healthcare

MDPI

Article

Morton Filter-Based Security Mechanism for Healthcare System in Cloud Computing

Sugandh Bhatia [1,*] and Jyoteesh Malhotra [2]

1 Department of Computer Science, Faculty of Engineering and Technology, Guru Nanak Dev University, Amritsar 143005, India
2 Department of Engineering and Technology, GNDU Regional Campus, Jalandhar 144007, India; jyoteesh@gmail.com
* Correspondence: sugandhcs.rsh@gndu.ac.in

Abstract: Electronic health records contain the patient's sensitive information. If these data are acquired by a malicious user, it will not only cause the pilferage of the patient's personal data but also affect the diagnosis and treatment. One of the most challenging tasks in cloud-based healthcare systems is to provide security and privacy to electronic health records. Various probabilistic data structures and watermarking techniques were used in the cloud-based healthcare systems to secure patient's data. Most of the existing studies focus on cuckoo and bloom filters, without considering their throughputs. In this research, a novel cloud security mechanism is introduced, which supersedes the shortcomings of existing approaches. The proposed solution enhances security with methods such as fragile watermark, least significant bit replacement watermarking, class reliability factor, and Morton filters included in the formation of the security mechanism. A Morton filter is an approximate set membership data structure (ASMDS) that proves many improvements to other data structures, such as cuckoo, bloom, semi-sorting cuckoo, and rank and select quotient filters. The Morton filter improves security; it supports insertions, deletions, and lookups operations and improves their respective throughputs by 0.9× to 15.5×, 1.3× to 1.6×, and 1.3× to 2.5×, when compared to cuckoo filters. We used Hadoop version 0.20.3, and the platform was Red Hat Enterprise Linux 6; we executed five experiments, and the average of the results has been taken. The results of the simulation work show that our proposed security mechanism provides an effective solution for secure data storage in cloud-based healthcare systems, with a load factor of 0.9. Furthermore, to aid cloud security in healthcare systems, we presented the motivation, objectives, related works, major research gaps, and materials and methods; we, thus, presented and implemented a cloud security mechanism, in the form of an algorithm and a set of results and conclusions.

Keywords: COVID-19; cloud computing; digital artifacts; Morton filter; E-health records

check for updates

Citation: Bhatia, S.; Malhotra, J. Morton Filter-Based Security Mechanism for Healthcare System in Cloud Computing. *Healthcare* **2021**, *9*, 1551. https://doi.org/10.3390/healthcare9111551

Academic Editors: Mahmoud Elkhodr, Omar Darwish, Belal Alsinglawi and Daniele Giansanti

Received: 17 September 2021
Accepted: 10 November 2021
Published: 15 November 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

The availability of medical records, in the digital form, has played a significant role during the first wave of COVID-19. There was no medicine and vaccine during the first wave of this deadly virus in the year 2020. The only way to combat this zoonotic virus was to maintain social distance. During the second spike of COVID-19, in the month of June 2021, when millions of people had been administering jabs of vaccine every day worldwide, the most prominent method to control this virus was social distancing. Many experts have designed machine learning algorithms to detect the coronavirus in a suspect. After the confirmation that the person is suffering from coronavirus, consultation with the qualified medical practitioner is required. Cloud computing has performed and is still performing a prominent role in the area of E-health services during this pandemic. The area of applications of cloud computing has been widening day by day. It is considered one of the most cost-efficient computing paradigms. At the same time, researchers and scientists criticized the cloud ecosystem [1] and services, due to the lack of digital forensic tools and technologies applied in the cloud computing environment. No technology or paradigm of computing can be declared as a 100% fool-proof secure system. This poses that the privacy and security of patient's information and records must be ensured. Digital medical transcripts are of great importance, as the overall diagnosis is dependent on these reports, and any tampering with the transcripts might be life threatening. To investigate any tampering or modification in the patient's data, digital forensic techniques must be used. Cloud forensic investigation [2] is of the prime concern today, due to the availability of sensitive data, such as medical and financial on the cloud. There is extreme necessity to take immediate action, such as seizing artifacts quickly, acquiring the evidence, and analyzing the dossier in a timely manner. There is a possibility that digital artifacts may be tampered or modified while transferring from cloud infrastructure to the cloud forensic investigation team. In addition, the seized digital artifacts must be reliable and impeccable. It is the need of the hour to design a framework [3] to ensure the reliability of these seized digital artifacts. The present article focuses on the development of a mechanism, which is specifically designed to point out the location of modified or altered artifacts, while moving from the cloud infrastructure to the forensic investigation team. In this article, we represent a space-efficient scheme, which is specifically designed to find out the position of modified digital artifacts. The principal advantage of this mechanism is that it can be comfortably integrated into various available forensic transmission schemes. Furthermore, this designed mechanism is used to locate the position of tampered data in a hierarchical block. To achieve this, a novel algorithm is designed, with the help of the Morton filter [4]. The major property of this system is that it is based upon space-efficient probabilistic data structure, which can be applied to verify whether an element belongs to a specific set or not. The Morton filter is selected, due to its high throughput, when compared with other probabilistic data structures, such as cuckoo, semi-sorting cuckoo, and rank and select quotient filters. Another data structure that is used to store the metadata at the application level is stack. When the modified digital artifact is detected, the proposed algorithm can promptly locate it and reveal its position in the hierarchical block.

