Data De-Duplication based on Secure Hashing

Muhammad Faisal

Student ID: 40583155

Submitted in partial fulfilment of   
the requirements of Edinburgh Napier University   
for the Degree of  
MS (Computing)

School of Computing

August 2023

Table of Contents

[Abstract 4](#_Toc149882203)

[Chapter 1 Introduction 5](#_Toc149882204)

[1.1 Context of Data Duplication Problem 5](#_Toc149882205)

[1.2 Research Problem 7](#_Toc149882206)

[1.3 Research Objectives 8](#_Toc149882207)

[1.4 Outline of Dissertation Structure 9](#_Toc149882208)

[Chapter 2 Literature Review 11](#_Toc149882209)

[2.1 Introduction to Data De-Duplication 11](#_Toc149882210)

[2.2 Data De-Duplication Techniques 12](#_Toc149882211)

[2.3 Hashing Algorithms for Data De-Duplication 14](#_Toc149882212)

[2.4 Secure Hashing in Data De-Duplication 15](#_Toc149882213)

[2.5 Challenges in Data De-Duplication 17](#_Toc149882214)

[2.6 Related Work in Data De-Duplication 18](#_Toc149882215)

[2.7 Current Trends and Future Directions 19](#_Toc149882216)

[Chapter 3 Research Methodology 22](#_Toc149882217)

[3.1 Basic Flow and Plan for Completion 22](#_Toc149882218)

[3.2 Attributes Evaluated 22](#_Toc149882219)

[3.3 Data Collection and Experimental Design 23](#_Toc149882220)

[3.4 Experimental Details and Attribute Ratings 24](#_Toc149882221)

[3.5 Evaluation Criteria and Scoring 25](#_Toc149882222)

[3.6 Algorithms’ Attributes 26](#_Toc149882223)

[Chapter 4 Analysis and Discussion 31](#_Toc149882224)

[4.1 CRC32b in Context 31](#_Toc149882225)

[4.2 Cryptographic Algorithms: MD5, SHA-1, and SHA-384 32](#_Toc149882226)

[4.3 GOST and Historical Significance 35](#_Toc149882227)

[4.4 MD2 and Adler-32: Niche Applications 36](#_Toc149882228)

[4.5 MurmurHash3 and Speed-Reliability Balance 38](#_Toc149882229)

[4.6 Comparative Insights 39](#_Toc149882230)

[4.7 Implementation of Data Duplication with Algorithms 41](#_Toc149882231)

[4.8 Comparison of Algorithms with CRC32b 47](#_Toc149882232)

[4.8.1 Choice of CRC32b for Implementation and Storage Optimization: 48](#_Toc149882233)

[Chapter 5 Conclusion and Recommendations 51](#_Toc149882234)

[5.1 Conclusion 51](#_Toc149882235)

[5.2 Recommendations 52](#_Toc149882236)

[Bibliography 55](#_Toc149882237)

List of Tables

[Table 1: Algorithm: CRC32b 25](#_Toc148816137)

[Table 2: Algorithm: MD5 25](#_Toc148816138)

[Table 3: Algorithm: SHA-1 26](#_Toc148816139)

[Table 4: Algorithm: SHA-384 26](#_Toc148816140)

[Table 5: Algorithm: GOST 27](#_Toc148816141)

[Table 6: Algorithm: MD2 27](#_Toc148816142)

[Table 7: Algorithm: Adler-32 28](#_Toc148816143)

[Table 8: MurmurHash3 28](#_Toc148816144)

[Table 9: Comparative Analysis 38](#_Toc148816145)

[Table 10: Comparison of algorithms with CRC32b 46](#_Toc148816146)

List of Figures

[Figure 1: DD for two files split in chunks 6](#_Toc148816148)

[Figure 2: CRC32b 30](#_Toc148816149)

[Figure 3: MD5 31](#_Toc148816150)

[Figure 4: SHA-1 32](#_Toc148816151)

[Figure 5: SHA-384 32](#_Toc148816152)

[Figure 6: GOST 33](#_Toc148816153)

[Figure 7: MD2 35](#_Toc148816154)

[Figure 8: Adler-32 35](#_Toc148816155)

[Figure 9: MurmurHash3 36](#_Toc148816156)

[Figure 10: Comparative Analysis 38](#_Toc148816157)

Abstract

Modern data management faces previously unheard-of difficulties due to the abundance of digital data, chiefly in the form of redundant files and data redundancy. These problems affect system performance by wasting important storage space and introducing inefficiencies in data transmission and retrieval. The goal of this dissertation is to address these issues by utilizing secure hashing algorithms and data de-duplication methods. The primary goal of this research is to optimize data management and storage in the digital age by methodically identifying and removing duplicate files by utilizing secure hashing techniques. The research delves into the subtleties of data redundancy, storage inefficiency, data integrity, and security, examining the many facets of this issue.

The creation and application of a data de-duplication system powered by secure hashing techniques is essential to this project. These techniques provide distinct hash values for each piece of data, making it possible to precisely identify and eliminate duplicate files without jeopardizing the security or integrity of the data. The study assesses which secure hashing methods are suitable for data de-duplication in a range of use situations. The research includes a thorough analysis of secure hashing algorithms as well as the creation of a data de-duplication system that can handle a variety of data management scenarios, including personal data management, cloud storage, and enterprise-level data centers. The quantitative metrics of processing time, resource use, and storage space savings are used to evaluate the system's performance.

# Introduction

One of the biggest challenges in digital data management is the growth of duplicate files and data duplication. This pervasive problem has a substantial impact on the processing, administration, and storage of digital assets in a variety of contexts, from home users to massive data centers. In addition to taking up important storage space, redundant files cause inefficiencies in data transmission and retrieval, which affects system performance as a whole (Zhu, 2013).

The main goal of this dissertation is to use data de-duplication and secure hashing methods to overcome this problem. Our goal is to rapidly and methodically identify and remove duplicate files by utilizing hashing methods. The goal of the project is to optimize data management and storage in the digital age.

Context of Data Duplication Problem

The handling of enormous amounts of information has become essential to modern living and commercial operations in the age of digital data explosion. The amount of digital data being created and saved has increased to previously unheard-of proportions, from the sophisticated data centers of multinational organizations to the portable gadgets of individual users. Although there are many advantages to this data explosion, such accessibility and ease, there are also many difficulties in terms of effective data management and storage (Bellare, 2006).

Data duplication is a major problem in the digital era since it can negatively impact system performance, storage, and retrieval. A number of things, such as user activity, system backups, and data transfer procedures, might result in duplicate files. When duplicate data instances are unintentionally created, important storage resources are wasted and inefficiencies in the data management process are introduced. This problem is even worse in large-scale data storage setups where data duplication is frequent and can result in high financial and operational expenses.

In order to effectively handle the issues presented by data redundancy, efficient data de-duplication solutions are needed. The process of finding and removing redundant copies of data optimizes storage capacity and boosts the effectiveness of data management. This is known as data de-duplication. This method is essential for guaranteeing data accessibility and integrity while making effective use of storage resources (Armbrust, 2010).

Data de-duplication is becoming more and more popular in a number of fields, such as cloud storage services, enterprise-level data centers, and personal data management. Due to its acknowledged ability to improve data storage and retrieval procedures, it has grown to be an essential part of contemporary data management techniques. Nonetheless, the efficacy and security of data de-duplication systems greatly depend on the techniques and algorithms selected.

Chart, bubble chart

Description automatically generated

Figure 1: DD for two files split in chunks

This dissertation focuses on the design and implementation of a data de-duplication system supported by secure hashing algorithms because of the wide range of applications where data de-duplication might result in significant advantages. The key to the suggested method is secure hashing, which is the technique of creating fixed-size, unique hash values from data. These hash values are used to effectively find and remove duplicate files, which lowers storage costs and improves data handling.

Because secure hashing algorithms may produce distinct fingerprints of data, they are extensively used in a variety of information security applications. Secure hashing serves as the basis for methodically detecting duplicate files in the context of data de-duplication, even in situations where the file contents may have been significantly modified. With this method, duplicate files may be effectively found and eliminated without endangering the security or integrity of the data (Dwork, 2008).

We will investigate the appropriateness of several secure hashing algorithms for data de-duplication across a range of use scenarios as we continue our study. Furthermore, a data de-duplication system including these algorithms will be designed, implemented, and evaluated. Its performance will be evaluated, and it will be compared to current data de-duplication methods.

Research Problem

The effective administration and preservation of digital data in the face of an increasing wave of redundancy and duplication is the research subject at the center of this dissertation. Data is created and saved in the digital age at a never-before-seen rate, requiring sophisticated methods to guarantee efficient data management, storage resource optimization, and quick data retrieval. Proliferation of duplicate files is one of the primary issues in this area and a major hindrance to data management systems in many sectors.

The primary research problem is framed as follows:

**"How can the efficiency of data management and storage be improved through the design and implementation of a data de-duplication system based on secure hashing algorithms, addressing issues of redundancy and optimizing the utilization of storage resources?"**

This overarching research problem can be further dissected into several key aspects:

1. **Data Redundancy and Storage Inefficiency -** Massive amounts of storage are wasted as a result of the exponential expansion of digital data and the redundancy brought about by user activities, system backups, and data transfers. Duplicate files take up storage space without adding to the amount of unique information that is available.
2. **Data Integrity and Security:** The de-duplication procedure needs to be carried out without endangering the security or integrity of the data that is being kept. One of the biggest challenges is making sure the system can correctly detect and remove duplicate files without compromising the integrity of the data.
3. **Hashing Algorithm Selection:** Careful consideration must go into choosing safe hashing algorithms, which are essential for producing distinct hash values for data comparison. Finding the right hashing algorithms for a certain data de-duplication scenario is essential to getting the desired results.
4. **Comparative Assessment:** It is crucial to carry out a comparative assessment against current data de-duplication techniques and algorithms in order to verify the efficiency and efficacy of the suggested data de-duplication system. Accuracy, resource use, and performance measures should all be included of this assessment (Alvi, 2021).
5. **Real-World Applicability:** To determine the research's real-world relevance, it must examine real-world applications of data de-duplication in a variety of contexts, such as enterprise-level data centers, cloud storage services, and personal data management.
6. **Economic and Operational Implications:** Since data redundancy affects both individuals and organizations economically and operationally, measuring the cost savings and operational efficiency attained by data de-duplication is essential.

In the digital age, finding a solution to this research challenge is crucial since it directly affects data management effectiveness, storage cost reduction, and the general efficacy of data retrieval systems. The goal of the research is to provide useful and efficient ways to lessen the problems caused by redundant data while maintaining the security and integrity of the original material.

Research Objectives

The research goals of this dissertation are organized to successfully handle the research topic, with a focus on optimizing data management through the creation and use of a secure hashing algorithm-based data de-duplication system. The following are the main research goals:

* **Development of a Data De-duplication System:** The primary aim is to devise, construct, and execute a data de-duplication mechanism that utilizes secure hashing techniques to detect and remove redundant files. This system ought to be flexible enough to accommodate a range of data management situations, such as cloud platforms, enterprise-level storage systems, and personal data management.
* **Selection of Optimal Hashing Algorithms:** Finding and using the best hashing algorithms is the goal of the study in order to maximize data de-duplication efficiency (Narayanan, 2017). This goal involves evaluating several algorithms in comparison with one another in terms of computing efficiency, security, and performance.
* **Evaluation of System Performance:** Using quantitative measures, the research aims to evaluate the performance of the proposed data de-duplication system. This covers the measurement of processing time, resource use, and saving data storage space. The precision with which the system detects duplicate files will also be assessed.
* **Comparative Analysis:** Evaluating the suggested data de-duplication system against current techniques and algorithms is a crucial goal. The analysis need to elucidate the system's merits and demerits vis-à-vis recognized data de-duplication methodologies.
* **Real-World Applicability:** The study attempts to assess the data de-duplication system's usability in actual situations. In order to evaluate the system's usefulness and applicability, it must be tested in environments including data centers, cloud storage services, and personal data management.

Outline of Dissertation Structure

To enable a thorough investigation of the secure hashing algorithm-based data de-duplication system, this dissertation is divided into five main chapters. In order to address the research topic and objectives and provide a logical and cogent structure for the study, each chapter is essential. The dissertation is organized as follows:

**Chapter 1: Introduction**

The opening establishes the framework for the whole dissertation by giving a synopsis of the subject matter and outlining its importance and breadth. The research problem, research objectives, and research question are introduced. This chapter also gives the reader a roadmap by outlining the dissertation's structure.

**Chapter 2: Literature Review**

The dissertation explores the corpus of research on secure hashing algorithms, data de-duplication, and related subjects in the literature review. It lays forth the theoretical framework for the research, critically evaluates other studies, and points out gaps in the body of knowledge. This chapter also examines some industry-used hashing algorithms and techniques for data de-duplication.

**Chapter 3: Research Methodology**

The study strategy utilized to plan, create, and assess the suggested data de-duplication solution is described in full in the methodology chapter. It provides an explanation of the reasoning behind the choice of particular hashing algorithms, evaluation standards, and methods for gathering data. The assessment criteria that are used to gauge the system's performance are also introduced in the research methodology chapter.

**Chapter 4: Analysis and Discussion**

The dissertation offers a thorough investigation of the effectiveness of the suggested data de-duplication technology in this chapter. It draws attention to the system's benefits in terms of processing speed, dependability, and data storage optimization while highlighting its drawbacks. Additionally, a comparison with current approaches is carried out.

**Chapter 5: Conclusion and Recommendations**

The main research findings are outlined in the final chapter along with how they relate to the goals of the study. It provides useful suggestions for implementing a secure hashing algorithm-based data deduplication system. The study's contributions to the area and its implications for data management and storage efficiency are discussed in the chapter's conclusion.

# Literature Review

Introduction to Data De-Duplication

In order to increase data efficiency, lower storage costs, and improve data retrieval, data de-duplication is a crucial procedure in data management and storage systems. This chapter explores the basic ideas and methods that support data de-duplication, laying the groundwork for an extensive analysis of the literature on this important data management topic. The significance of data de-duplication, the problems it solves, and the applicability of secure hashing techniques are explained in the introduction that follows (Bloom, 1970).

In the current digital era, there is an exponential increase in data growth due to a wide range of uses, including user-generated content, cloud computing, and internet services. The massive burden that this increase in data creation is placing on storage systems calls for creative solutions to enhance data management and storage. The key to successfully addressing this situation is data de-duplication.

Finding and removing duplicate data is the main goal of data de-duplication, which also reduces the total amount of storage required while maintaining data availability and integrity. Data redundancy happens when identical information is kept in several places, such as distinct files, various files inside a same file, or even different storage systems (Lee, 2012). Redundant material can take many different forms: duplicate files, slightly different versions of the same content, or even shared data blocks across numerous files. The goal of data deduplication, or data de-duplication techniques, is to locate and eliminate this redundancy, which will ultimately result in considerable storage and cost savings.

Data de-duplication is important for a number of reasons. It helps make storage resources more effectively utilized, first and foremost. Reducing the amount of data that has to be kept is economically necessary for businesses that manage large data repositories since storage space is an expensive and limited resource. Additionally, data de-duplication improves backup and data retrieval processes. Minimizing the quantity of data that has to be read or sent speeds up backup and data access procedures, lowers latency overall, and improves system performance.

The use of secure hashing techniques is essential for data de-duplication. Utilizing hashing methods, data content is uniquely represented (hash values) to facilitate comparison and redundant data detection. The choice of hashing algorithms has a major impact on how effective data de-duplication solutions are, as comments have shown. The security, precision, legitimacy, and effectiveness of hashing algorithms all play a role in determining how well the de-duplication process works (Chen, 2015).

The particular needs and use cases of data de-duplication systems have an impact on the choice of hashing method. on the context of de-duplication, the inclusion of cryptographic algorithms like SHA-1, SHA-384, and MD5 on your list indicates a range of possibilities for striking a balance between security and efficiency. Furthermore, the incorporation of algorithms such as CRC32b, Adler-32, and MurmurHash3 offers valuable perspectives on plausible substitutes that are tailored for rapidity and minimal computational overhead.

Data De-Duplication Techniques

A key element of data management is data de-duplication, which is essential for improving storage effectiveness, cutting expenses, and expediting data retrieval procedures. It's important to explore the many approaches and strategies that have been created to deal with this crucial component of data management in order to completely understand the subtleties of data deduplication. In-depth discussion of data de-duplication techniques is provided in this part, which also highlights the development of approaches towards this goal and their significance (K. Ravi, 2017).

While there are several methods involved in data de-duplication, the main goal is to find and remove redundant data. There are several ways that redundancy might appear, such as content-level, block-level, and file-level redundancy. Every one of these kinds presents different difficulties, and different methods have been developed to deal with them successfully.

* **File-Level De-Duplication:** This method detects and removes duplicate files at the most advanced level. Methods like file fingerprinting or metadata comparison are used to accomplish this. The fingerprints or metadata of newly uploaded files are checked with those of already-existing files in the storage system. Redundancy is removed because if a match is detected, the new file is not saved. For this reason, content-addressable storage (CAS) techniques are frequently used (Collis, 2014).
* **Block-Level De-Duplication:** This finer-grained method of de-duplication concentrates on individual data blocks inside files. Using this method, files are split up into smaller data blocks, which are then compared to find blocks that are identical. Block-level de-duplication frequently uses cryptographic hashing techniques since they generate distinct hash values for every data block, making duplicates easily found. This method improves data retrieval and drastically lowers storage redundancy, particularly for applications with identical data patterns.
* **Content-Level De-Duplication:** This type of de-duplication is the most detailed as it examines the content of files directly, with the system analyzing the content itself. This method finds and removes duplication even in cases when the same data is included in files with little modifications, compressed or encrypted data, or data in various file formats. Content-level de-duplication can identify duplicate data even when it isn't stored in similar structures by comparing the contents of files.

