid Computing (pp. 398-401). IEEE.

Stafford, V.A., 2020. Zero trust architecture. NIST special publication, 800, p.207.
Velásquez, I., Caro, A. and Rodríguez, A., 2018. Authentication schemes and methods: A systematic literature review. Information and Software Technology, 94, pp.30-37.

Ometov, A., Bezzateev, S., Mäkitalo, N., Andreev, S., Mikkonen, T. and Koucheryavy, Y., 2018. Multi-factor authentication: A survey. Cryptography, 2(1), p.1.

Reese, K., Smith, T., Dutson, J., Armknecht, J., Cameron, J. and Seamons, K., 2019. A usability study of five {two-factor} authentication methods. In Fifteenth Symposium on UsablePrivacy and Security (SOUPS 2019) (pp. 357-370).

Alappat, M.R., 2023. Multifactor Authentication Using Zero Trust (Doctoral dissertation, Rochester Institute of Technology).

Mehraj, S. and Banday, M.T., 2020, January. Establishing a zero-trust strategy in cloud computing environment. In 2020 International Conference on Computer Communication and Informatics (ICCCI) (pp. 1-6). IEEE.

Abhiram, D., Harish, R. and Praveen, K., 2022. Zero-trust security implementation using sdp over vpn. In Inventive Communication and Computational Technologies: Proceedings of ICICCT 2021 (pp. 267-276). Springer Singapore.

Abwnawar, N., 2020. A policy-based management approach to security in cloud systems (Doctoral dissertation, De Montfort University).

Alappat, M.R., 2023. Multifactor Authentication Using Zero Trust (Doctoral dissertation, Rochester Institute of Technology).

Li, S., Iqbal, M. and Saxena, N., 2022. Future industry internet of things with zero-trust security. Information Systems Frontiers, pp.1-14.

Mir, A.W. and Ram Kumar, K.R., 2021. Zero trust user access and identity security in smart grid based scada systems. In Proceedings of the 12th International Conference on Soft Computing and Pattern Recognition (SoCPaR 2020) 12 (pp. 716-726). Springer International Publishing.

Muhammad, T., Munir, M.T., Munir, M.Z. and Zafar, M.W., 2022. Integrative Cybersecurity: Merging Zero Trust, Layered Defense, and Global Standards for a Resilient Digital Future. INTERNATIONAL JOURNAL OF COMPUTER SCIENCE AND TECHNOLOGY, 6(4), pp.99-135.

Singh, M., Mehtre, B.M., Sangeetha, S. and Govindaraju, V., 2023. User Behaviour based Insider Threat Detection using a Hybrid Learning Approach. Journal of Ambient Intelligence and Humanized Computing, 14(4), pp.4573-4593.

Talan, A., 2022. Zero Trust Network Access with Cybersecurity Challenges and Potential Solutions (Doctoral dissertation, Dublin, National College of Ireland).

Tanimoto, S., Yangchen, P., Sato, H. and Kanai, A., 2023. Suitable Scalability Management Model for Software-Defined Perimeter Based on Zero-Trust Model. International Journal of Service and Knowledge Management, 7(1).

Uwaoma, C., 2023. The Challenges and Processes of Achieving Optimal Implementation of Zero Trust Architecture in Workplace (No. 10878). EasyChair.

Abdul, D. S., Hatem, M. A. K. & Hadhoud, M. M., 2008. Performance Evaluation of Symmetric Encryption. IJCSNS International Journal of Computer Science and Network Security.

Ghosh, S. N., Biradar, D. T., Shinde, G. C. & Bhojaned, S. D., 2015. Performance Analysis of AES, DES, RSA And. International Journal of Innovative and Emerging.

R.Udendhran, 2017. A Hybrid Approach to Enhance Data Security in Cloud.

S., Olanrewaju, R. F. O. F. & Abdullah, . K., 2018. Enhancing Cloud Data Security using Hybrid of. East Indonesia Conference and Information Technology.