The rest of this paper is organized as follows. Section 2 describes the motivation, main objectives of the proposed mechanism, related works, and major research gaps. Section 3 explains the mechanism design, along with explanation of symbols used in the mechanism. Section 4 introduces the working of the system and introduction of designed algorithm. Section 5 explains the functionality of the algorithm. Section 6 highlights the benefits of using the Morton filter. Section 7 presents the performance evaluation and results. Finally, we give concluding remarks in Section 8.

2. Motivation

In cloud-based healthcare systems, the treatment of the patient is dependent upon digital medical transcripts. The reliability and accuracy of these medical transcripts must be ensured during the transfer of E-health records from patient to doctors. Cloud forensic is a prominent digital forensic technique that can be applied to perform examinations in cloud-based systems. The result of forensic analysis is dependent upon the digital artifacts. The reliability of digital artifacts is one of the most important objectives to achieve in the entire forensic process. When the job of collection of artifacts is over, then the phases of identification, collection, preservation, analysis, examination, and presentation start. Even after the completion of analysis, all the artifacts must be preserved [5]. Any alteration or tampering of digital artifacts will break the chain of custody. In each service model of cloud computing, forensic investigators have demonstrated the procedure to perform live forensics. There are various challenges that revoke the live cloud forensics in real applications, and we are discussing some of them. When digital artifacts are seized from the channel, it must be promptly transferred to the forensic analysts. If it is required to retransfer the altered or tampered data [6], then it must be ensured that only the part of altered data or artifact is transferred and not all of the data. In such a way, it is economically feasible for forensic analysts to find out the position of tampered digital artifacts. Another major problem in sending the digital artifacts as streaming data packets is the absence of metadata and other semantic values. It is a tedious and complex task to relocate the corrupted or altered data within a hierarchical block of artifacts during real-time analysis. It is possible to transfer raw data and metadata simultaneously to handle the problem, but it will not be economically feasible for the organization. Moreover, it will slow-down the entire process, and redundancy in the metadata can occur at some level. Therefore, an effort has been made to develop a mechanism to find the location or position of altered, modified, and tampered data in a block of data hierarchy. The fundamental motivation of this mechanism is that it specifies the position of tampered data within a block of data stream [7]. This mechanism takes the benefits of the Morton filter [8] and a stack [9] to place and fetch the metadata on the application level. When the altered data is identified, the mechanism can restrain it immediately and conveys its position in the hierarchical block. Hence, this space-efficient mechanism can be applied to ensure the reliability of digital artifacts in cloud forensics [10] for cloud security [11].

2.1. Objectives of the Mechanism

The principal objectives of the proposed security mechanism are given below.

- Various secure, cloud-based healthcare systems were designed with probabilistic data structures, such as cuckoo [12], bloom, and attribute bloom filters. The most advanced and efficient filter is the Morton filter, and it is the best approach to design a cloud security mechanism with Morton filter as throughput, when compared with other probabilistic data structures.
- Designing a security mechanism that provides secure data storage in cloud-based healthcare systems.
- Employing fragile watermark that has the capability to find any tampering of data and digital artifacts.
- Designing a generic security mechanism that can be applied to various cloud-based healthcare systems.
- Realizing a performance evaluation of the proposed security mechanism, based on metrics such as throughput for lookups, insertions, and deletions load factor, but also by making a security and privacy analysis and assessment of the designed security mechanism, by comparison with other similar approaches.

2.2. Related Works

Various data structures have been used over the years by researchers to accomplish the task of tamper detection. A non-binary Merkle tree can be used in case of a hierarchical dataset whenever there is a limited number of operations. This method has been applied only for the case of hierarchical data and is not applicable for streaming data. Code self-checksumming has been implemented to detect static and dynamic patches, delivered by potentially unwanted programs (PUP) [13]. This tamper detection mechanism was able to checksum instruction with the combination of absolute address and relocation. Proof of past data possession (PDDP) [14] contains the data possession proof to check whether the user originality possessed the evidence or not. As per this mechanism, it is possible for the user to delete the records from the storage, but PDDP modification is not possible. Bloom filter storage was used in this mechanism. Tamper evident logs were used for forensic purposes. This method was based upon regular and continuous auditing. Tamper evident log design was completely dependent on the Merkle tree, and it allows a logger to furnish the proof of its functioning. Cryptographic, one-way hash functions were used to develop the mechanism. Database management system stores the audit logs. Authors explored a novel idea to provide audit logs to detect tampering effectively in transaction processing systems. The cuckoo filter has been used to develop a fast tamper detection mechanism for the hierarchical data in cloud forensic. The authors claimed that the designed mechanism has full compatibility with existing forensic data transmission schemes. The bloom, cuckoo, and counting bloom filters have been implemented to verify the reliability of digital artifacts. In barcode recognition and processing [15], the bloom filter performed a significant role. The single-hash lookup cuckoo filter [16] was used, with one hash function, to find out any specific item. Redundant records can be identified by implementing the accurate counting bloom filter [17]. Remya et al. [18] presented a mechanism for the security of critical electronic health records (EHRs). In this mechanism, security can be achieved with the support of identity-based secure and encrypted data-sharing techniques. Multiple solutions were suggested to secure cloud-based data and records in e-health systems. Security threats and attacks [19] in cloud computing were discussed in detail. Various types of machine learning algorithms were investigated and implemented as a tool to provide security in cloud computing applications. Secure and efficient digital data sharing system for cloud environments [20] have been extensively developed and scholars have presented a cloud-based, student-centered mechanism to solve the problem of user management. The system was based on Lagrange interpolating polynomials. To secure software-defined networks (SDNs) [21] from threats and attacks, a system was proposed that utilizes the benefits of software-defined networks characteristics, along with data mining, to find any malicious activity in the data plane of SDN. Qiang et al. [22] explained the role of digital technologies to control the coronavirus spread. Digital technologies, such as cloud computing, artificial intelligence, internet of things, wearable devices, blockchain technology, and 5G communication networks, have played an irreplaceable role with public health systems in controlling the second wave of COVID-19 in China. Cloud and network security was achieved by applying network and cloud forensics, in association with software-defined networking. Recommendations were given by the authors to apply security and privacy in cloud forensic by using static, dynamic, and remote cloud forensics by Quadri et al. [23]. Shangbin et al. [24] demonstrated a framework that can be used to eliminate the effects of noise and anomalies by implementing robust online evolving anomaly detection (ROEAD) and a robust feature extractor (RFE). A comprehensive analysis, in the area of cloud computing, enabling the internet of things was performed. Privacy protection models, frameworks, and methods were explored. Key challenges, threats, and attacks in cloud computing security were discussed by Tahirkheli et al. [25]. A security and privacy-preserving mechanism was developed to maintain the integrity and confidentiality in electronic health records of patients. It is economically feasible to store, process, and update the electronic data of patients in cloud computing [26]. Graph models [27] were developed and tested to provide security in cloud computing. Cryptographic threshold techniques were applied, along with intelligent secret management and data sharing techniques. Seh et al. [28] discussed the simple moving average and simple exponential smoothing methods, in order to investigate data breaches in healthcare systems. Results of the time series analysis revealed that the cost of data breaches and number of data breaches will increase in near future. Liu et al. [29] proposed a system based on blockchain and distributed ledger-based improved biomedical security system (BDL-IBS), in order to strengthen privacy and security in healthcare applications. Chadwick et al. [30] designed an architecture for sharing the cyber threat information. It is a five trust model, which is based upon five levels. Data sharing agreements (DSA) and data protected objects (DPOs) are important constituents of the architecture, and these constituents perform a significant role in the successful implementation of cloud-edge based data security architecture.