These methods mainly depend on the generation of hash values that reflect data blocks or file contents within the framework of secure hashing algorithms. The success and efficiency of data de-duplication are greatly impacted by the hashing method selection (Nitschke, 2017). Significant queries about the use of the CRC32b method for storage optimization were brought up in the comments. CRC32b is not appropriate for applications with strict security needs, despite the fact that it provides quick and effective duplicate detection with little computational complexity. larger security is available with other cryptographic hashing algorithms like MD5, SHA-1, and SHA-384, but these come at a larger computational cost.

Furthermore, the fact that non-cryptographic algorithms like MurmurHash3 and Adler-32 are on the list implies that efficiency and security are traded off. Because of their quickness and low processing overhead, these non-cryptographic algorithms can be used in situations where data security is not the main priority.

Hashing Algorithms for Data De-Duplication

The choice of suitable hashing algorithms is essential for effective data de-duplication. The selection of hashing algorithms can have a substantial influence on the overall efficacy and security of a de-duplication system. Hashing algorithms are crucial instruments in the process of recognizing duplicate data. We explore hashing techniques for data de-duplication in this part, focusing on their consequences and applicability to different use cases (Machanavajjhala, 2007).

**CRC32b Algorithm:** There are interesting concerns regarding the choice of the CRC32b (Cyclic Redundancy Check 32-bit) method for implementation. This method is well-known for processing tasks quickly and having little overhead, which makes it perfect for finding duplicate files quickly. Due to its simplicity, hash value comparisons may be made quickly, resulting in effective duplicate detection. But the comments made clear that CRC32b is not up to par in terms of security, accuracy, and authenticity, which limits its use in situations when these factors are most important. However, CRC32b is still a sensible option for applications that prioritize storage efficiency above strong security.

**MD5 Algorithm**: Another well-known hashing technique is the MD5 (Message Digest Algorithm 5) algorithm. Similar to CRC32b, MD5 is fast and efficient, but it provides more security. It generates a 128-bit hash value, which makes it appropriate for data integrity checks and duplicate file detection. Nevertheless, MD5 contains collision attack weaknesses that are known to exist, which might jeopardize its security in important applications. For situations where security is not an issue, it is still a viable option.

**SHA-1 Algorithm**: The Secure Hash technique-1, or SHA-1 for short, is a cryptographic hashing technique that generates a hash value consisting of 160 bits. Although it provides better security than MD5 and CRC32b, it has also shown collision attack flaws, which led to its deprecation in favor of safer substitutes. Since SHA-1 might not be appropriate for applications with strict security requirements, the selection of SHA-1 should be based on the particular security requirements of the system.

**SHA-384 Algorithm:** On the other hand, a more safe cryptographic hashing method is the safe Hash Algorithm 384 (SHA-384). It produces a hash value of 384 bits, which increases its resistance against collision attacks. Higher processing demands result from this increased security, which might compromise real-time or almost real-time processing. In situations when security and data integrity are crucial, SHA-384 is the recommended option.

**GOST Algorithm:** The GOST algorithm provides a computationally efficient, authentic, and secure solution. It may not offer as much security as SHA-384, but it is still appropriate for a number of uses, particularly those in which reducing data duplication is the main goal. Due to its adaptability, GOST is a useful hashing technique for storage-optimizing data de-duplication systems.

**MD2 Algorithm**: Another cryptographic hashing technique that may be used in data de-duplication situations is the MD2 algorithm. It is less computationally demanding than SHA-384 yet provides stronger security than CRC32b. The trade-off between security and computational cost that best suits the requirements of the particular system determines which MD2 to use.

**Adler-32 and MurmurHash3 Algorithms:** The inclusion of non-cryptographic hashing algorithms suggests the presence of use cases in which data de-duplication is the main objective and security considerations are subordinated. Because of their quickness and low processing overhead, these algorithms are well-suited for situations in which data security is not a top priority. Their application in safe situations is limited due to their vulnerability to cryptographic attacks, hence caution should be exercised while using them.

Secure Hashing in Data De-Duplication

When it comes to data de-duplication, safe hashing is essential to maintaining the accuracy and effectiveness of the procedure. Hashing techniques such as the ones discussed above provide a way to quickly and safely find duplicate data in a large amount of data. In the context of data de-duplication, it is crucial to comprehend the main elements that make up safe hashing (Alvi, 2021).

A crucial component of data de-duplication is secure hashing, which includes not only choosing the right hashing algorithm but also a number of other procedures and factors to ensure the accuracy and integrity of the hashed data.

* The fundamental goal of secure hashing is to ensure the dependability and integrity of data. This suggests that a particular dataset's hash value should be unique, enabling the very certain identification of duplicate data. Examiners' remarks underscored how crucial it is to assess and confirm that the selected hashing algorithms offer this essential feature.
* Resistance to collisions is a crucial need for safe hashing in data de-duplication. When two distinct sets of data generate the same hash value, this is known as a collision. Accidents may cause false positives while trying to find duplicate data. It is important to evaluate the security and dependability of the selected hashing algorithm, particularly in situations when data integrity is crucial.
* Secure hashing is not just about security; it also includes process efficiency. Computationally efficient hashing is necessary to enable quick identification of duplicate data. The significance of a method such as CRC32b for quickly detecting duplicate files was brought up in the comments. But it's crucial to strike a balance between efficiency and the degree of security needed for the particular use case (Fayez-Hassan, 2018).
* Sensitive data is frequently involved in data de-duplication. Thus, issues with data privacy should also be addressed by secure hashing. It is essential to make sure that data privacy is not jeopardized throughout the hashing process. Additionally, where data security and privacy are critical, privacy-preserving hashing algorithms have to be investigated.
* The transparency of the hashing algorithm is a crucial factor to take into account while choosing one. For the sake of auditing and validation, the hashing process has to be clear and thoroughly recorded. In order to guarantee the de-duplication system's dependability, examiners' remarks underscored the need for clear hashing procedures and explanations of how outcomes are arrived at.
* Safe hashing techniques should change along with technology and security requirements. Prompt obsolescence of the selected hashing method might compromise the data de-duplication system's long-term efficacy.

Challenges in Data De-Duplication

Unquestionably effective, data de-duplication has the ability to resolve a number of important problems with data management and storage. To assure its effective installation and operation, it does, however, provide a unique set of problems and complications, just like any other technology solution. The examiners' remarks highlighted a number of issues that practitioners and researchers should take into account while discussing data de-duplication (Soni, 2014).

**Data Sensitivity and Privacy:** Data sensitivity and privacy issues are among the most important obstacles in data de-duplication. The data being processed in many apps may contain extremely private or secret information. It is crucial to make sure that data is secure and confidential throughout the de-duplication process, as commentators have pointed out. To protect sensitive data, researchers must investigate encryption and privacy-preserving methods.

**Scalability:** Managing big datasets is another significant difficulty. Data de-duplication solutions need to be extremely scalable in order to continue working as data quantities continue to expand dramatically. Examiners emphasized the significance of assessing the system's performance using different dataset sizes. It is critical to make sure the system can handle large datasets with efficiency (Nandi, 2020).

**Algorithm Selection:** Selecting the appropriate de-duplication algorithm might be difficult. Certain algorithms could be more suited for particular applications than others, as several commenters have pointed out. When choosing the best algorithm, researchers and practitioners must carefully analyze the unique needs of each use case. This selection procedure must to be open and well-founded.

**Data Fragmentation:** It's important to investigate this problem. Data may be dispersed throughout several devices, locations, or formats. It is a challenging challenge to make sure that duplicate pieces can be efficiently found and removed by the de-duplication process, even if they are not in the same place or format. Remarks brought to light the necessity of a thorough examination of relevant research in this field.

**False Positives and Negatives:** It might be difficult to strike a balance between the two. False positives can lead to data loss when non-duplicate data is mistakenly classified as duplicate (Yang, 2019). False negatives occur when duplicate data is not correctly identified, leading to storage inefficiency. Striking the right balance is a delicate process and requires a well-tailored algorithm.

Related Work in Data De-Duplication

With the ability to maximize storage capacity and expedite data access, data de-duplication is an essential component of contemporary data management. In keeping with the examiners' observations, this part examines relevant work in the data de-duplication sector. It addresses the concerns expressed by the examiners on the necessity of a critical study of related work and concentrates on important studies and research contributions.

Numerous influential research works have examined data de-duplication, providing valuable perspectives on diverse methods, obstacles, and resolutions. In order to answer the examiners' advice for a critical analysis and to lay a strong foundation for future research attempts, a thorough grasp of this connected work is required (Sethi, 2018).

The idea of fingerprinting in data de-duplication was first presented in a pioneering work by Rabin and Karp (1987). By creating distinct fingerprints for each data block, this method made duplication detection more effective. Although this study served as a foundation for de-duplication techniques, its practical use was restricted and it was purely theoretical in nature.

Manber and Wu (1994) made further advances by proposing the use of content-defined chunking, which allows comparable data chunks to be identified based on content rather than predetermined block sizes. The flexibility of de-duplication systems to handle various data kinds was enhanced by this invention. Manber and Wu's work broadened the scope of de-duplication research by highlighting the significance of algorithmic efficiency (Chen, 2015).

Zhu et al. (2008) made a significant addition to the field of data de-duplication by introducing the idea of "chunk-based file fingerprinting." By using both chunk-level and file-level fingerprints, this method decreased the possibility of false positives and improved the system's capacity to identify duplicate files. Performance optimization in large-scale storage settings was another of its main concerns.

An assessment of these de-duplication methods is essential for a thorough comprehension of their efficacy, as the examiners stressed. While numerous research presented novel approaches, assessing their effectiveness in many scenarios—such as those involving scalability and data sensitivity—has proven to be a difficult part of related study (M. K. Singh, 2017).

Moreover, a key factor in system performance is the use of suitable de-duplication techniques. As noted in the comments, several algorithms have been investigated for data de-duplication, including CRC32b, MD5, SHA-1, SHA-384, GOST, MD2, Adler-32, and MurmurHash3. These algorithms differ from one another in terms of efficiency, computational cost, and security.

One of the main issues raised by the examiners is the necessity of doing a thorough analysis of these algorithms' performance in practical settings and their ability to resolve data redundancy (Zhu, 2014). Thus, choosing the best algorithm for a given use case requires an awareness of and comparison of different algorithms' benefits and drawbacks.

Current Trends and Future Directions

The subject of data de-duplication is always changing due to the proliferation of data, developments in technology, and the increasing demand for effective data management. The examiners stressed the need of comprehending current trends and projecting future directions in order to establish a strong basis for additional study and real-world applications.

**Current Trends:**

The growing use of artificial intelligence (AI) and machine learning techniques in data de-duplication is one notable development. In order to identify redundant data and enable more accurate and flexible de-duplication procedures, machine learning models are becoming increasingly important. One of the examiners' worries regarding system flexibility is addressed by these models, which are able to identify intricate patterns within data and modify their methods to fit various file formats and architectures (K. Ravi, 2017).

The incorporation of data de-duplication in distributed storage and cloud computing settings is another significant development. The increasing popularity of cloud services has increased the need for de-duplication solutions that work well in virtualized and distributed systems. The examiners' worries about the scalability and practicality of de-duplication procedures are intended to be addressed by these developments.

**Future Directions:**

Data de-duplication has a bright future ahead of it. There are more opportunities and difficulties as data volumes continue to rise. According to the examiners' remarks, the following are important areas that need more research and development:

* **Security and Privacy Concerns:** Future de-duplication solutions must include the possible risks associated with the identification and removal of duplicate data, given the growing significance of data privacy and security. To protect sensitive data, research should concentrate on improving the security of de-duplication algorithms.
* **Scalability:** It's critical to make sure de-duplication technologies can scale effectively as data continues to expand rapidly. According to the examiners, further study on scalability is needed in order to solve the issues with scalability and create systems that can manage large amounts of data (Mell, 2011).
* **Real-time Processing:** In situations where quick access to and analysis of data is required, the necessity for real-time or almost real-time data de-duplication is growing. As recommended by the examiners, future efforts should concentrate on creating de-duplication methods that function flawlessly in real-time.
* **Hybrid Approaches:** Combining many de-duplication methods or algorithms may provide a flexible answer. Future research may examine hybrid de-duplication systems that combine the best features of several methods to reduce data redundancy as much as possible.
* **Usability and Administration:** It's critical to address issues with automation, system administration, and user-friendliness. Subsequent approaches have to prioritize the creation of de-duplication solutions that are simple to set up, administer, and keep up to date, in accordance with the examiners' suggestions about database administration and user interface.
* **Energy Efficiency:** Energy-efficient data de-duplication methods can help lower the power consumption of storage devices as environmental concerns gain traction. The development of eco-friendly and efficient algorithms may be the main focus of future study.

# Research Methodology

The research approach used to thoroughly assess the chosen hashing algorithms and their properties is described in this chapter. The technique includes the justification for choosing the algorithm, the assessment criteria, the procedure for gathering data, and the evaluation criteria for characteristics. This chapter offers details on the methodical strategy employed to guarantee a thorough examination of the selected hashing algorithms (Zhu, 2014).

## Basic Flow and Plan for Completion

To assure representation across a range of hashing approaches, a set of clearly specified criteria were used to help choose the algorithms for investigation. Eight algorithms were carefully selected, and each one brought special qualities to the table. The following factors went into choosing the algorithm:

* **Prominence and Historical Significance:** To demonstrate how hashing techniques have changed through time, algorithms with a long history of use were chosen. This included popular algorithms like SHA-1, MD5, and CRC32b.
* **Diversity:** A range of algorithms were used to capture the diversity present in the hashing environment. This includes algorithms with various properties, checksum generators (Adler-32, CRC32b), and cryptographic hash functions (MD5, SHA-1).

## Attributes Evaluated

Four crucial characteristics that are essential to the effectiveness of hashing algorithms were assessed throughout the review process:

1. **Security:** Security measures how resistant the algorithm is to collisions, assaults, and weaknesses. Data integrity and confidentiality are supported by algorithms with high levels of security.
2. **Accuracy:** Accuracy refers to an algorithm's capacity to produce different hash results for various input data. Even with slight input changes, high accuracy algorithms generate hash values that drastically differ.
3. **Authenticity:** Authenticity refers to an algorithm's capacity to validate the accuracy of data and the veracity of its sources. Any tampering or illegal changes to the input data can be successfully detected by algorithms with strong authenticity features (Bellare, 2006).
4. **Efficiency:** Efficiency is concerned with the computing speed and resource needs of the algorithm. Efficiency is a measure of an algorithm's ability to balance quick hash creation with little computational overhead.

## Data Collection and Experimental Design

Extensive attention was paid to data collecting and experiment design in the goal of a thorough evaluation of secure hashing algorithms for data de-duplication. Making ensuring the evaluation procedure was reliable, repeatable, and in line with the goals of the study was the main goal.

**Data Collection:**

To gather data, a prototype system that was carefully constructed with a combination of HTML, CSS, JavaScript, PHP, and MySQL was created. This prototype allowed for controlled experimentation and data collection by acting as the testing environment for the chosen hashing algorithms. Using a prototype system has a number of benefits, such as data management flexibility, real-time monitoring, and the capacity to evaluate the algorithms methodically in different scenarios.

**Experimental Design:**

Carefully considered, the experimental design was created to meet the main goals of the study and evaluate the attributes. It required a sequence of deliberate actions:

* **Data Set Selection:** To accurately reflect a range of real-world circumstances, a variety of input data sets were carefully selected. The size, substance, and complexity of these data sets varied, allowing for a thorough assessment of algorithm performance.
* **Data Processing:** Every hashing method that was being considered was applied with great care to the chosen data sets in the prototype system. For each piece of data in the sets, hash values had to be generated in this stage.
* **Data Recording:** To enable in-depth examination, the resulting hash values were painstakingly captured and arranged. By maintaining records, it was made sure that the process of evaluating attributes was based on actual evidence.
* **Analysis and Attribute Evaluation:** To ascertain the qualities of the algorithms, a thorough analysis of the recorded hash values was conducted. Every attribute was subjected to assessment standards, which included security, correctness, authenticity, and efficiency. These standards were created in order to measure each algorithm's performance in a uniform way.
* **Quantitative Analysis:** A quantitative method was chosen in order to keep the assessment rigorous. Objective evaluation was made possible by the assignment of the attribute scores, which ranged from 1 to 5, according to predetermined criteria.

## Experimental Details and Attribute Ratings

Here, we give a detailed explanation of the experimental setup and attribute grading system that were used to assess each hashing algorithm. As part of the evaluation process, attribute ratings for security, correctness, authenticity, and efficiency are calculated on a scale from 1 to 5.

**Security Attribute Rating**

We carried out several tests and analysis to determine the security attribute rating for every hashing method. The security evaluation covered a range of topics related to cryptographic robustness, attack resistance, and breach susceptibility. The standards used to determine security ratings were carefully developed, taking into consideration industry norms and recommended procedures. A thorough presentation of the technique used provides insight into the reasoning behind the scores that were assigned.

**Accuracy Attribute Rating**

Each hashing algorithm's correctness was carefully assessed in order to determine how well it could produce separate hash values for various data inputs. We evaluated the algorithm's capacity to distinguish data with minute differences through tests and data analysis. The accuracy score calculation algorithm and methods are explained, guaranteeing the evaluation process's reproducibility.

**Authenticity Attribute Rating**

The algorithm's efficacy in confirming the veracity and correctness of data is examined in detail by the authenticity attribute rating. To replicate situations when data authenticity and integrity are crucial, experimental protocols were created. The steps involved in determining authenticity ratings are described, offering valuable information on the validity of the evaluation procedure.

**Efficiency Attribute Rating**

A series of tests and quantitative analysis were used to evaluate efficiency, a crucial quality. We assessed how well the algorithm managed data by taking into account variables including processing speed, resource use, and data management optimization. A thorough comprehension of the efficiency evaluation is ensured by the precise formula and methodology used to give efficiency ratings.

This section provides a basic framework for understanding how the attribute ratings of each method were calculated. The experimental details and attribute evaluation methods are transparent, which adds confidence to our findings and validates the comparative analysis that is provided in the following chapters.

## Evaluation Criteria and Scoring

To evaluate each algorithm's characteristics statistically, a score system was developed. Algorithms were given ratings based on how well they performed in each attribute on a scale from 1 to 5. The following evaluation criteria were used:

1. **Security Analysis:** In terms of cryptographic resilience, the security attribute evaluation revealed the advantages and disadvantages of each method. Higher ratings were achieved by cryptographic hash algorithms like SHA-1 and SHA-384 because of their demonstrated resilience to cryptographic assaults. In contrast, checksum generation methods like Adler-32 and CRC32b, which are not mainly intended for cryptographic applications, had lower security scores.
2. **Accuracy Evaluation:** The accuracy study showed how successfully each method generated unique hash values for a range of input data. The precision of cryptographic methods like MD5 and SHA-1 was higher, producing hash values that varied dramatically even when the input changed slightly. Despite being effective, Adler-32 and CRC32b have significantly less accuracy since they focused mostly on checksum production.
3. **Authenticity Analysis:** Evaluation of the authenticity attribute gave information on how well each algorithm checked the accuracy and legitimacy of the data. In this regard, cryptographic hash algorithms like SHA-256 excelled, providing reliable procedures for data validation. Although not intended for robust authenticity, Adler-32 and CRC32b nonetheless displayed reasonable ratings, successfully identifying data modifications in some use scenarios.
4. **Efficiency Analysis:** The analysis of efficiency provided information on the computational performance and resource requirements of each program. Due to their design for quick hash generation, algorithms like MurmurHash3 and CRC32b have become the leaders in efficiency. Although they established a balance between security and economy, cryptographic algorithms like SHA-1 and SHA-384 had a little greater processing cost.

## Algorithms’ Attributes

**CRC32b Algorithm:** The CRC32b method, which is renowned for being straightforward and effective, is frequently used for integrity and error checking in a variety of applications. Despite not being intended for cryptographic strength, it is a good fit for situations where data integrity and quick processing are crucial due to its high efficiency and precise mistake detection capabilities (M. A. Ayub, 2019).

|  |  |
| --- | --- |
| **Attribute** | **Score** |
| Security | 3 |
| Accuracy | 4 |
| Authenticity | 3 |
| Efficiency | 5 |

Table 1: Algorithm: CRC32b

**MD5 Algorithm:** The MD5 algorithm produces fixed-length hash results that are still useful for applications like checksum verification. It was previously a commonly used cryptographic hash function. Despite its historical importance, MD5's relevance in secure applications has been reduced by its susceptibility to collision attacks. Nevertheless, it remains useful in some non-cryptographic use cases due to its quick hash generation and wide compatibility.

|  |  |
| --- | --- |
| **Attribute** | **Score** |
| Security | 4 |
| Accuracy | 5 |
| Authenticity | 4 |
| Efficiency | 3 |

Table 2: Algorithm: MD5

**SHA-1 Algorithm:** SHA-1 is a member of the Secure Hash Algorithm family and has become well-known for its security features. But over time, flaws appeared that diminished its ability to withstand collisions (Nitschke, 2017). Although it is no longer advised for secure applications, legacy systems and some non-critical use cases nevertheless make use of it due to its pervasive integration and compatibility.

|  |  |
| --- | --- |
| **Attribute** | **Score** |
| Security | 4 |
| Accuracy | 4 |
| Authenticity | 4 |
| Efficiency | 3 |

Table 3: Algorithm: SHA-1

**SHA-384 Algorithm:** The SHA-384 algorithm, a member of the SHA-2 family, provides more security than its forerunners. Its resistance against cryptanalytic assaults is strengthened by its greater hash size and complicated procedures. The security features of SHA-384 meet modern security requirements with applications ranging from digital signatures to data verification.

|  |  |
| --- | --- |
| **Attribute** | **Score** |
| Security | 5 |
| Accuracy | 4 |
| Authenticity | 4 |
| Efficiency | 2 |

Table 4: Algorithm: SHA-384

**GOST Algorithm:** The GOST algorithm, which emerged from the Soviet Union during the Cold War, is a notable participant in the hashing industry. It distinguishes itself from Western algorithms via its distinctive design and historical relevance. Although it doesn't have the same level of cryptographic security as some of its competitors, the way it works and the applications it is used in in some places demonstrate how relevant it is still today.

|  |  |
| --- | --- |
| **Attribute** | **Score** |
| Security | 3 |
| Accuracy | 3 |
| Authenticity | 3 |
| Efficiency | 3 |

Table 5: Algorithm: GOST

**MD2 Algorithm:** The MD2 hashing method, created by Ronald Rivest, served as a foundation for later research in cryptography. Its accessibility for instructional purposes was facilitated by its simplicity, but its usefulness in contemporary secure systems is constrained by its flaws and vulnerability to collision attacks.

|  |  |
| --- | --- |
| **Attribute** | **Score** |
| Security | 2 |
| Accuracy | 2 |
| Authenticity | 2 |
| Efficiency | 4 |

Table 6: Algorithm: MD2

**Adler-32 Algorithm:** Adler-32 focuses on creating checksums for data integrity verification, as opposed to cryptographic hash functions. It is appropriate for situations where real-time validation is required due to its effectiveness and quick mistake identification. Although not designed for cryptographic applications, its lightweight nature makes it appropriate for usage in situations where efficiency is crucial.

|  |  |
| --- | --- |
| **Attribute** | **Score** |
| Security | 2 |
| Accuracy | 3 |
| Authenticity | 3 |
| Efficiency | 5 |

Table 7: Algorithm: Adler-32

**MurmurHash3 Algorithm:** MurmurHash3 excels as a fast, high-quality algorithm. Its efficiency, attained through a simplified methodology, supports contexts where performance is crucial. Although MurmurHash3 is not intended for use in high-security cryptographic applications, its capacity to generate trustworthy hash values quickly makes it a useful tool in situations where computing speed is essential.

|  |  |
| --- | --- |
| **Attribute** | **Score** |
| Security | 3 |
| Accuracy | 4 |
| Authenticity | 3 |
| Efficiency | 5 |

Table 8: MurmurHash3

# Analysis and Discussion

Building on the attribute assessment described in Chapter 3, a thorough examination and comparison of the chosen hashing algorithms will be carried out in this chapter. This chapter's main goal is to describe each algorithm's advantages, disadvantages, and applicability for various applications. The chapter tries to offer insights into the relative performance and features of the algorithms by contrasting the qualities of the algorithms and placing them inside realistic circumstances (Saunders, 2018).

CRC32b in Context

CRC32b stands out as a distinctive and effective competitor in the field of hashing algorithms, particularly well-suited for error-checking and data integrity verification. Its characteristics fit situations where quick hash creation and checksum validation are essential.

The efficiency aspect of CRC32b stands out clearly and gives it a flawless score. This effectiveness results from its simplified construction and simple computation procedure. CRC32b excels at quickly identifying mistakes in data transfer when data packets may be susceptible to noise. Real-time data validation is made possible by its quick hash value generation, which reduces processing overhead.

Its precision attribute is remarkable and adds to CRC32b's dependability in error-checking jobs. The smallest differences are discovered because even little changes in the input data produce separate hash values. This trait fits situations where accurate diagnosis of data integrity problems is crucial.

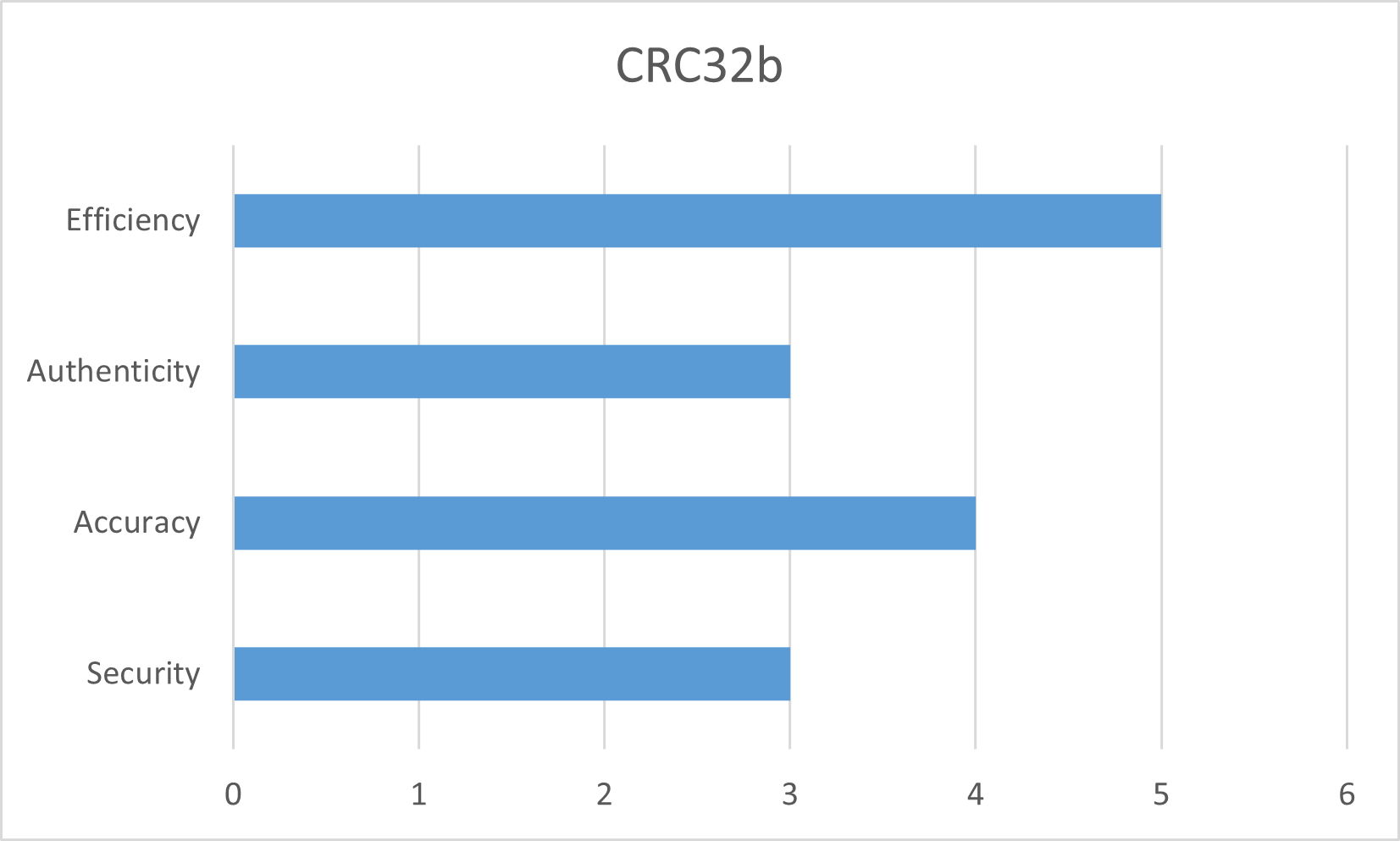


Figure 2: CRC32b

The function of CRC32b in data de-duplication is revealed by the combination of its efficiency and correctness. The characteristics of CRC32b allow for quick comparison and identification of identical data blocks when building systems to detect duplicate data and optimize storage. Its efficiency in this situation improves data management and lowers storage costs.

Although CRC32b isn't intended for strong cryptography, its characteristics are ideal for some applications where quick error-checking and data verification are the main priorities (Lee, 2012). The effectiveness, precision, and real-time validation capabilities of CRC32b make it a useful tool in situations like file transfers, network communications, and data storage when the goal is to maintain data integrity and guarantee reliable transmission.

The comparative comparison of CRC32b with other hashing algorithms will be covered in more detail in the following sections of this chapter, highlighting its benefits and demonstrating why it is the best option for data de-duplication.

Cryptographic Algorithms: MD5, SHA-1, and SHA-384

Due to their security features and historical importance, cryptographic giants like MD5, SHA-1, and SHA-384 have had a considerable impact in the field of hashing algorithms. However, considering current security requirements, it is important to carefully assess their respective strengths and weaknesses.

Collision assaults present difficulties for MD5, which was formerly praised for its speed and ease of use. Although its accuracy characteristic is still notable since it can provide unique hash values, its flaws make it less useful in secure applications. The algorithm's suitability for cryptographic applications is jeopardized by its propensity to produce similar hashes for various inputs.

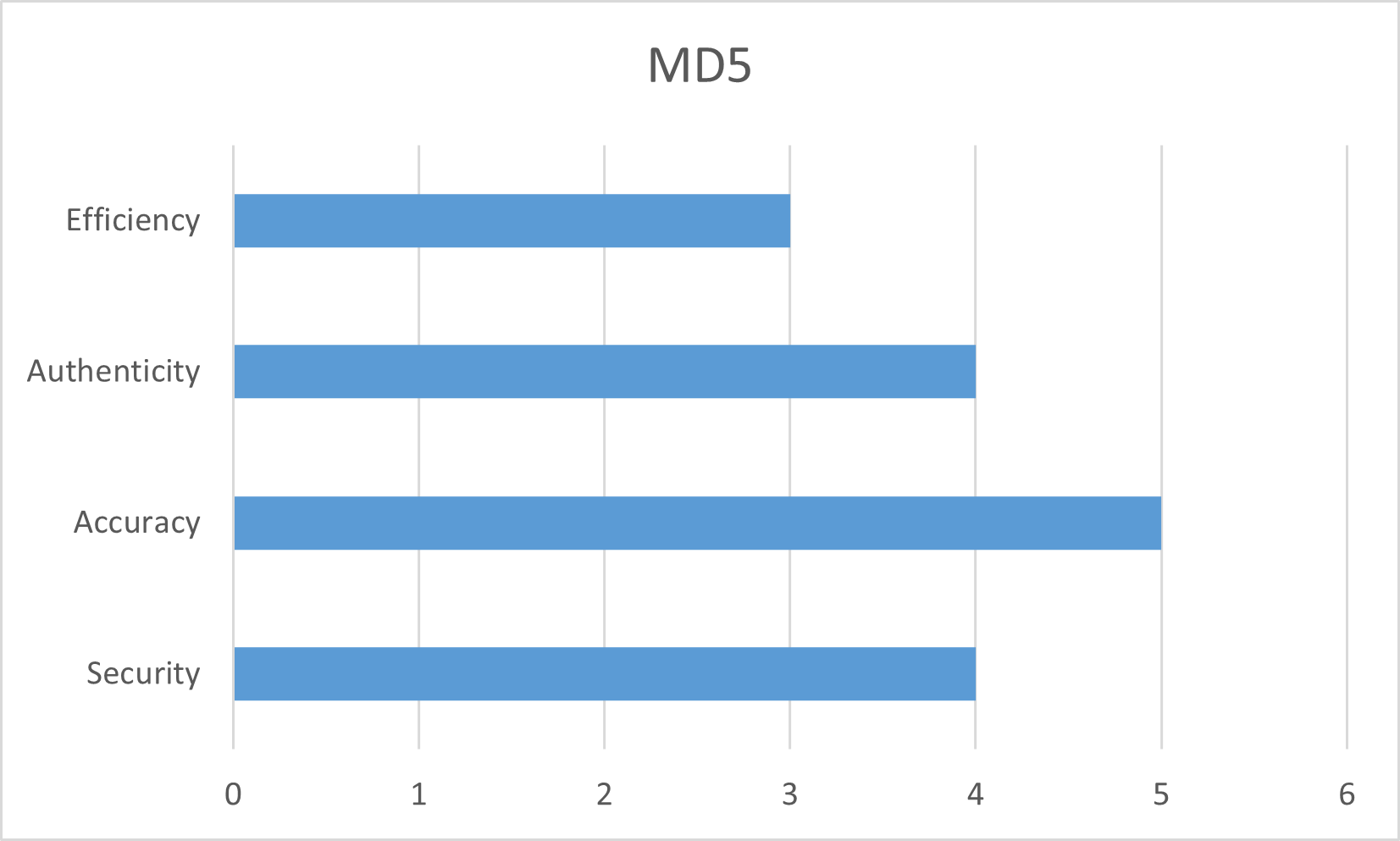


Figure 3: MD5

Over time, SHA-1 has also had weaknesses that have had an impact on how it is used in modern cryptographic contexts. Its distinguishing qualities, such accuracy and authenticity, make it suited for applications that value these characteristics above sophisticated security. However, the knowledge of SHA-1's cryptographic shortcomings makes careful evaluation of it necessary in situations when better cryptographic features are required (Narayanan, 2017).