2.3. Major Research Gaps

The related work and research studies draw an intense observation that security and privacy for healthcare systems in cloud computing is enormously challenging. Thus, we are able to find significant gaps and shortcomings of previous studies. A preliminary review of the research literature reveals that, despite the abundance of work in the area of cloud-based healthcare system security and privacy in electronic health records, there is a need for studies that can present the findings and analyze the limitations of previous studies, in terms of probabilistic data structures, such as the cuckoo, bloom, semi-sorting cuckoo, and rank and select quotient filters. To identify the research gaps, articles between 2016 and 2021 were analyzed. Table 1 summarizes work on the cloud-based healthcare system security and privacy of electronic health records, through various probabilistic data structures. As shown in Table 1, the majority of the security mechanisms are designed on bloom and cuckoo filters. None of the mechanisms were designed on the basis of Morton filter, which is considered one of the most efficient probabilistic data structures [8]. Filling this gap, we have designed a Morton filter-based security mechanism for healthcare systems in cloud computing. Another major gap is that no cloud security mechanism has been developed with the integration of least significant bit replacement watermarking and class reliability factors. Our proposed security mechanism, for the healthcare system, is developed with least significant bit replacement watermarking. No cloud security system is capable of performing the reliability test on digital artifacts in cloud forensics. Our proposed mechanism is capable of performing reliability tests on the acquired digital artifacts.

Table 1. Security and privacy mechanisms, designed with probabilistic data structures.

Reference	DataStructure/Technology Used	Cloud-Based Healthcare System	Contribution
Ying et al. (2021) [31]	Cuckoo Filter	No	Suggested a security-enhanced attribute cuckoo filter to hide the access policy and designed ciphertext-policy attribute-based encryption
Xie et al. (2021) [32]	Cuckoo Filter	No	Proposed a lattice signature method, with Cuckoo filter, that can simplify the computational overhead
Kumar et al. (2021) [33]	Bloom Filter	Yes	Explained technique to protect cloud datasets with bloom filter, based ciphertext-policy attribute-based encryption
Cano et al. (2020) [34]	Elliptic Curve Cryptography	Yes	Presented a solution to achieve security and the preservation of data privacy in internet of medical things and the cloud
Breidenbach et al. (2020) [35]	Bloom Filter	No	Discussed privacy-preserving concept, by using bloom filter and cryptographic functions
Shi et al. (2020) [36]	Block Chain	Yes	Investigated various approaches of E-health records in blockchain technology and proposed different applications of healthcare in blockchain
Adamu et al. (2020) [37]	Laravel Security Features	Yes	Proposed a framework that can be used to apply security and privacy to electronic medical record
Breslow et al. (2019) [8]	Morton Filter	No	Designed a mechanism to prove that the Morton filter is an improvement over the cuckoo filter
Jeong et al. (2019) [38]	Bloom Filter	No	Proposed a secure cloud storage service, on the basis of bloom filter and provable data possession model
Patgiri et al. (2019) [39]	Bloom Filter	No	Explored the adaption of bloom filter in network security, packet filtering, and IP address lookup.
Ming et al. (2018) [40]	Cuckoo Filter	Yes	Designed an attribute-based signcryption scheme (ABSC) for privacy-preserving in electronic health record
Decouchant et al. (2018) [41]	Bloom Filter	No	Presented a bloom filter-based novel filtering method that can be applied to reads of any length
Ramu (2018) [42]	Attribute Bloom Filter	Yes	Proposed a secure cloud mechanism to share health records among various users, using ciphertext-policy attribute-based encryption and attribute bloom filter
Brown et al. (2017) [43]	Bloom Filter	Yes	Discussed privacy-preserving record linkage (PPRL) model, along with bloom filter, to overcome the problems of data integration and privacy
Vatsalan et al. (2016) [44]	Counting Bloom Filter	Yes	Proposed a novel method to provide privacy for multi-party privacy-preserving record linkage with counting bloom filter

ABSC: attribute-based signcryption scheme; PPRL: privacy-preserving record linkage.