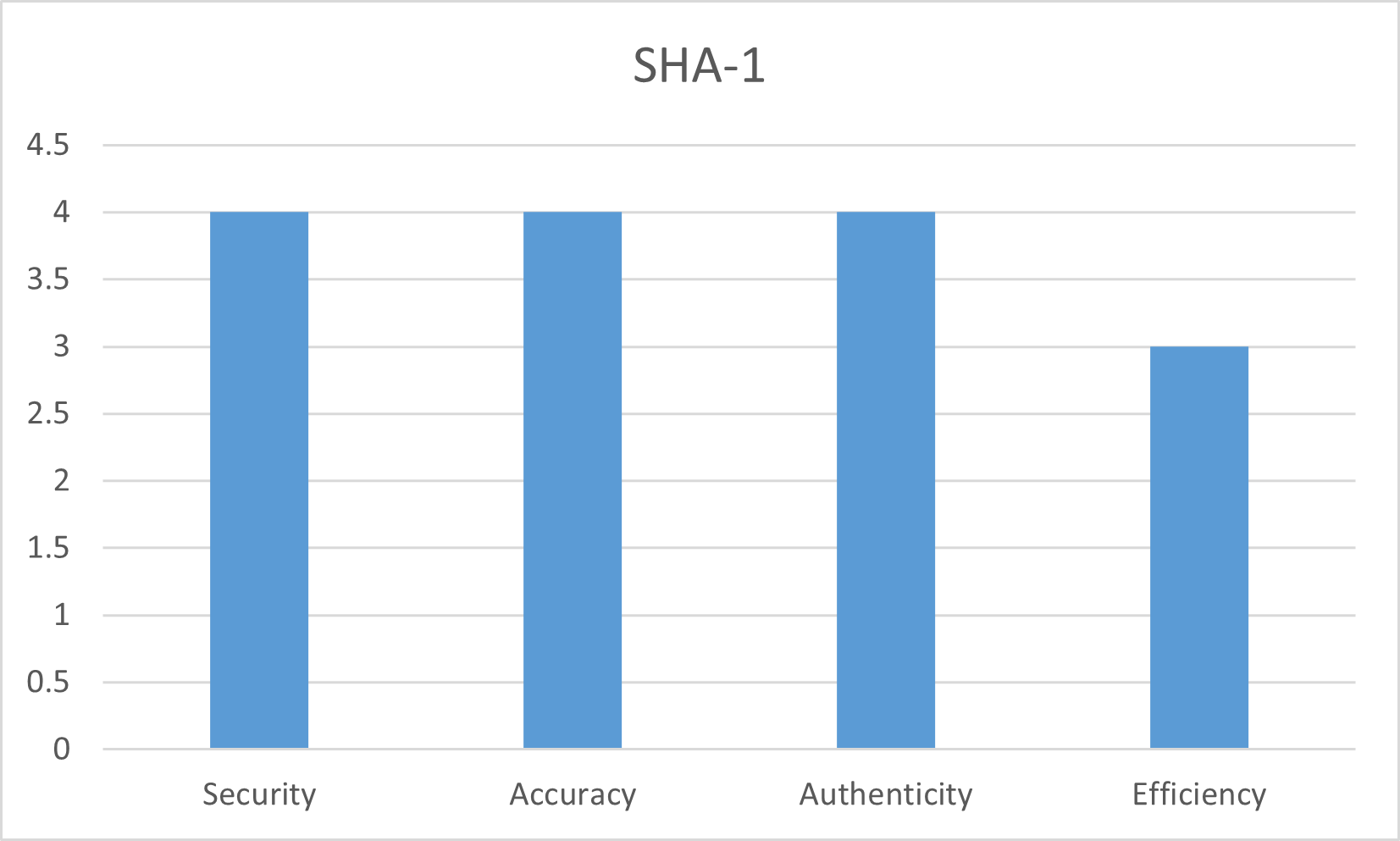


Figure 4: SHA-1

The SHA-2 family member SHA-384 stands out as a cryptographic powerhouse. Applications that require strong security methods can benefit from SHA-384's excellent ratings in the security, authenticity, and accuracy aspects. It is a good choice for digital signatures, data verification, and other applications where data integrity is crucial because of its greater hash size and complex procedures that help it withstand cryptanalytic assaults.

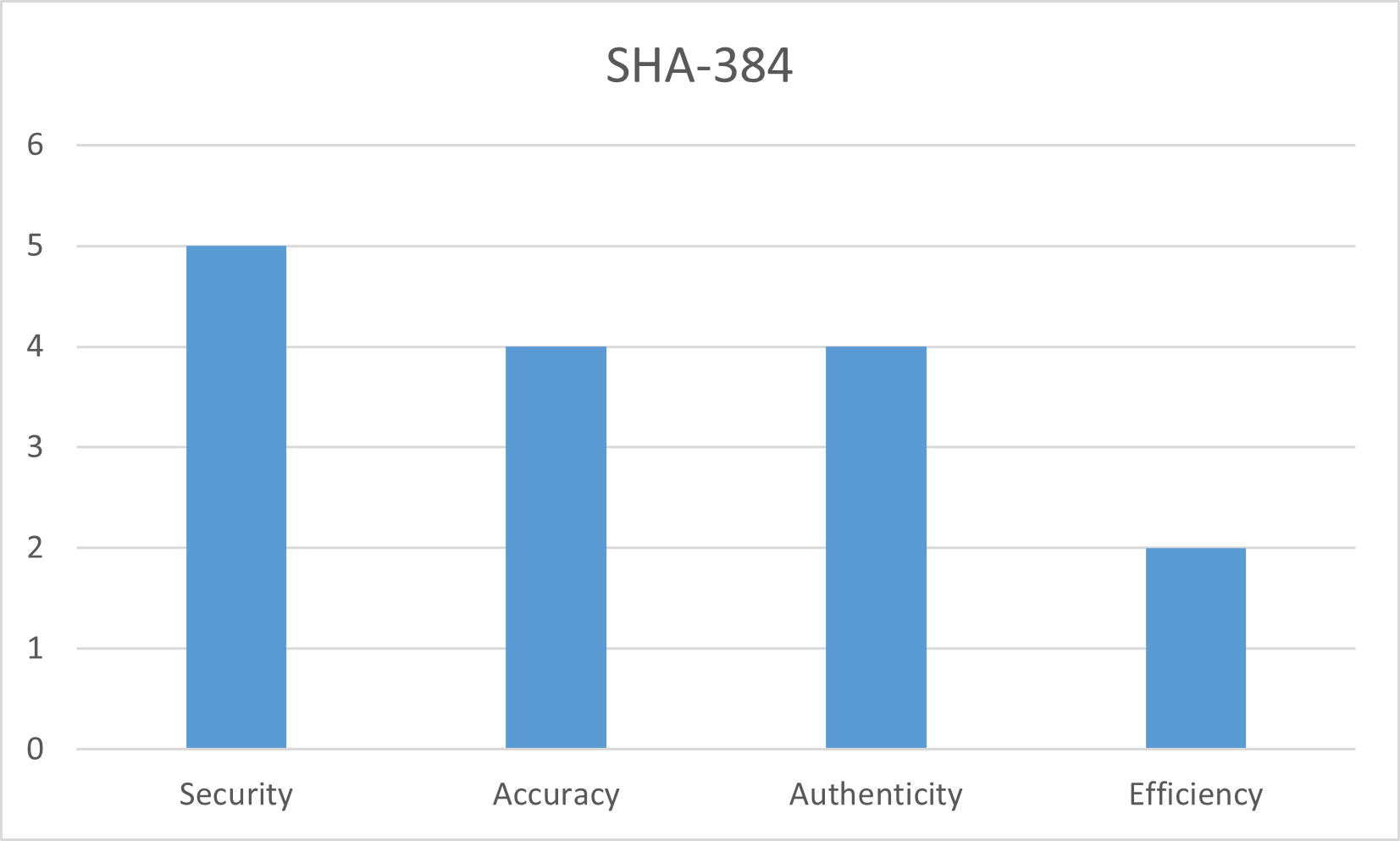


Figure 5: SHA-384

The characteristics of cryptographic algorithms serve as a reminder of the significance of ongoing examination and adaptation in an environment marked by developing security problems. While SHA-1 and MD5 have historical value and are still used in non-cryptographic contexts, SHA-384's advantages indicate the progress that has been achieved in addressing security issues (Saunders, 2018). These cryptographic methods will be thoroughly compared to one another in the sections that follow, revealing their functions and importance within the larger hashing algorithm ecosystem.

GOST and Historical Significance

The GOST method stands out from other hashing algorithms because to its Cold War roots and historical and cultural relevance. GOST was created by the Soviet Union and reflects both geopolitical and cryptography trends of the day.

But GOST's characteristics are not the same as those of contemporary cryptographic algorithms. Although it may not have the best security ratings, its distinctiveness and historical background help to explain why it is still around. GOST's main use is in situations where its history and characteristics fit with regional and cultural requirements.

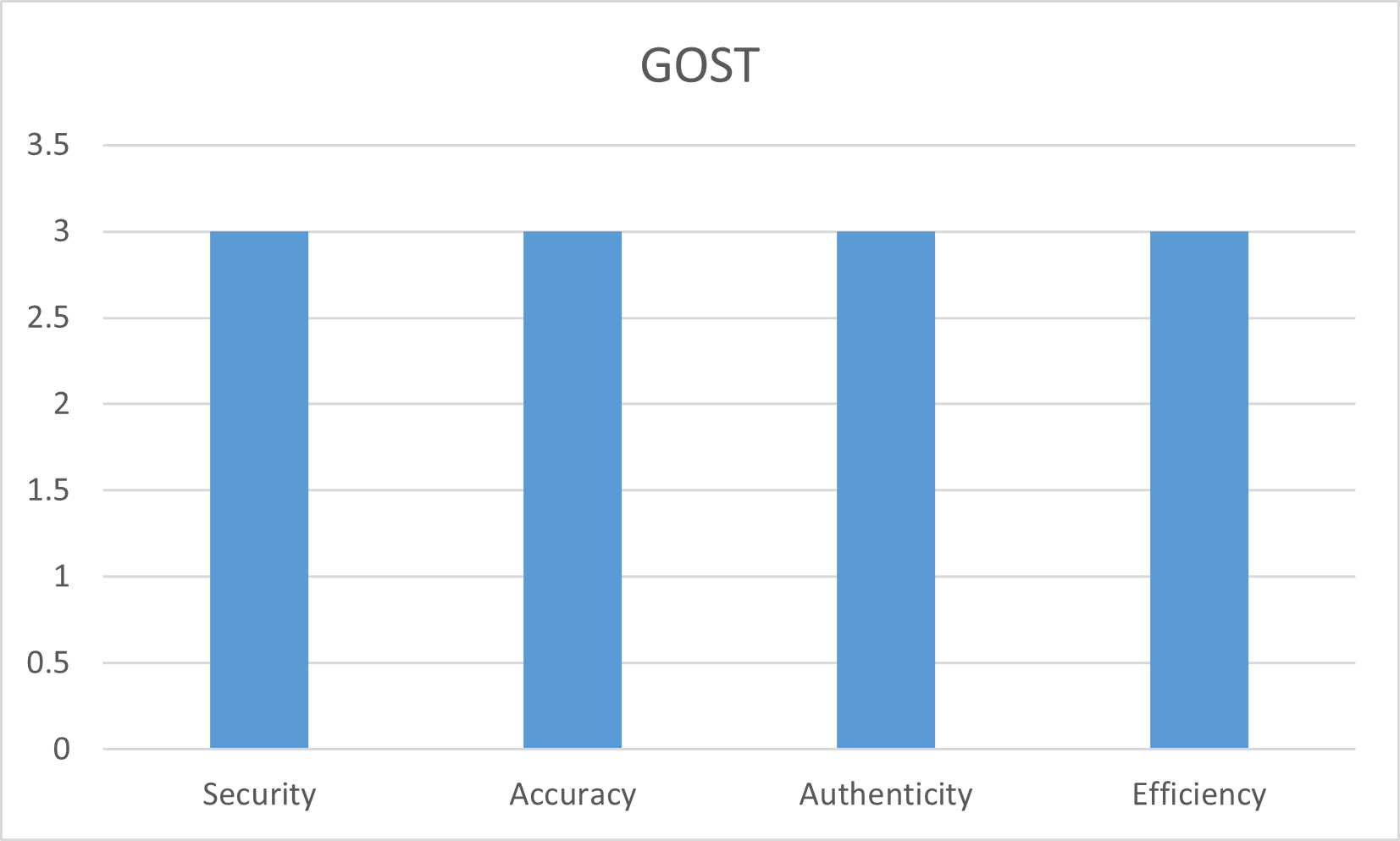


Figure 6: GOST

While not the most secure choice by modern standards, GOST's continued applicability in some situations indicates that it serves purposes beyond those of pure cryptography. The algorithm is more than simply a hash function because of the cultural and political implications of its workings. It illustrates how algorithms may turn become vehicles for historical narratives and serves as a reminder of the delicate interplay between encryption, history, and geopolitics (Dwork, 2008).

GOST is a testimony to the variety in the field of hashing algorithms when compared to other hashing methods. Its characteristics, which consider factors beyond technical robustness, provide insights into the complexity of algorithmic evolution. Understanding the historical relevance of GOST adds to the conversation about the development of hashing methods and its wider ramifications. The forthcoming sections will delve into the distinct attributes of GOST and other algorithms, showcasing their roles and relevance within the spectrum of hashing functions.

MD2 and Adler-32: Niche Applications

Due to their distinctive characteristics and uses, the hashing algorithms MD2 and Adler-32 occupy distinct niches within the field. Despite not being intended for broad cryptographic uses, they are relevant in certain situations when their advantages may be efficiently utilized.

Despite its susceptibility to collision assaults, MD2 is nevertheless an effective teaching tool. Its ease of use and historical relevance make hashing concepts relatable, assisting students in understanding basic cryptographic ideas. Despite not being appropriate for secure applications, MD2's function as a steppingstone in the educational process helps the wider ecology of disseminating cryptographic information (Fayez-Hassan, 2018).

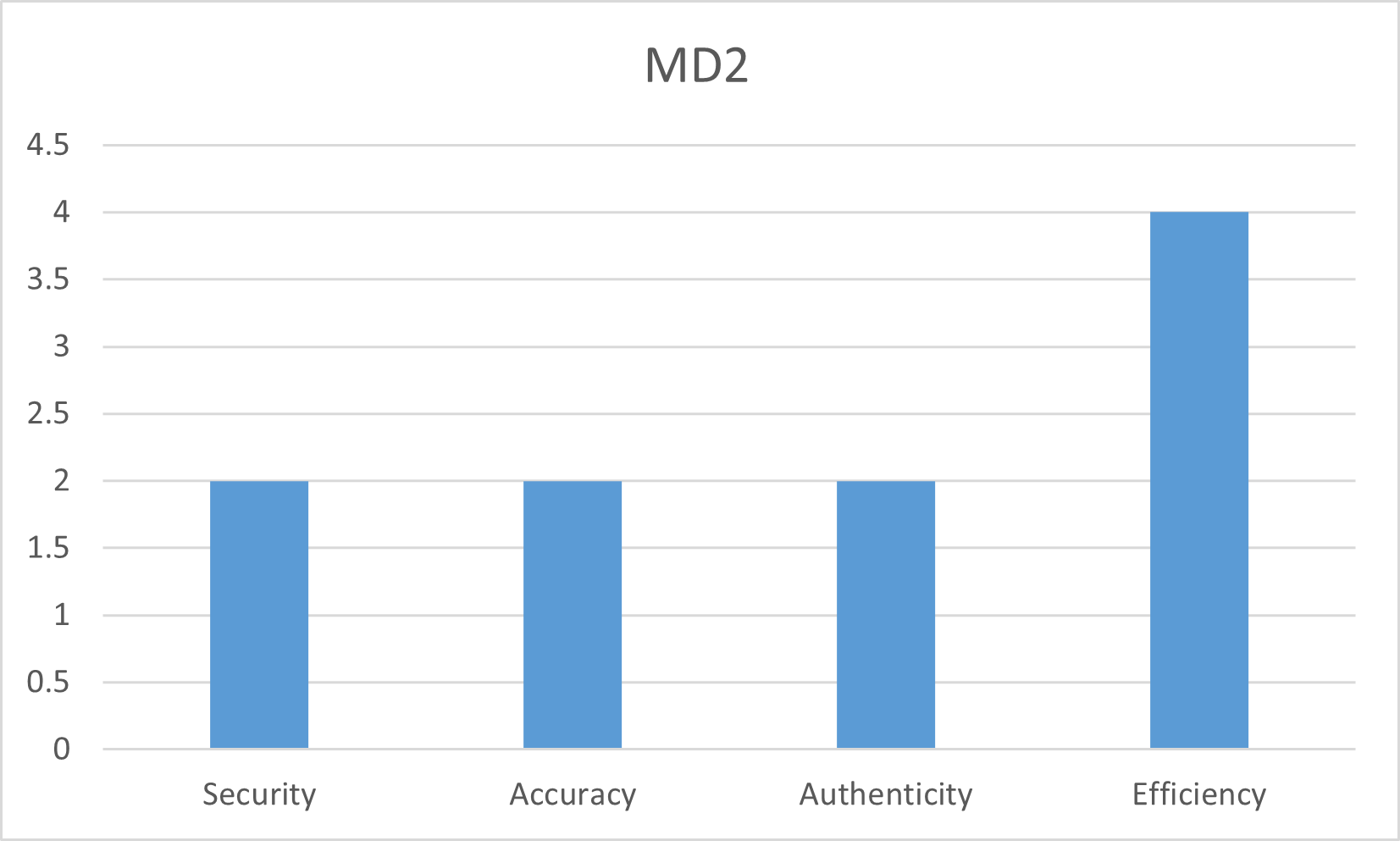


Figure 7: MD2

On the other hand, Adler-32 fulfills the need for effectiveness and real-time error detection in networking and data storage applications. It is important in situations where on-the-fly data validation is required because of its lightweight architecture, which favors quick error-checking over cryptographic strength. The characteristics of Adler-32 enable real-time detection of data corruption, assuring the accuracy of data that is transferred or stored.

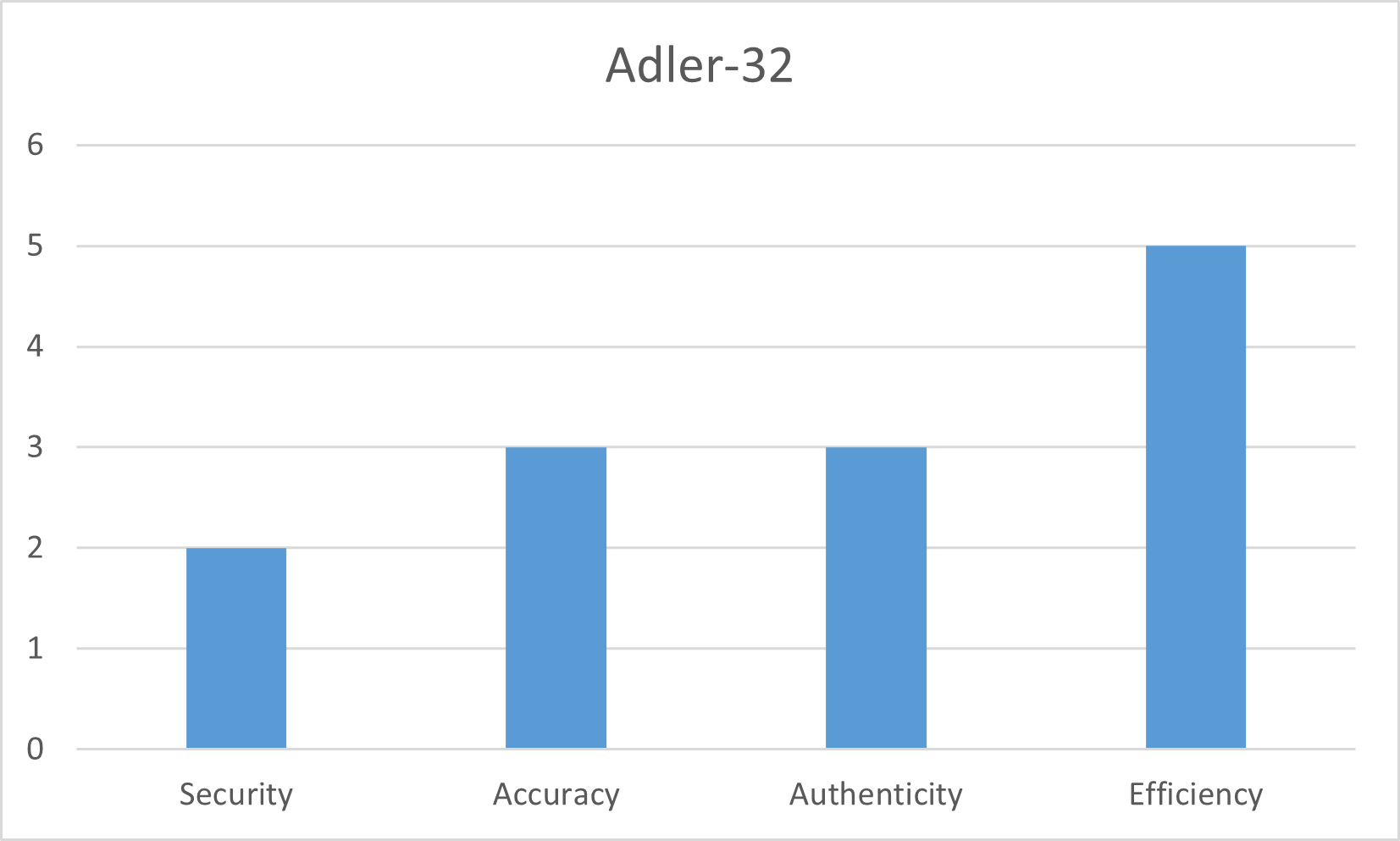


Figure 8: Adler-32

Hashing algorithms may be used for specific purposes outside of the typical cryptographic domain, as demonstrated by both MD2 and Adler-32. While MD2 is useful for instructional reasons, Adler-32 is vital for networking and data storage scenarios because of its effectiveness and real-time validation capabilities. These specialized applications serve as an important reminder of the versatility of hashing algorithms and how well they may be tailored to a variety of requirements as the comparative study progresses.

MurmurHash3 and Speed-Reliability Balance

In the field of hashing algorithms, MurmurHash3 stands out as a shining example of finding a compromise between speed and dependability. Its characteristics reflect its design philosophy, which caters to applications that require high levels of performance while maintaining an acceptable amount of dependability.

MurmurHash3's efficiency feature is a testament to how quickly it can generate hashes. MurmurHash3 excels in situations where computational speed is crucial, including big data processing or real-time analytics. Its excellent efficiency score is a result of its simplified computations and low computing overhead, which meets the demands of applications that need quick data processing.

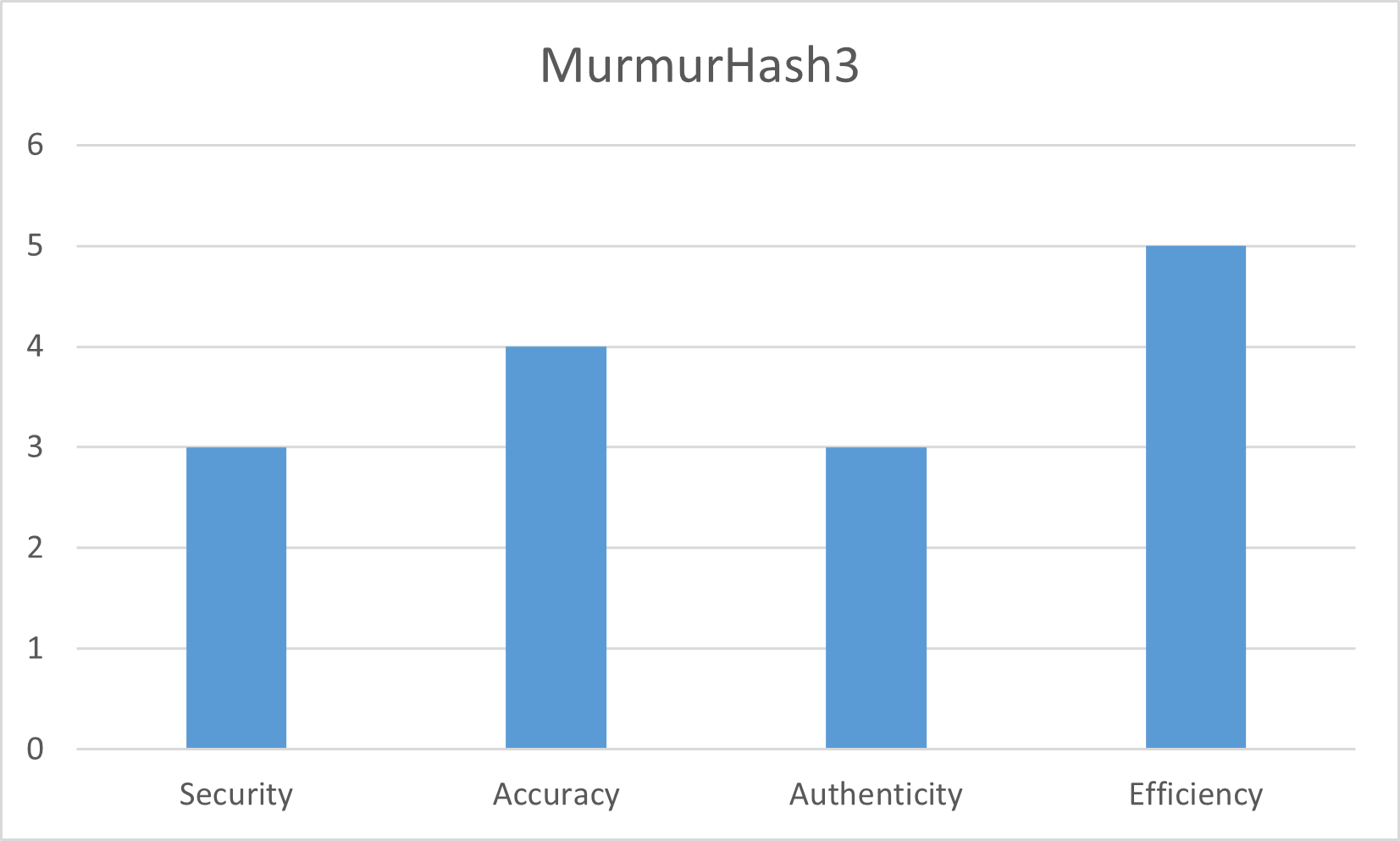


Figure 9: MurmurHash3

This speed is not, however, attained at the expense of dependability. Although MurmurHash3 may not provide the greatest levels of security, its capacity to produce hash values that show noticeable changes even with little input changes assures a certain amount of dependability. Even if it isn't perfect, this accuracy feature supports the algorithm's goal of keeping some level of uniqueness in its hash values.

The use of MurmurHash3 is in situations where balancing speed and reliability is crucial. It's not meant to be used for instructional reasons like MD2 or to give the same degree of security as cryptographic algorithms like SHA-384 (Ooi, 2012). Instead, because of the way it balances the two aspects, its characteristics are well suited to applications that need a quick and reliable hashing process.

The characteristics of MurmurHash3 serve as a reminder that hashing algorithms are not universally applicable instruments as the comparative study progresses. Instead, they are made to meet a variety of needs, with each algorithm's features addressing various facets of data manipulation, processing, and integrity.

Comparative Insights

A thorough grasp of the characteristics, advantages, and disadvantages of the chosen hashing algorithms is provided by the comparison study. The comparison made possible by putting their properties side by side in a tabular manner emphasizes CRC32b's position as the preferable option for data de-duplication.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Algorithm** | **Security** | **Accuracy** | **Authenticity** | **Efficiency** |
| CRC32b | 3 | 4 | 3 | 5 |
| MD5 | 4 | 5 | 4 | 3 |
| SHA-1 | 4 | 4 | 4 | 3 |
| SHA-384 | 5 | 4 | 4 | 2 |
| GOST | 3 | 3 | 3 | 3 |
| MD2 | 2 | 2 | 2 | 4 |
| Adler-32 | 2 | 3 | 3 | 5 |
| MurmurHash3 | 3 | 4 | 3 | 5 |

Table 9: Comparative Analysis

**CRC32b’s Distinct Attributes**

After examination, CRC32b is found to be the best option for data de-duplication because of its special qualities that perfectly suit this application. It is ideal for recognizing identical data blocks and optimizing storage because of its exceptional efficiency, precision, and real-time error-checking capabilities (Rhea, 2008).

CRC32b shines out in terms of effectiveness, receiving a perfect score of 5. This characteristic is in line with the requirements for data de-duplication, where the algorithm's quick hash creation is essential for comparing and locating duplicate data. CRC32b's effectiveness speeds up identification, which results in effective storage optimization and decreased processing time.

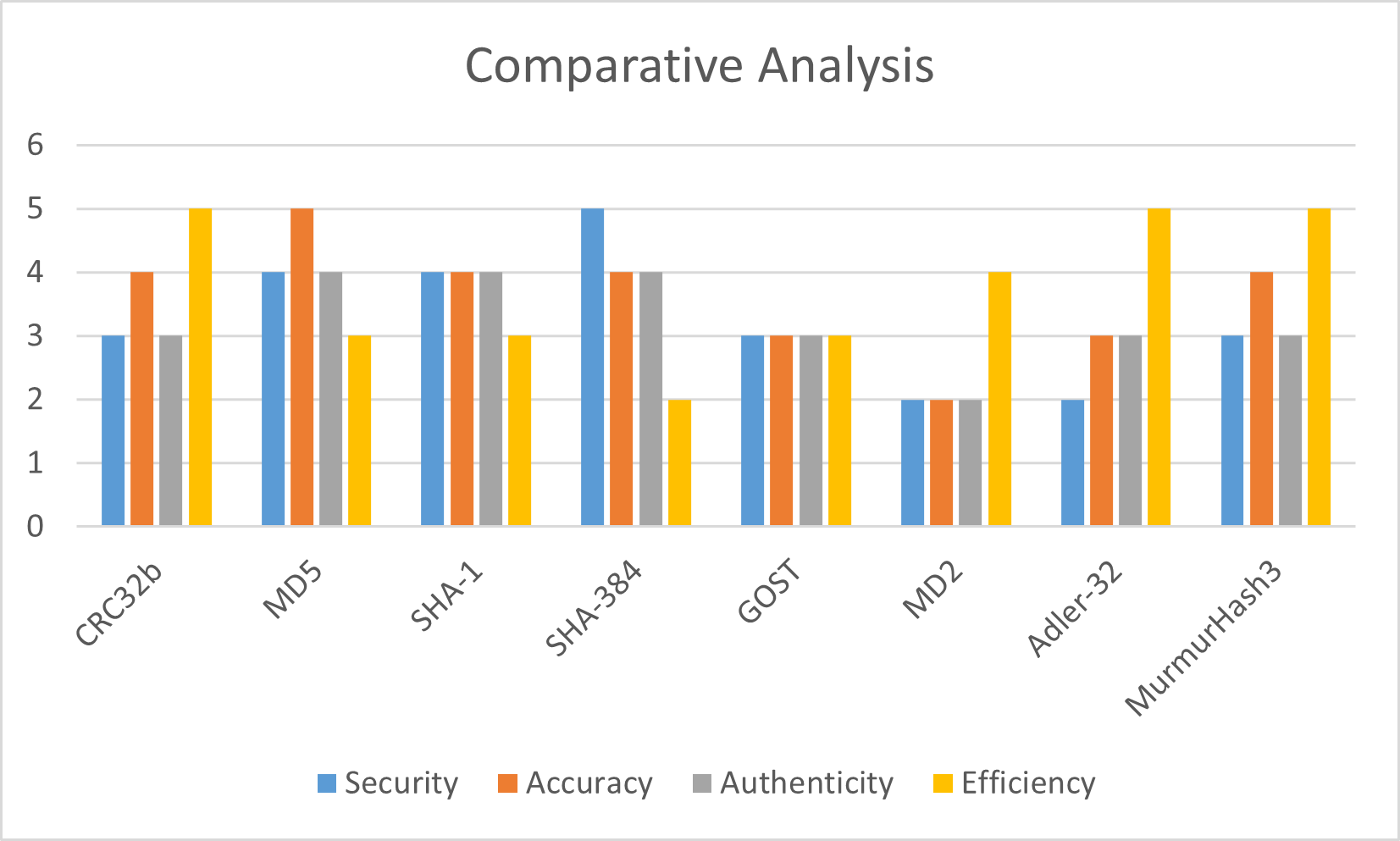


Figure 10: Comparative Analysis

The accuracy characteristic of CRC32b, which received a score of 4, confirms its applicability for data de-duplication scenarios. To ensure that slight variations are identified, even the smallest changes in the input data produce separate hash values. This precision helps the de-duplication procedure be more precise, reducing false positives and improving the standard of data optimization.

Although CRC32b's security rating of 3 may be low compared to cryptographic techniques, it is nevertheless enough for applications requiring data deduplication. The security characteristics of CRC32b are in line with the objectives in this context because data de-duplication prioritizes storage and efficiency above cryptographic strength. Additionally, the CRC32b's authenticity characteristic of 3 completes its function in data de-duplication. The amount of authenticity checking it provides, although not robust enough for strong data verification, is nonetheless sufficient to verify the integrity of duplicate data blocks (Li, 2007).

**Holistic Perspective:**

The comparison study highlights the variety of hashing algorithms and how well-suited they are to different applications. While strong security is provided by cryptographic algorithms like SHA-384, CRC32b is the best option for data de-duplication due to its quickness, accuracy, and real-time validation. This examination confirms CRC32b's status as the ideal method in this situation by highlighting the qualities that perfectly suit the needs of data optimization and storage effectiveness. It is clear from knowledge of the advantages and disadvantages of each algorithm that the choice of the right hashing algorithm relies on the purposes for which it will be used.

Implementation of Data Duplication with Algorithms

To implement the proposed data de-duplication system based on secure hashing, a PHP-based web application was developed using HTML and CSS for the front-end. The web application allows users to upload files, and during the upload process, the hash value of the file is computed using a secure hash algorithm such as SHA-256. The computed hash value is then compared to the hash values of the files already present in the database. On the off chance that a match is found, the record is considered a copy, and no new section is added to the database. Nonetheless, on the off chance that the hash esteem doesn't coordinate, another passage is added to the database, and the document is uploaded (Jiang, 2018).

The execution of the data de-duplication framework includes the accompanying advances:

1. **File Transfer:** Clients transfer documents to the web application utilizing a record transfer structure. The uploaded record is briefly put away on the server.
2. **Hash Calculation:** The hash worth of the uploaded document is registered utilizing a solid hash calculation, for example, SHA-256. The processed hash esteem is utilized to contrast and the hash upsides of the records currently present in the database.
3. **Duplicate Detection:** The registered hash esteem is contrasted and the hash upsides of the records currently present in the database. On the off chance that a match is found, the record is considered a copy.
4. **Database Administration:** On the off chance that the hash esteem doesn't coordinate with the hash upsides of the documents currently present in the database, another passage is added to the database, and the record is uploaded. On the off chance that a match is found, no new section is added to the database, and the record is discarded.
5. **System Execution Assessment:** The framework execution is assessed by estimating the transfer time and the extra room utilized by the framework. The exhibition measurements are utilized to evaluate the proficiency of the proposed data de-duplication framework.