In this research work, the Morton filter has been used, due to better insert, delete, and lookup throughput. The Morton filter [8] is compared with the cuckoo, semi-sorting cuckoo, and rank and select quotient filters, as all these are space-efficient probabilistic data structures. These data structures can be implemented to check that a particular element is a member of a set or not.

3. Discussion

The study proposes a new Morton filter-based security mechanism for healthcare systems in cloud computing to provide the security and to maintain the integrity of electronic health records. The most important issue, which is resolved in this research article, is how to find out the position of altered data or artifacts from the hierarchical block. A block can be defined as B = {b_1, b_2 . . . b_n}, where b_i ($1 \leq i \leq n$) is an indivisible data constituent available in the block. The data constituent can be further divided into the p data class, such as B = {C_1, C_2 . . . C_p} ($1 \leq r \leq p$). A data class C_r can be demarcated if, and only if, its last constituent b_0^r of class C_r can be considered as the boundary integral element of C_r. While acquiring the digital artifacts, a hierarchical block structure can be generated as a tree. A class of data entities, such as a records, can be formed as an internal node of a tree, whereas a slice of data element can be taken as a terminal node of the tree. Moreover, along with data class C_r, another significant class is described as the demarcated class C_d, in this C_d (value = Hash (fdr_i) ($1 \leq i \leq 0$) C_d.group ∈ { "€", "£" }). Presented as a block of data in the form of a tree that is traversed with iterative deepening bidirectional search [45]. Two inputs are given in C_d and in a hierarchical block, one is "€", which is inserted before an internal node is traversed; the second is "£", and it is inserted when a terminal node has been traversed.

Figure 1 illustrates the demarcated class and the moment of inputting into the tree. Nodes will be traversed in the following order: Fdr_0, Fdr_1, fl_1, fl_2, Fdr_3, fl_6, Fdr_2, fl_3, fl_4, fl_5. The designed tree depicts that the internal nodes are Fdr_0, Fdr_1, Fdr_2, Fdr_3. Hence, a demarcated class of "€" is inserted. Similarly, a demarcated class of "£" is entered after the traversing of their sub-trees.

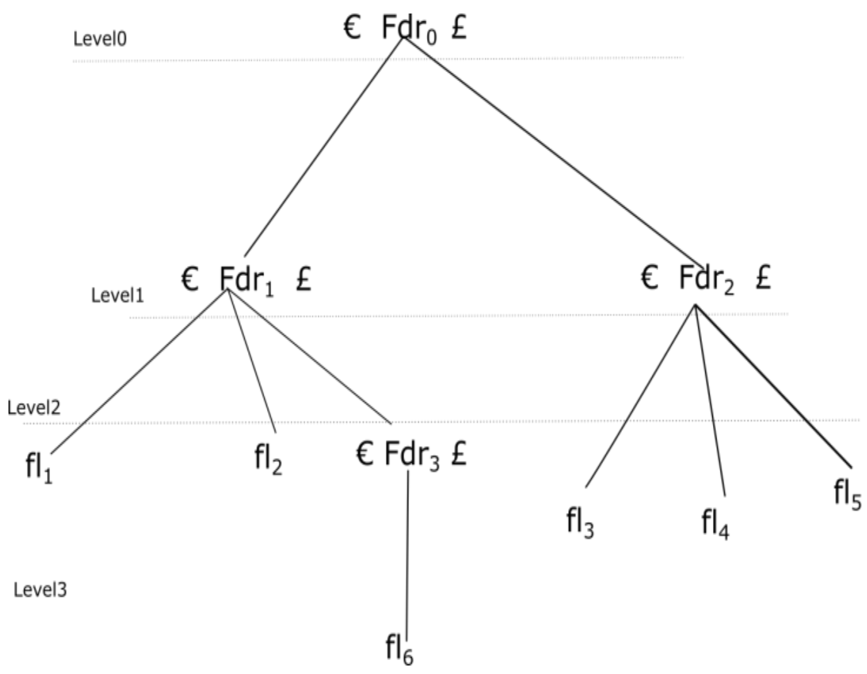

Figure 1. Traversal operations on hierarchical block. Fdr: Directory; fl: File; €: inserted into the data stream before a non-leaf node; £: inserted into the data stream after a non-leaf node has been traversed.

Figure 2 shows the flow of operations in the designed mechanism, which executes the system and consists of five major constituents: LSB replacement watermarking mechanism [46], accommodative extractor [47], watermark validator [48], packet builder, and positional details. The transmitting data is classified into data classes C_i (i = 1, 2 . . . n). The hash values of class C_i are calculated as $HC_{i\,=}$ $H(C_i)$ (i = 1, 2 . . . n), where H is taken as a hash function [49]. Two contiguous class hash values are grouped and computed simultaneously as $H\ (H(C_i)\ ||\ H\ (C_{i+1})$, it is denoted as the class reliability factor (CRF). The dependability in CRF can be achieved with the private key [50] is given by the sender. The CRF will be implanted, such as fragile watermarks [51], within the packets. Fragile watermarks have the capability to detect any modification or tampering of data. After the delivery of the data packet, the accommodative extractor initially extracts the watermark associated with that packet.