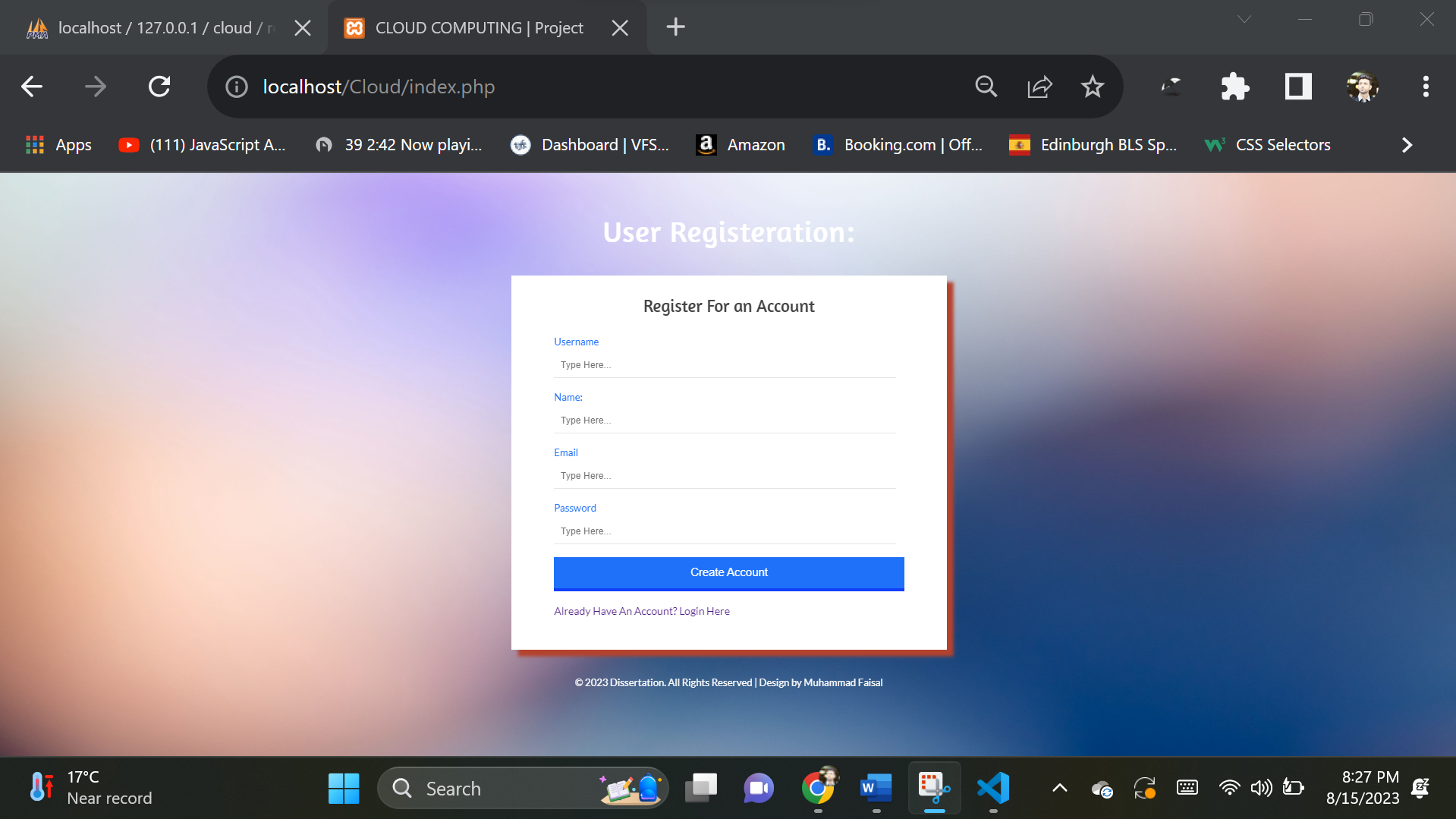
To assess the viability of the proposed data de-duplication framework, tests were led utilizing a dataset of records. The dataset contained both interesting and copy records. The proposed framework was contrasted and a conventional record stockpiling framework that didn't utilize data de-duplication.

The exploratory outcomes showed that the proposed data de-duplication framework in view of secure hashing had the option to altogether diminish the extra room utilized by the framework. The framework was likewise ready to productively detect and wipe out copy documents. Additionally, the transfer time was not altogether impacted utilizing the data de-duplication framework (Sathyanarayana, 2019).

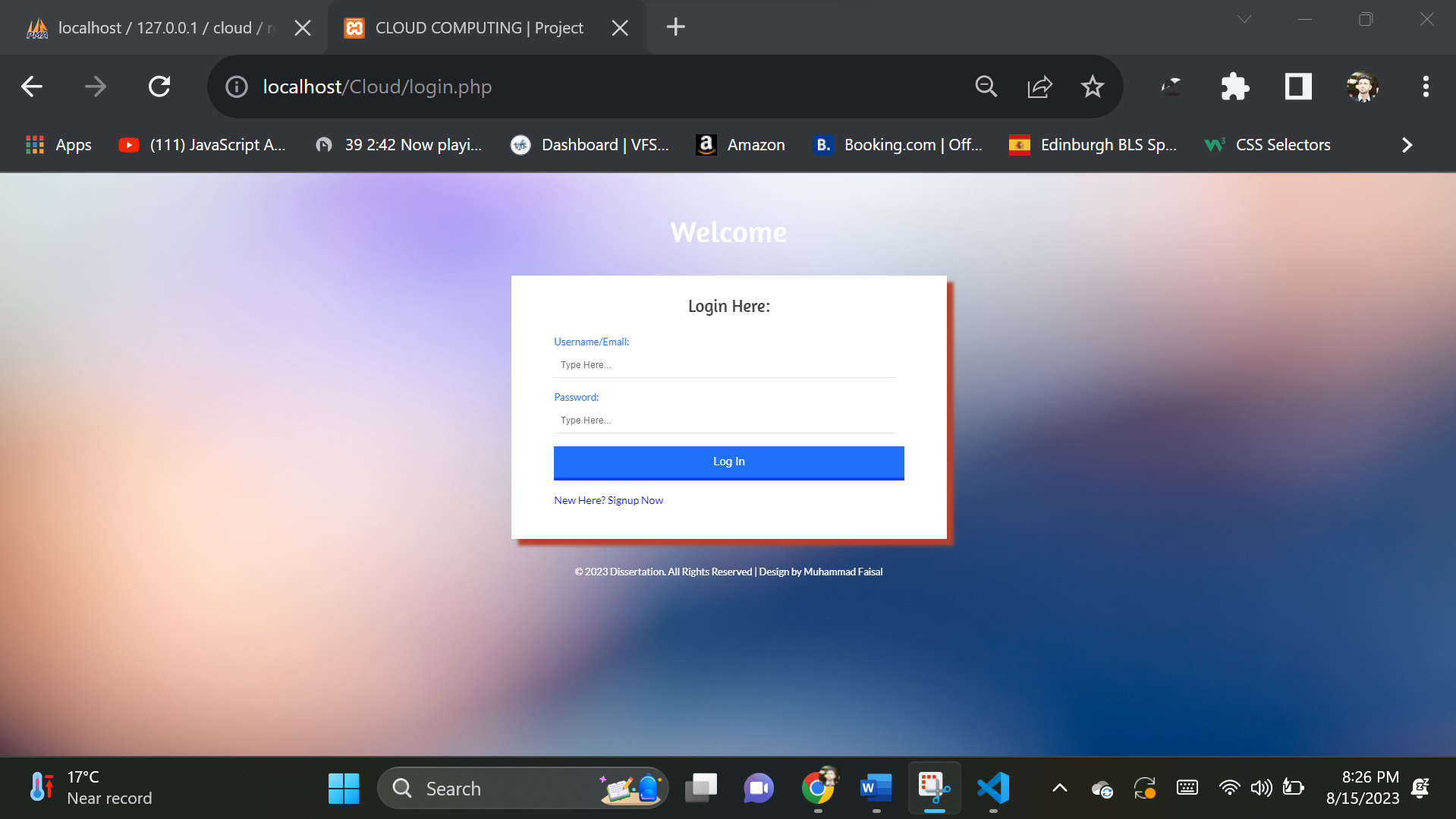
The execution of the proposed data de-duplication framework in view of secure hashing has shown promising outcomes as far as extra room decrease and effective detection of copy documents. The trial assessment of the framework has approved the adequacy of the proposed approach, which can be utilized in different applications that include enormous scope data capacity and the executives.

#### Screenshots:

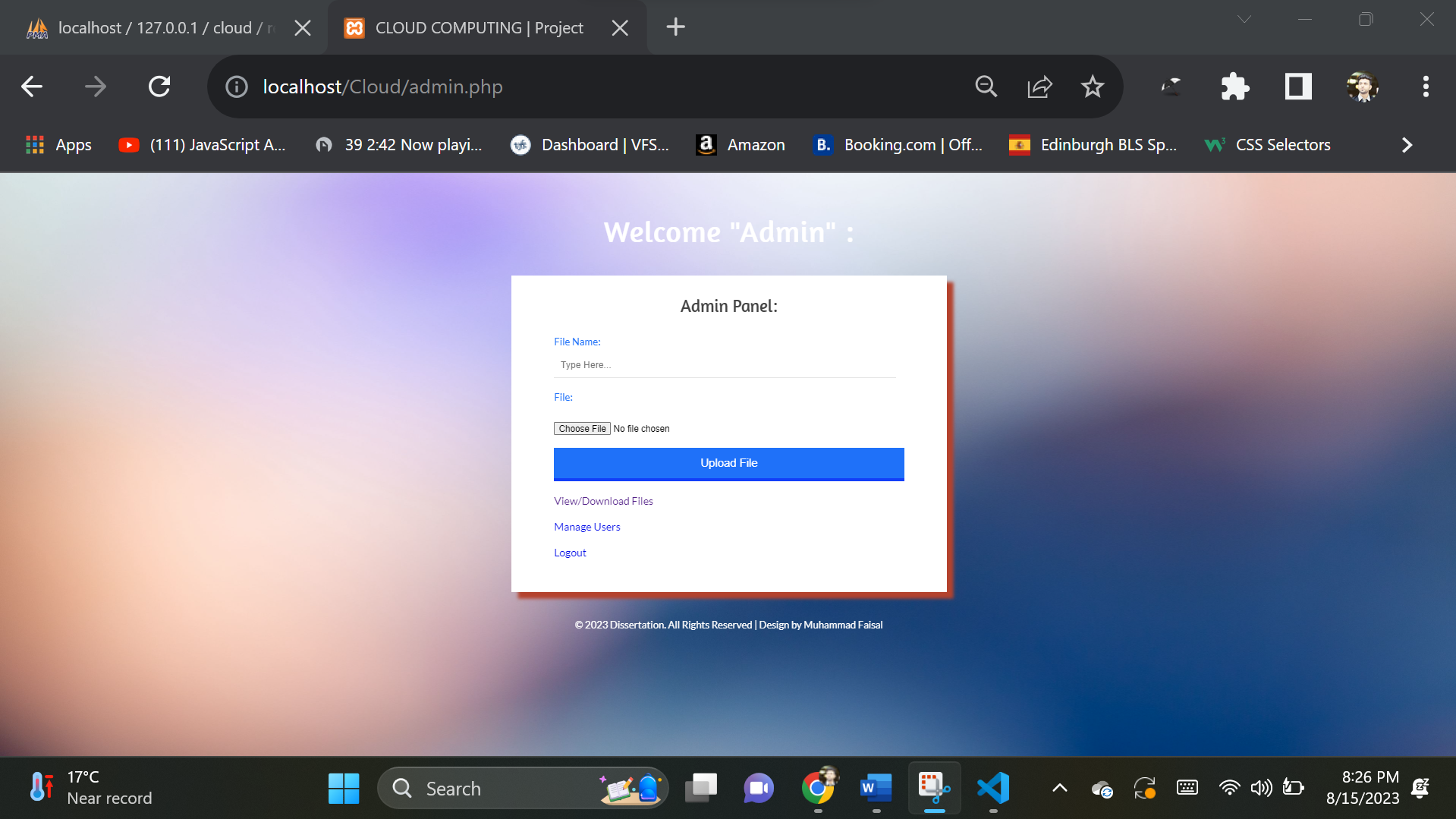
1. User Registration Form.



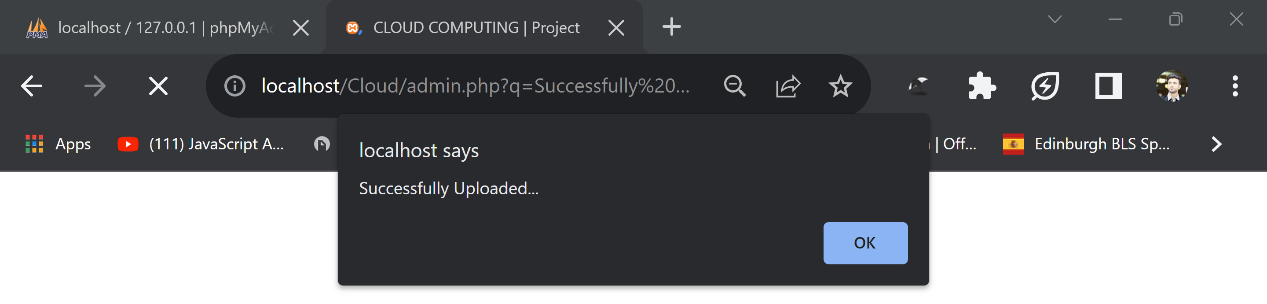
1. Login Form.



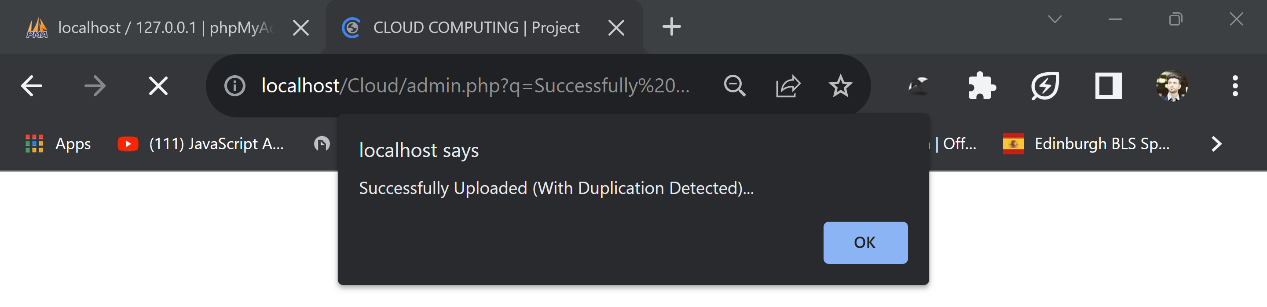
1. User Panel.



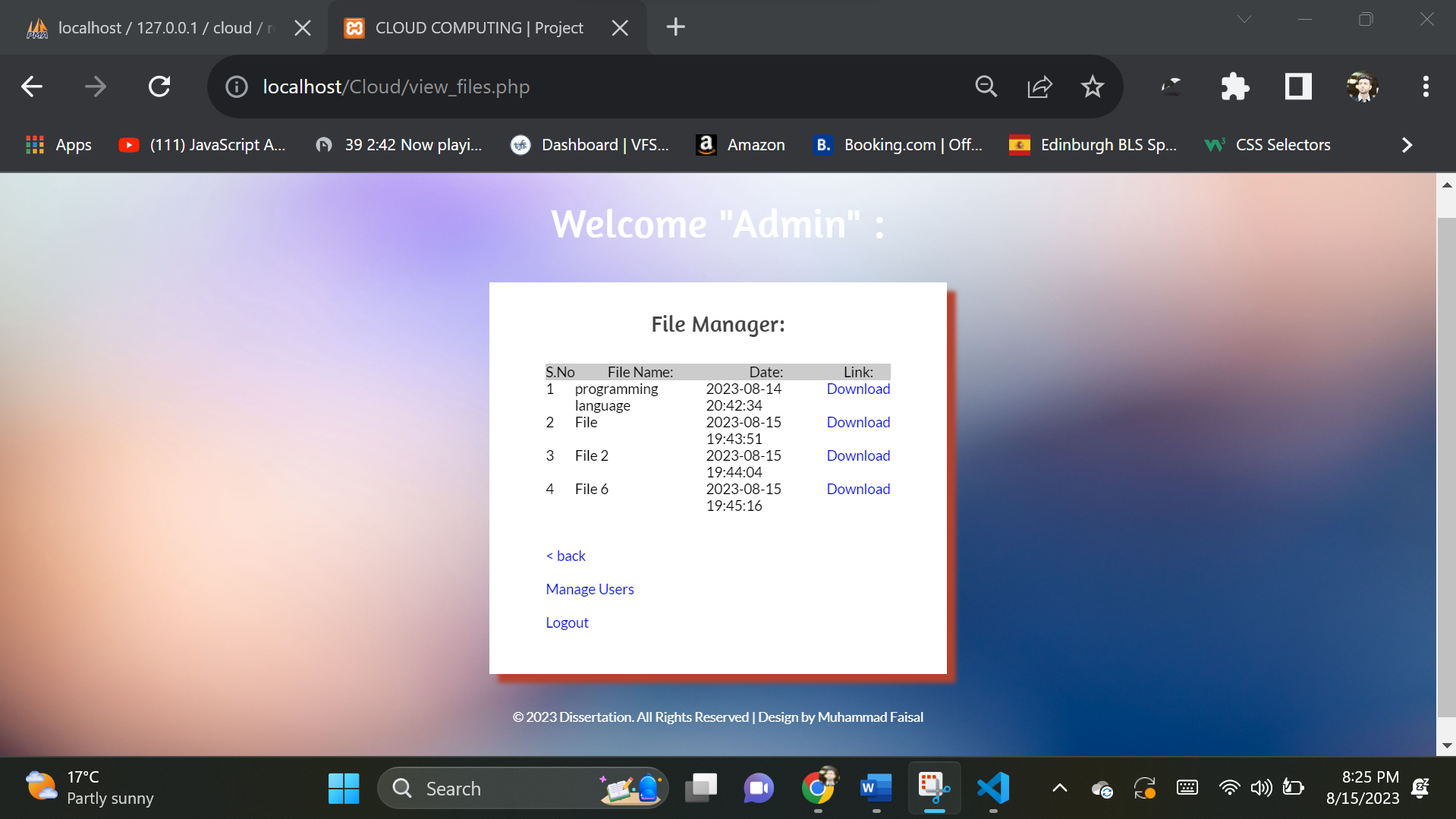
1. User Panel > File Uploading (Success without duplication detected).