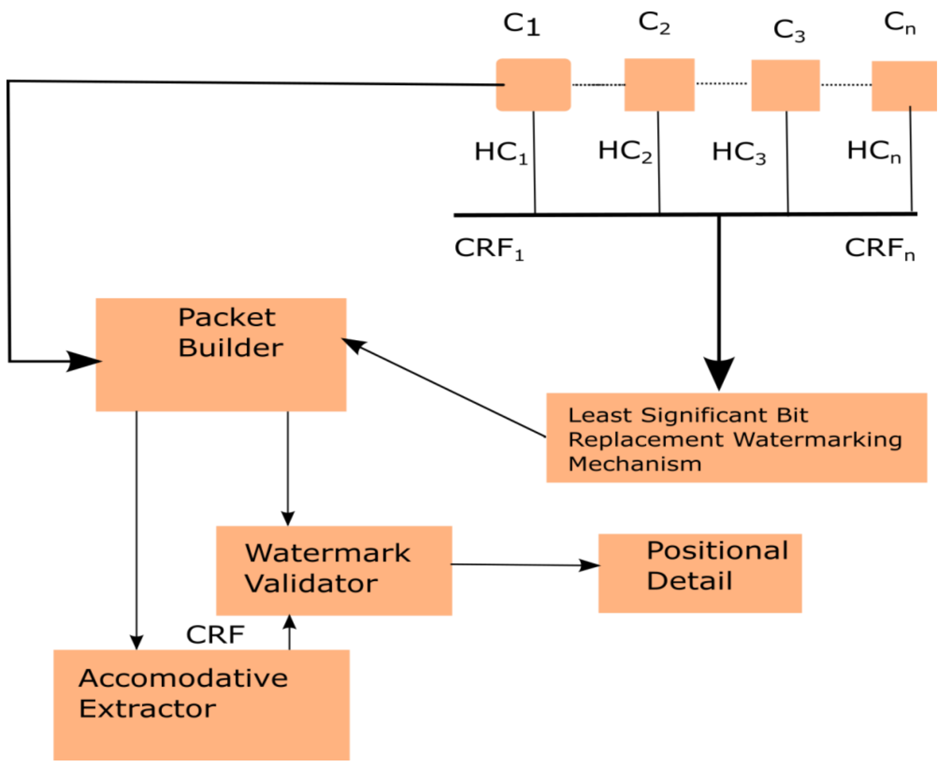

Figure 2. Working of digital artifacts reliability system. HC: Hash value of class; C: Class; CRF: Class reliability factor.

At the same time, raw data is required to calculate CRF, and if the value of calculated CRF matches with the extracted watermarks, then it will be considered that the data class is without any alteration or tampered data; else, the class will be treated as an altered class. For more details, regarding the algorithm for the detection of malicious activity, watermarking as a service (WaaS) [52] can be used.

4. Materials and Methods

After inputting the demarcating class in the data stream, the location of altered or tampered data can be found from the hierarchical block (tree). Various well-discussed data structures are available that can be used for the purpose of designing the data block, such as tree, hash map [53], block chain, and link list. The primary glitch in all these data structures is the requirement of more and more space when private keys are attached with the data structure. Therefore, it is significant to use a space-efficient probabilistic data structure [54] for the storage of internal nodes. The Morton filter, an approximate set membership data structure, is used to overcome the problem. It is an improved data structure, when compared with bloom [55] and cuckoo filters [56], in terms of space-efficient insertion, deletion, and lookup. An algorithm named as identify_positions (Algorithm 1) has been designed, and it explains the algorithm that can detect the position or location of modified or tampered data in the hierarchical block. Four inputs are given: an uninitialized Morton filter (MF), a data stream (DS), an empty stack (TK), and a counter initialized as 0. It provides the result as a set of modified positions (MPS). Once the modified positions are detected and available, it is possible to fetch modified or altered data from MPS. Different subroutines are incorporated in the algorithm discussed below. The subroutine initial is used to initialize the data structure of MF and TK. Another subroutine, chkcls, verifies the number of classes available in streaming data. The subroutine ensures whether MF has a private key. The subroutines inst and del perform operations to insert and delete the key from the MF. Other routines, such as pop, push, peek, and stacktraceback, represent the basic operations of stack data structure.

Algorithm 1. Identify positions.

```
1. Procedure: identify_positions of the modified data
2. Start
3. Input: The data class $C_i$ in the data stream DS, the Morton Filter MF, Stack TK, counter i.
4. DS ← ø ; initial (MF), initial (TK);
5. i ← 0
6. while $C_i$ ← DS.chkcls ( ) do
7. i++;
8. switch $C_i$.type do
9. Case " €"
10. h ← DS ($C_i$)
11. if MF.involve ($DS_i$) = = false
12. then DS.push (h);
13. MF.inst (h);
14. Case "£"
15. If MF.lookup ($DS_i$) = = true
16. then h ← DS.pop ( );
17. MF.del (h);
18. Case "Data"
19. If detect ($C_i$) = = true
20. then Temp_DS = peek (TK)
21. if MF.involve (Temp_D) = = true
22. then
23. $D_j$ = stacktraceback ( );
24. MPS = MPS ∪ {$D_j$}
25. End
Output: The set of modified positions MPS and modified data is detected.
```

5. Functionality of Algorithm

The functionality of the algorithm is discussed here, with the help of an illustration drawn in Figure 3. The streaming data has been distributed into the classes C_1, C_2, … C_n. The demarcating groups are DG_1, DG_2, and DG_3, which are associated with the folders "/Res", "/Res/acc1", and "/Res/acc2". In case of demarcating the group "€", it will be given to the MF. Subsequently, the hash of a similar class will be pushed into the TK. On the other hand, if the demarcating group is "£", it will be wiped out from the MF and from TK, and the hashed will be popped out. The detection algorithm [57] is capable of finding any alteration or tampering within the classes C_1, C_2, … C_n.

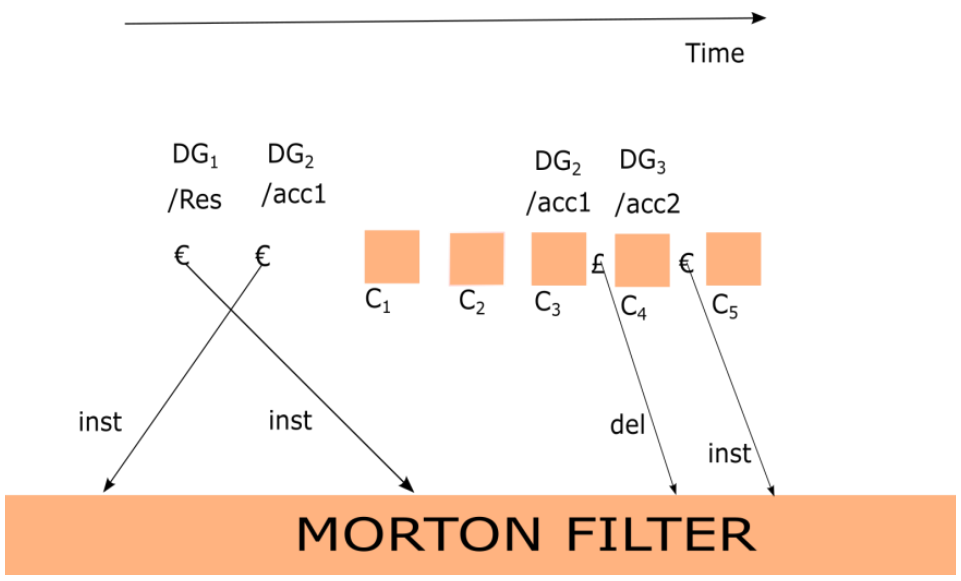

Figure 3. Morton filter and operations. DG: Demarcating group; inst: insert; del: delete; C: Class; DG: Demarcating groups; Res: Folder; acc1: folder; acc2: folder; €: inserted into the data stream before a non-leaf node; £: inserted into the data stream after a non-leaf node has been traversed.

The value at the top of the stack will be fetched and then checked against MF. If MF holds the value, the stack trace will be included into the set of positions (PS) from where the modified data was detected. If class C_2 is detected as a modified class, then DG_2 will be recovered from the top of stack. As it has been injected into MF, the subroutine stacktraceback will be executed and placed at the location "/Res/acc1" into MPS. On the same note, the modified class C_3 will perform the same operations on MPS, and MPS will be constant. When modified class C_4 will be detected, the stack trace will hold the value of DG_1; hence, DG_1 is the only value in the stack, and the location "/Res" is included into the MPS. Finally, the modified class, C_4, will identify that DG_1 and DG_3 are in TK. Therefore, "/Res/acc2" will be inserted into the MPS. As the outcome, the position of the modified class in the hierarchical block and data class will be provided by the designed mechanism.

6. Benefits of Implementing the Morton Filter

The complexity of the procedure identify_positions is linear to the number of classes. For each class, the time complexities of various operations, such as insert, delete, and lookup, are O (1). Moreover, time complexity for basic stack operations, such as push and pop is O (1), as it is possible to work at only one end of this linear data structure. The cuckoo filter [58] is considered a better option than the bloom filter in three ways. It supports the dynamic deletion of items. It has better lookup performance, and, most importantly, it has better space efficiency in applications that require low false positive rates ($\epsilon < 3\%$). However, a Morton filter is an improvement to cuckoo filters. It is called a modified cuckoo filter. It supports various operations, such as insertion, deletion, and lookup, as cuckoo filters. Additionally, it achieves a high throughput and quotient filter, which enhances the capability of the Morton filter, by leveraging its internal representation. It also requires less memory than the cuckoo filter for a similar error rate. Its insert throughput is 15.5$\times$, delete throughput is 1.3$\times$, and lookup throughput is 2.5$\times$ higher, when compared to cuckoo filter. A comparison is made in the Figure 4 between the Morton, cuckoo, semi-sorting cuckoo, and rank and select quotient filters. The comparison reveals that an MF is approximately 0.9$\times$ to 15.5$\times$ faster than a CF for insertions.