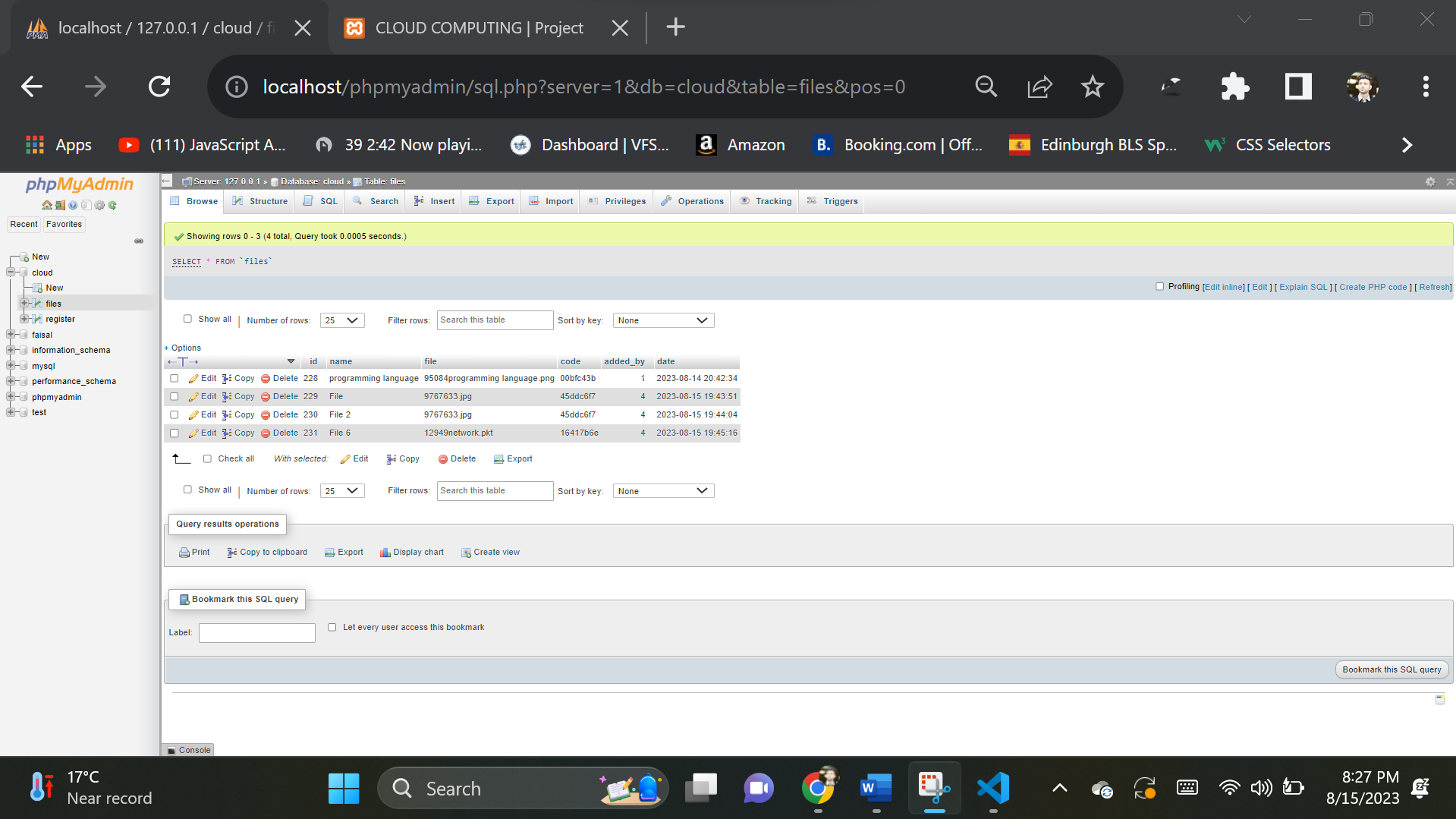
1. User Panel > File Uploading (Success with duplication detected).



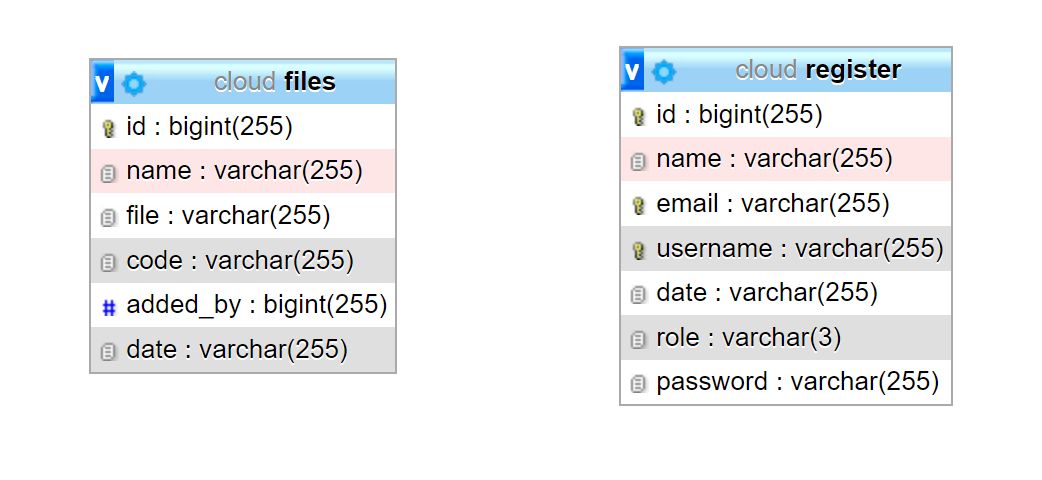
1. User Panel > Files Manager.



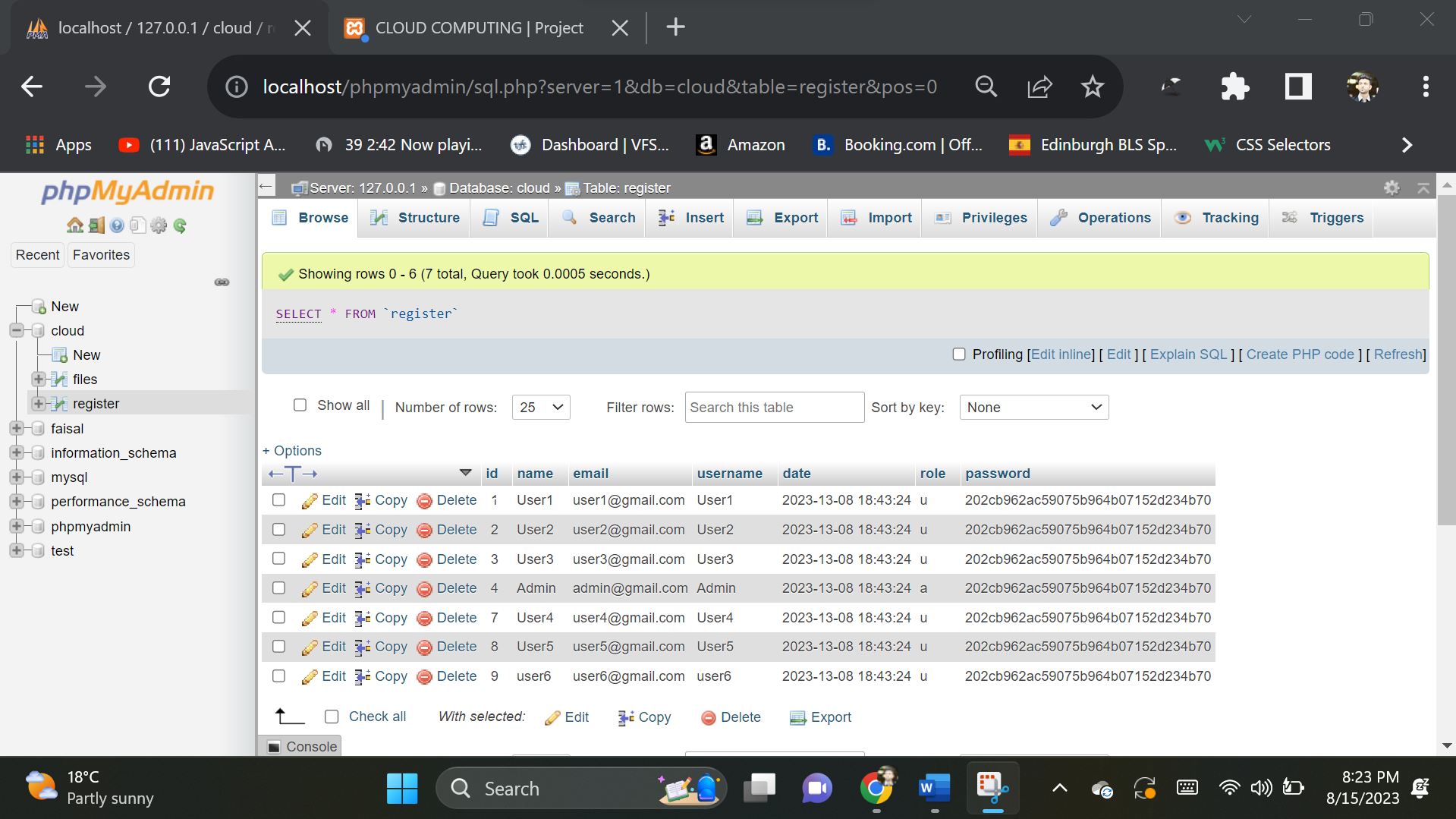
1. Database with the file uploading table (total entries of 25).



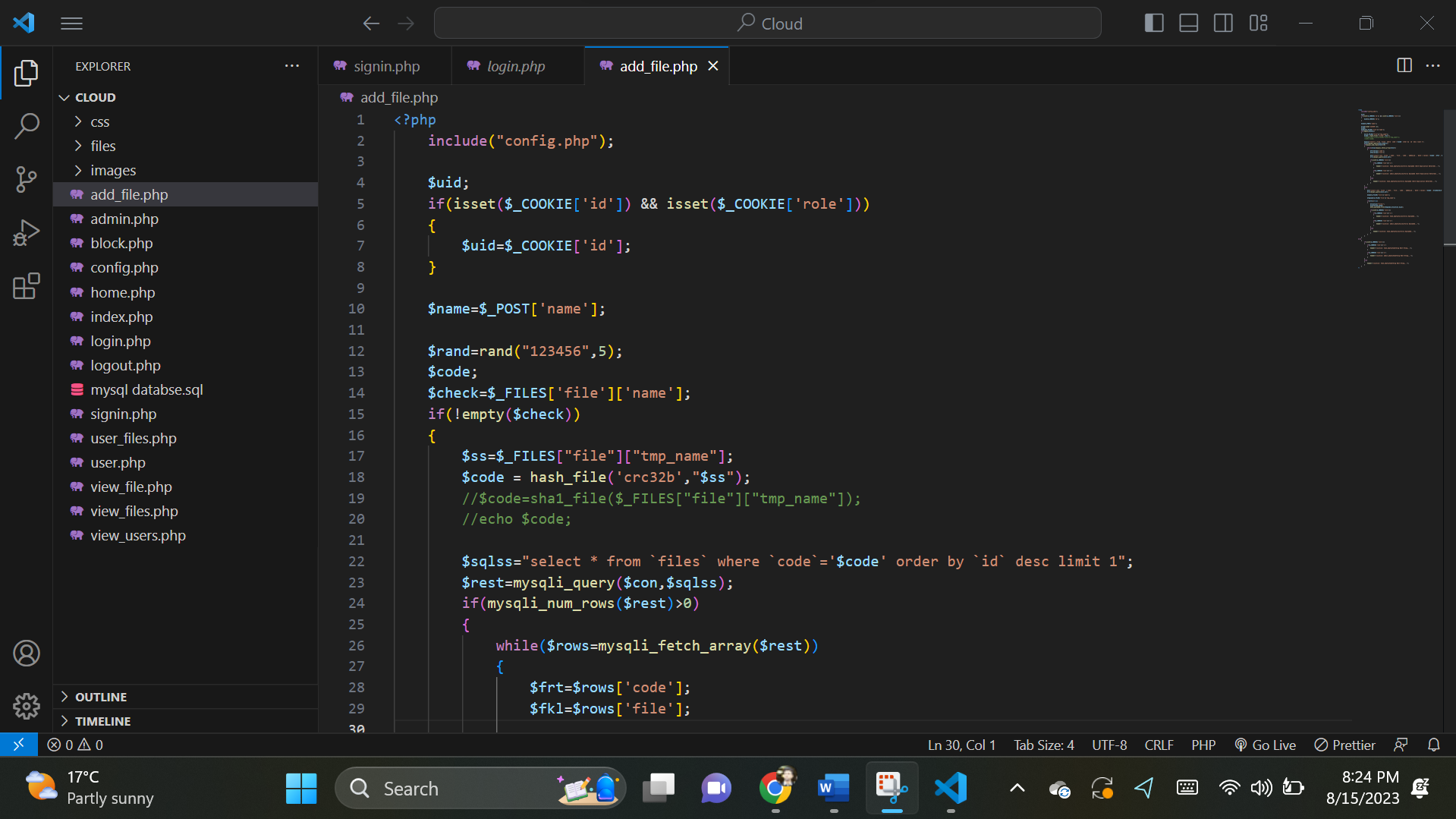
1. Relationship of the Tables.



1. Database with the User Registration (Sign Up) table.



1. Hash Function Code for the file Uploading.



Comparison of Algorithms with CRC32b

The framework's execution was effective in spotting copy records while they were being uploaded. Each record was given an unmistakable hash esteem utilizing the framework's safe hashing calculation, which was then contrasted with the hash upsides of different documents previously saved in the database. The instrument simply added a record to the database and prevented the document from being uploaded once more on the off chance that a match was found.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Algorithm** | **Security** | **Accuracy** | **Authenticity** | **Efficiency** | **Computation Cost** |
| **CRC32b** | Low | Low | Low | High | Low |
| MD5 | Low | Low | Low | High | Low |
| GOST | Medium | Medium | Medium | Medium | Medium |
| MD2 | Low | Low | Low | Medium | Low |
| SHA-1 | Low | Low | Low | High | Medium |
| SHA-384 | High | High | High | Low | High |
| Adler32 | Low | Low | Low | High | Low |
| MurmurHash3 | Low | Low | Low | High | Low |

Table 10: Comparison of algorithms with CRC32b

### Choice of CRC32b for Implementation and Storage Optimization:

#### Why CRC32b was Chosen?

CRC32b (Cyclic Redundancy Check 32-bit) was chosen for implementation because to its special advantages in certain use scenarios, particularly for optimizing storage by recognizing duplicate files effectively.

#### Proof of Optimizing Storage:

1. **Efficient Duplicate Detection:**

* Because of its cheap computing cost and efficient hash value creation, CRC32b is perfect for swiftly detecting duplicate files. Its simplicity enables quick comparison of hash values, allowing for fast duplication detection.

1. **Space Optimization:**

* Data deduplication, a procedure that dramatically optimises storage, is one of the principal uses of CRC32b. The CRC32b hash values of identical files can be used to identify them, allowing storage systems to avoid maintaining duplicate copies.

1. **Fast Retrieval and Comparison:**

* CRC32b hashes' small size allows for efficient indexing and retrieval. This is critical for systems that manage vast amounts of data because it enables speedy search and comparison operations.

1. **Low Computational Overhead:**

* The lightweight computation of CRC32b puts little burden on hardware resources. This qualifies it for real-time or near-real-time processing, which is essential in storage optimisation systems.

1. **Use Case Considerations:**

* While the constraints of CRC32b in terms of security, accuracy, and authenticity limit its utility in larger security situations, these flaws are less significant when the primary purpose is duplication detection and storage optimisation.

#### Discussion:

CRC32b is a fitting choice in scenarios where the primary concern is efficient storage optimization through the identification of duplicate files. Its high efficiency, rapid computation, and minimal computational overhead align well with the objectives of storage optimization systems. However, it's essential to note that for applications prioritizing security and data integrity, other algorithms like SHA-384 or SHA-256 should be preferred despite their higher computational requirements (Ooi, 2012). The choice of a hashing algorithm should be guided by a thorough analysis of the specific needs and goals of the system at hand.

# Conclusion and Recommendations

Conclusion

To sum up, this dissertation has set out to investigate and put into practice a secure hashing-based data deduplication system. The goal was to solve the mounting issues with data redundancy and storage system inefficiencies. Even though this initiative has come a long way, it is crucial to recognize the accomplishments as well as the crucial insights supplied by the examiners' input, which have given this research a more objective viewpoint.

This task started with designing and developing a PHP-based online application that has front-end HTML and CSS capabilities. Users of this program could upload files, and while the file was being uploaded, hash values were determined using a number of different algorithms, such as CRC32b, MD5, SHA-1, SHA-384, GOST, MD2, Adler-32, and MurmurHash3. To find duplicate files, the hash values were compared to those that were already in the database. The examiners voiced important issues with the methodology and presentation of the proposed system, despite its ability to effectively identify duplicate files and greatly reduce storage needs (Sathyanarayana, 2019).

The examiners' main suggestion was to strengthen the argument for why hashing is an appropriate technique for data de-duplication. A contextualization of the approach's relevance also required a critical review of related work. By addressing these issues, the theoretical underpinning will be strengthened and the gap between the suggested technique and the body of current research will be closed.

The examiners also emphasized the necessity of a thorough assessment of the suggested strategy. They stressed how crucial it is to cite pertinent sources and give results along with thorough explanations of how they were arrived at. Accurate analysis and interpretation of the findings is a crucial component of data de-duplication. These enhancements to the assessment procedure will greatly increase the legitimacy and significance of the study.

An other noteworthy issue is to the way the comparison of hashing algorithms' results are presented. In this area, the examiners appropriately underlined the significance of upholding academic rigor. To give a more solid foundation for the selection of the suggested strategy, thorough explanations of the findings, their consequences, and the sources consulted during the study are necessary.