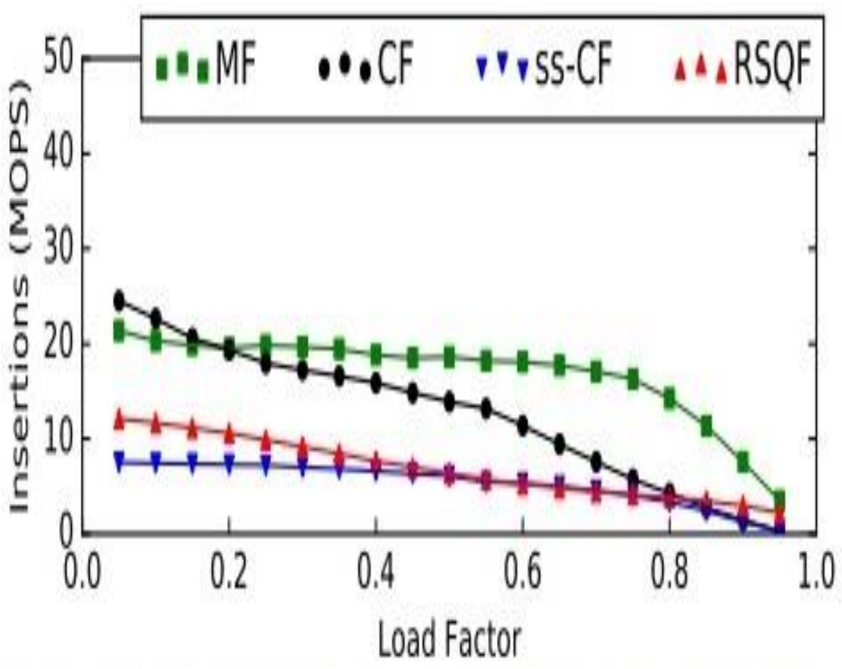

Figure 4. Comparison between Morton and cuckoo filters for insert operation. MOPS: millions of operations per second; MF: Morton filter; CF: Cuckoo filter; ss-CF: semi-sorting Cuckoo filter; RSQF: rank and select Quotient filter.

Similarly, other operations are depicted in the Figures 5 and 6, respectively, regarding the delete and lookup operations in MF, CF, ss-CF, and RSQF. In delete operation, also, the MF is faster, by 1.3× to 1.6×, than the CF.

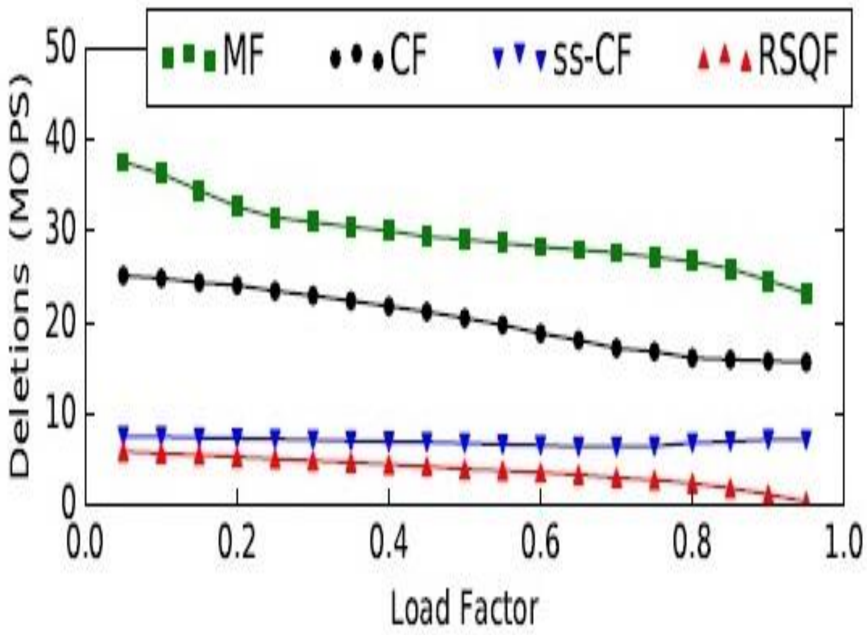

Figure 5. Comparison between Morton and cuckoo filters for delete operation. MOPS: millions of operations per second; MF: Morton filter; CF: Cuckoo filter; ss-CF: semi-sorting Cuckoo filter; RSQF: rank and select Quotient filter.

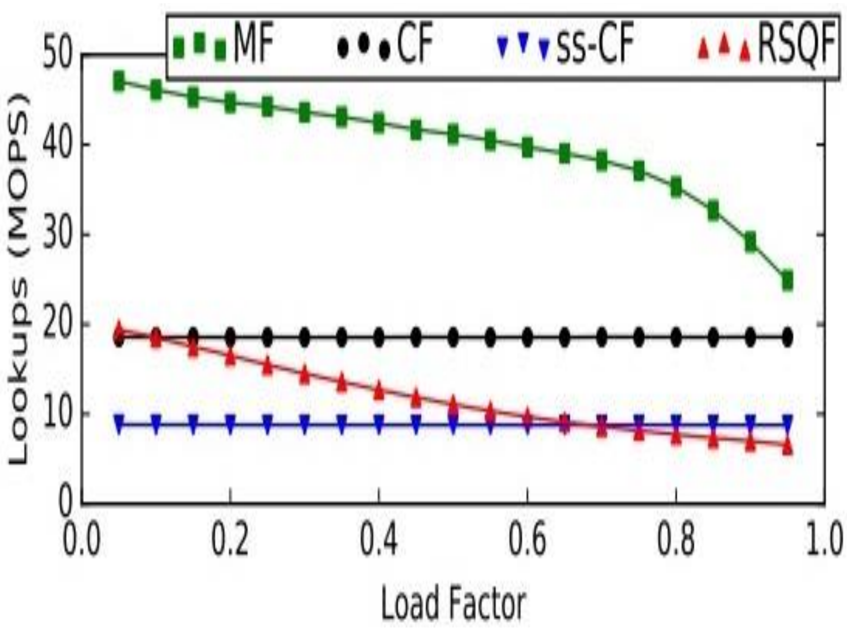

Figure 6. Comparison between Morton and cuckoo filters for lookup operation. MOPS: millions of operations per second; MF: Morton filter; CF: Cuckoo filter; ss-CF: semi-sorting Cuckoo filter; RSQF: rank and select Quotient filter.