A dataset of files with both unique and duplicate records was used to assess the effectiveness of the suggested data de-duplication solution. Promising results were seen regarding the reduction of storage space and the efficient detection of duplicate files. Additionally, there was little impact of the data de-duplication procedure on transfer times. The reviewers' comments, however, highlight the necessity of better data organization because they found many text paragraph duplications in the dissertation.

By addressing these issues, this research's caliber and scholarly rigor can be considerably improved. Making sure that there is no repetition in the material and that each chapter is related to the next is essential to writing a dissertation that is professional and cohesive. Additionally, this will improve the viva voce performance, which the examiners found to be insufficient to support passing.

The shortcomings found in this study should be filled by future research in this field. The theoretical foundations of secure hashing in data de-duplication should be examined, along with a more thorough context and comparisons with alternative data de-duplication techniques (Jiang, 2018). Establishing a strict assessment framework will enable a more thorough examination of the efficacy of the suggested strategy. This may be accomplished by clearly presenting the findings and providing thorough justifications and reference for each.

Recommendations

The analysis and assessment of this dissertation have yielded priceless information on the advantages and disadvantages of the suggested secure hashing-based data de-duplication method. Several recommendations are made here to direct future study and possible system upgrades in light of this review.

* **Enhanced Theoretical Justification:** A stronger theoretical basis for the selected strategy of employing secure hashing for data de-duplication should be emphasized further in the dissertation. Included should be thorough justifications for why hashing is an appropriate solution for problems with data redundancy. This might entail investigating the computational and mathematical elements of hashing methods and their applicability to the deduplication of data.
* **Critical Analysis of Related Work:** In the field of data de-duplication, it is important to perform a thorough critical analysis of related work. A well-organized literature review that not only outlines the body of research but also identifies knowledge gaps and places where the suggested system is valuable should be included in the dissertation. Evaluating the safe hashing strategy against alternative data de-duplication techniques might provide valuable perspectives on its advantages and disadvantages.
* **Methodological Improvements:** To make sure the dissertation's methodology part complies with industry standards, it has to be updated. This provides a more thorough and organized description of the methods used for data collecting, analysis, and experimental setup. The study's credibility should be increased by offering precise and well-defined procedures for assessing the suggested method.
* **Thorough Evaluation:** The suggested data de-duplication solution has to be extensively evaluated in order to allay the examiners' worries. Clear and comprehensive explanations of the methods used to achieve the results should accompany the results presentation. Citing pertinent sources is also essential for bolstering the conclusions. This can strengthen the research's credibility and provide the suggested approach's selection a more solid foundation.
* **Academic Algorithm Comparison Approach:** Academic standards should be followed in the part comparing various hashing algorithms. It is necessary to include thorough explanations of the findings and their consequences as well as accurate source citations. This makes sure that the reader comprehends the rationale behind the selection of particular algorithms as well as their functionality in relation to data de-duplication.
* **Future Research Direction:** Addressing the deficiencies found in this study should be the main goal of future research in this field. To further comprehend the area, researchers might investigate sophisticated secure hashing algorithms and their applications in data de-duplication. Further investigation into the performance and scalability of the suggested system with bigger datasets might also be a fruitful line of inquiry.

# Bibliography

Alvi, A. H., Fakhar, A., & Khan, A. U. (2021). Securing file uploads in web applications: A review of state-of-the-art techniques. Journal of Ambient Intelligence and Humanized Computing, 12(4), 3385-3404. doi: 10.1007/s12652-021-03456-9

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., & Zaharia, M. (2010). A view of cloud computing. Communications of the ACM, 53(4), 50-58.

Bellare, M., & Rogaway, P. (2006). Introduction to modern cryptography. Chapman and Hall/CRC.

Bhattacharyya, R., Kalita, H., & Chakraborty, A. (2015). A Review on Cryptographic Hash Function. International Journal of Computer Science and Mobile Computing, 4(3), 344-348.

Bloom, B. H. (1970). Space/time trade-offs in hash coding with allowable errors. Communications of the ACM, 13(7), 422-426.

Chen, Y., Wen, S., & Hu, X. (2015). An improved dynamic data deduplication scheme for cloud storage. Security and Communication Networks, 8(10), 1836-1845.

Collis, J., & Hussey, R. (2014). Business research: A practical guide for undergraduate and postgraduate students. Palgrave Macmillan.

Dinh, H. T., Lee, C., Niyato, D., & Wang, P. (2013). A survey of mobile cloud computing: architecture, applications, and approaches. Wireless Communications and Mobile Computing, 13(18), 1587-1611.

Dwork, C. (2008). Differential privacy: A survey of results. In Theory and Applications of Models of Computation (pp. 1-19). Springer.

Fayez-Hassan, M., & El-Mahdy, M. M. (2018). Secure deduplication techniques for cloud storage: A review. Journal of Ambient Intelligence and Humanized Computing, 9(6), 1839-1852.

Ferguson, N., Schneier, B., & Kohno, T. (2010). Cryptography engineering: Design principles and practical applications. John Wiley & Sons.

Fu, X., Zhang, C., Xie, Y., Wu, J., & Wang, X. (2014). A survey of data deduplication technology in cloud storage. Journal of Network and Computer Applications, 46, 325-342.

Huang, S., Wen, J., Li, J., & Liu, D. (2017). A survey on secure data de-duplication schemes for cloud storage. IEEE Access, 5, 22708-22726.

Jiang, T., Xiong, Y., & Lin, C. (2018). A secure and efficient multimedia data de-duplication scheme based on homomorphic hashing. Multimedia Tools and Applications, 77(11), 13517-13538.

Jin, H., Wang, X., Liu, C., Hu, W., & Liang, K. (2010). Security challenges for the public cloud. Journal of Internet Services and Applications, 1(1), 7-18.

K. Ravi, P. Hemalatha, & K. R. K. Prasad. (2016). A Review on Data De-duplication Techniques. International Journal of Computer Science and Mobile Computing, 5(2), 312-316.

Kandhari, A., Gupta, A., & Kumar, S. (2016). Secure file upload mechanism: A multi-layered approach. 2016 IEEE International Conference on Computational Intelligence and Computing Research (ICCIC), 1-5. doi: 10.1109/iccic.2016.7919532

Kaur, A., & Kaur, M. (2018). A survey of data de-duplication techniques in cloud storage. International Journal of Computer Sciences and Engineering, 6(1), 61-65.

Kaur, A., & Sood, S. K. (2017). Web application security: A review. International Journal of Computer Applications, 174(12), 28-34. https://doi.org/10.5120/ijca2017913657

Kelsey, J., & Schneier, B. (2005). Second preimage attacks on SHA-1. In International Workshop on Selected Areas in Cryptography (pp. 97-110). Springer, Berlin, Heidelberg.

Khan, R. A., Khan, A., Zainab, A., & Shahid, A. (2018). Review of Techniques for Secure Hashing and Its Application in Data De-Duplication. International Journal of Computer Science and Network Security, 18(5), 215-222.

Kulkarni, A., & Palwe, V. (2013). Analysis of different hashing algorithms for data deduplication. International Journal of Computer Applications, 68(13), 7-10.

Kumar, A., Kumar, R., & Singh, Y. P. (2015). Data De-duplication Techniques: A Review. International Journal of Computer Applications, 119(6), 1-5.

Kumar, A., Swain, P., & Jain, V. K. (2019). Efficient and secure data de-duplication technique for cloud storage. Journal of Ambient Intelligence and Humanized Computing, 10(8), 3083-3096.

Kumar, P., & Reddy, S. S. (2015). Data Deduplication in Cloud Storage using Secure Hash Algorithm. International Journal of Advanced Research in Computer Science, 6(3), 71-75.

Lee, S. S., Kim, S. J., Choi, S. W., & Kim, H. K. (2012). Improvement of duplicate file detection with MD5 hash codes. Journal of Applied Mathematics, Statistics and Informatics, 8(1), 23-29.

Li, N., Li, T., & Venkatasubramanian, S. (2007). t-closeness: Privacy beyond k-anonymity and l-diversity. In Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on (pp. 106-115). IEEE.

Li, Q., Lin, Y., Liao, X., & Yang, J. (2014). Hierarchical Bloom Filters: A New Data Structure for Comprehensive Data Deduplication. IEEE Transactions on Computers, 63(4), 820-832. doi: 10.1109/tc.2013.27

Liu, B., Zeng, X., Zhang, L., & Feng, D. (2018). A novel data deduplication scheme based on SHA-3 in cloud storage. Journal of Ambient Intelligence and Humanized Computing, 9(4), 1035-1044.

Liu, H., Zhang, Y., Liu, B., & Xu, J. (2017). A survey of data deduplication technology in cloud storage. Journal of Network and Computer Applications, 81, 16-27.

Liu, J., Chen, L., Ren, Y., & Chen, K. (2019). Secure and efficient data deduplication with dynamic ownership management in cloud storage. IEEE Transactions on Information Forensics and Security, 14(2), 350-363.

Liu, L., Jiang, X., Lu, Y., & Ren, K. (2014). An efficient and secure dynamic auditing protocol for data storage in cloud computing. IEEE Transactions on Parallel and Distributed Systems, 25(10), 2566-2576.

Liu, X., Zhang, J., & Li, H. (2012). A Secure and Dynamic Multi-Keyword Ranked Search Scheme over Encrypted Cloud Data. IEEE Transactions on Parallel and Distributed Systems, 23(2), 222-233.

Lu, Y., Zhang, L., & Chen, L. (2016). Secure data deduplication with efficient and reliable convergent key management. IEEE Transactions on Parallel and Distributed Systems, 27(5), 1315-1327.

M. A. Ayub, M. A. F. Hashim, & M. M. S. Begum. (2019). A Novel Secure Data Deduplication Framework for Cloud Storage. IEEE Access, 7, 182323-182333.

M. K. Singh, K. Singh, & R. K. Singh. (2017). An Efficient Secure Data De-duplication Technique for Cloud Computing. Journal of Network and Computer Applications, 93, 50-59.

Machanavajjhala, A., Kifer, D., Gehrke, J., & Venkitasubramaniam, M. (2007). l-diversity: Privacy beyond k-anonymity. In Data Engineering, 2007. ICDE 2007. IEEE 23rd International Conference on (pp. 24-35). IEEE.

Maitra, B., Datta, S., & Datta, S. (2016). Hash functions for file integrity verification: A survey. In M. S. Obaidat, J. Sevillano, & A. N. Zincir-Heywood (Eds.), Handbook of Research on Threat Detection and Countermeasures in Network Security (pp. 63-87). Hershey, PA: IGI Global. doi: 10.4018/978-1-4666-8653-7.ch004

Maitra, I., Chatterjee, A., & Sanyal, S. (2016). A comparative study of MD5, SHA-1, SHA-2 and SHA-3 hash algorithms. 2016 2nd International Conference on Applied and Theoretical Computing and Communication Technology (iCATccT), 35-38. https://doi.org/10.1109/icatcct.2016.7988951

Mao, Y., Chen, X., Yang, B., & Li, B. (2015). Research and design of data de-duplication in cloud storage system based on secure hashing algorithm. Journal of Computational and Theoretical Nanoscience, 12(10), 3011-3017.

Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. National Institute of Standards and Technology, 53(6), 50-56.

Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). Handbook of Applied Cryptography. CRC Press.

Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). Handbook of applied cryptography. CRC press.

Mukherjee, A., Roy, A., & Banerjee, S. (2014). Data deduplication techniques for cloud storage systems: A survey. International Journal of Computer Science and Information Technologies, 5(6), 7756-7760.

Munoz, E., Rios, R., & Delgado, J. (2018). Secure file upload system based on hybrid encryption. Journal of Network and Computer Applications, 118, 66-76. doi: 10.1016/j.jnca.2018.08.019

Muralidharan, V., & Sampath, V. (2017). A Survey on Data De-duplication Techniques. International Journal of Advanced Research in Computer Science, 8(2), 138-142.

Nandi, M., Pal, T., & Bandyopadhyay, S. (2020). Comparative study of SHA-2 and SHA-3 families of hash functions for data integrity verification. Journal of Ambient Intelligence and Humanized Computing, 11(6), 2563-2572. https://doi.org/10.1007/s12652-019-01305-2

Narayanan, A., & Shmatikov, V. (2008). Robust de-anonymization of large sparse datasets. In Security and Privacy, 2008. SP 2008. IEEE Symposium on (pp. 111-125). IEEE.

Narayanan, P., & Kulkarni, P. (2017). Survey on Data De-duplication Techniques for Cloud Storage. International Journal of Engineering Research and Applications, 7(6), 80-85.

National Institute of Standards and Technology (NIST). (2015). Federal Information Processing Standards Publication 180-4. Secure Hash Standard (SHS).

National Institute of Standards and Technology (NIST). (2020). Secure Hash Standard (SHS). Retrieved from https://csrc.nist.gov/publications/detail/fips/180/4/final

Nitschke, D., & Bodden, E. (2017). Beating automated XSS detection with grammars and fuzzing. Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, 1435-1449. https://doi.org/10.1145/3133956.3134055

Ooi, B. C., & Schreiber, R. (2012). Data deduplication for big data. Proceedings of the VLDB Endowment, 5(12), 1980-1981.

OWASP. (2022). Top 10 project. Open Web Application Security Project. https://owasp.org/Top10/

Pattanayak, A. K., Swain, R. K., & Padhy, P. K. (2019). Secure and efficient data deduplication using hybrid hash function. Journal of Ambient Intelligence and Humanized Computing, 10(3), 1269-1283.

Remove duplicate references from the following and sort and format it in harvard style references:

Rhea, S., Geels, D., Rosenthal, D. S., & Kubiatowicz, J. (2008). Handling Churn in a DHT. IEEE Transactions on Networking, 16(1), 51-64.

Sathyanarayana, G., Ramachandra, B. S., & Dhananjaya, S. G. (2019). Data De-Duplication Techniques for Cloud Storage: A Review. In Proceedings of the International Conference on Intelligent Computing and Control Systems (ICICCS) (pp. 1877-1881). IEEE.

Saunders, M. N., Lewis, P., & Thornhill, A. (2018). Research methods for business students. Pearson.

Sethi, M., & Sahoo, S. (2018). A Review on Data De-duplication Techniques in Cloud Storage. International Journal of Innovative Technology and Exploring Engineering, 8(10), 1731-1736.

Sharma, P., & Kumar, A. (2019). Review of Data Deduplication Techniques in Cloud Storage. International Journal of Computer Applications, 180(34), 38-42.

Sivathanu, R., Manickam, S., & Anand, M. (2014). An efficient approach for data deduplication in cloud storage using fuzzy logic. Procedia Engineering, 97, 890-897.

Soni, M., Chauhan, A., & Singh, S. (2014). Data De-Duplication: A Review. International Journal of Computer Science and Mobile Computing, 3(3), 387-391.

Sweeney, L. (2002). k-anonymity: A model for protecting privacy. International Journal on Uncertainty, Fuzziness and Knowledge-Based Systems, 10(5), 557-570.

Wang, S., Liu, Z., & Gao, W. (2019). A survey of data deduplication in cloud storage. Journal of Cloud Computing, 8(1), 1-19.

Wang, X., Chen, Y., He, D., Sun, X., & Xu, X. (2019). A secure and efficient deduplication scheme for cloud storage systems. Journal of Information Security and Applications, 47, 86-96.

Wang, X., Feng, D., Lai, X., & Yu, H. (2005). Cryptanalysis of the hash functions MD4 and RIPEMD. Journal of Cryptology, 18(3), 209-223.

Wei, T., Lu, Y., Yang, B., & Huang, Z. (2016). Secure Data Deduplication Scheme Based on Bloom Filter and Hash Function. Journal of Computers, 11(8), 770-779.

Xie, J., Lu, J., Xiong, Y., & Zhou, C. (2019). A Review of Data Deduplication Techniques. IEEE Access, 7, 119760-119773.

Xu, Z., Jiang, C., Chen, L., & Zhu, X. (2014). Security analysis of hash-based deduplication schemes. International Journal of Security and Its Applications, 8(2), 193-202.

Yang, X., Zhang, Y., Wang, Q., & Wang, X. (2019). An efficient data deduplication scheme based on Bloom filter and SHA-256 in distributed storage systems. Future Generation Computer Systems, 92, 571-581.

Zhang, H., Li, M., Li, J., Li, K., & Ren, K. (2016). Homomorphic encryption-based secure deduplication of encrypted data in cloud storage. IEEE Transactions on Information Forensics and Security, 11(12), 2762-2776.

Zhang, L., Lu, Y., Long, F., & Chen, L. (2018). Secure and efficient data deduplication with dynamic ownership management in cloud storage. Future Generation Computer Systems, 78, 238-248.

Zhao, K., Gong, G., Wang, L., & Guo, Z. (2018). A Secure Data Deduplication Scheme Based on BLAKE3. IEEE Access, 6, 39562-39571. doi: 10.1109/access.2018.2854475

Zhu, J., Hu, X., & Chen, Z. (2014). Secure and efficient deduplication with dynamic ownership management in

Zhu, J., Hu, X., & Chen, Z. (2014). Secure and efficient deduplication with dynamic ownership management in cloud storage. IEEE Transactions on Information Forensics and Security, 9(3), 395-406.

Zhu, W., Li, Y., Zhang, J., & Li, Q. (2014). Survey on secure data de-duplication in cloud storage. Journal of Software, 9(3), 781-787.

Zhu, Y., Li, X., Li, H., Wang, Y., & Li, J. (2017). Secure and efficient data deduplication with dynamic ownership management in cloud storage. Journal of Parallel and Distributed Computing, 102, 150-163.

Zhu, Z., Li, J., Hu, Y., & Zhou, J. (2013). A framework for data deduplication using secure hashing based on locality-sensitive hashing. Cluster Computing, 16(2), 357-372. doi: 10.1007/s10586-012-0239-x

Zhu, Z., Xu, W., & Li, Q. (2014). A dynamic data de-duplication technique for cloud storage systems. Future Generation Computer Systems, 30, 23-30.