7. Results

Experiments were conducted on three servers with a 2.5 GHz Intel i5 10th generation CPU and 8 GB RAM (Lenovo, Beijing, China). We used Hadoop version 0.20.3 (Apache Software Foundation, Forest Hill, MD, US), and the platform was Red Hat Enterprise Linux 6 (Red Hat, Inc., Raleigh, NC, US). We executed five experiments, and the average of the results was taken. The proposed mechanism exhibits space efficiency. Total numbers of 100,000 bits were taken and accommodated 10,000 different constituents of metadata with a false positive rate of 7.496×10^{-12}. However, the mechanism can organize 805 different metadata pieces, in the case of the implementation of the SHA-256 hash function. In other situations, it is possible to organize 1610 metadata constituents, if the MD5 hash function is implemented. The experiments initially analyzed a false positive rate [59] of the Morton filter, with the increasing quantity of metadata. A comparison is performed between the false positive rate of the Morton and cuckoo filters. The details of both the Morton and cuckoo filters are given below in Tables 2 and 3.

Table 2. Parameters used by cuckoo filter.

Token and Its Explanation	α (Load Factor)	m (No. of Buckets)	b (No. of Entries per Bucket)	f (Length of Fingerprints in Bits)
Value	0.8	8	8	36

Table 3. Parameters used by Morton filter.

Token and Its Explanation	α (Load Factor)	m (No. of Buckets)	b (No. of Entries per Bucket)	f (Length of Fingerprints in Bits)
Value	0.9	8	8	36

The Morton filter is an improvement over the cuckoo filter in three ways.

1. Memory utilization is one of the important advantages of using the Morton filter.
2. It is placed in a better position, when compared to the cuckoo filter, in the context of insert, delete, and lookup [60] throughput.

3. Its implementation is not a complex and tedious task, as in the case of earlier filters, such as the bloom, quotient, and cuckoo filters.

Therefore, the proposed that a digital artifacts reliability system can be implemented as a security mechanism, in order to ensure the reliability of electronic health records and medical transcripts in cloud-based healthcare systems, and it can be used to strengthen the cloud forensics [61] process. An effort has been made by the authors to overcome all the limitations and drawbacks of bloom, quotient, and cuckoo filters, by using the Morton filter.

8. Conclusions

In this research paper, we discuss the design and implementation of a Morton filter-based security mechanism for cloud-based healthcare systems, which can provide security and privacy to electronic health records or digital medical transcripts. In contrast to the existing schemes, the developed mechanism uses the Morton filter for storage, and it can be applied in the area of cloud forensics. For the successful implementation and execution of the discussed mechanism, a couple of algorithms and forensic data transmission schemes are integrated with this system. Tampered digital artifacts and their location can be pinpointed in real-time. In addition, there are many options for probabilistic data structures, such as cuckoo, bloom, semi-sorting cuckoo, and rank and select quotient filters. Compared to the cuckoo filter, this Morton filter-based security mechanism is lighter and has less computational overhead, with time complexity $O\ (1)$ and load factor 0.9, in finding the location of altered or tampered data in hierarchical data block in cloud-based healthcare systems. Hence, it permits the digital forensic team to localize the tampered or altered data and its location promptly. For further studies, we intend to apply advanced hashing techniques, along with other probabilistic data structures.

Author Contributions: Conceptualization S.B. and J.M.; methodology and formal analysis S.B. and J.M.; writing—original draft preparation S.B.; writing—review and editing S.B. and J.M.; supervision and project administration J.M.; funding acquisition S.B. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Ko, R.; Choo, K.-K.R. *The Cloud Security Ecosystem: Technical, Legal, Business and Management Issues*; Syngress Is an Imprint of Elsevier: Waltham, MA, USA, 2015.
2. Park, S.; Kim, Y.; Park, G.; Na, O.; Chang, H. Research on Digital Forensic Readiness Design in a Cloud Computing-Based Smart Work Environment. *Sustainability* **2018**, *10*, 1203. [CrossRef]
3. Kim, P.; Jo, E.; Lee, Y. An Efficient Search Algorithm for Large Encrypted Data by Homomorphic Encryption. *Electronics* **2021**, *10*, 484. [CrossRef]
4. Fan, B.; Andersen, D.G.; Kaminsky, M.; Mitzenmacher, M.D. Cuckoo Filter. In Proceedings of the 10th ACM International on Conference on emerging Networking Experiments and Technologies, Sydney, Australia, 2–5 December 2014. [CrossRef]
5. Kävrestad, J. *Fundamentals of Digital Forensics*; Springer: Berlin/Heidelberg, Germany, 2018.
6. Ray, B.R.; Chowdhury, M.; Abawajy, J. Hybrid Approach to Ensure Data Confidentiality and Tampered Data Recovery for RFID Tag. *Int. J. Networked Distrib. Comput.* **2013**, *1*, 79–88. [CrossRef]
7. Chun, Y.; Han, K.; Hong, Y. High-Performance Multi-Stream Management for SSDs. *Electronics* **2021**, *10*, 486. [CrossRef]
8. Breslow, A.D.; Jayasena, N.S. Morton filters: Fast, compressed sparse cuckoo filters. *VLDB J.* **2020**, *29*, 731–754. [CrossRef]
9. Yang, H.; Kim, Y. Design and Implementation of High-Availability Architecture for IoT-Cloud Services. *Sensors* **2019**, *19*, 3276. [CrossRef]
10. Bhatia, S.; Malhotra, J. CFRF: Cloud Forensic Readiness Framework—A Dependable Framework for Forensic Readiness in Cloud Computing Environment. *Lect. Notes Data Eng. Commun. Technol.* **2020**, 765–775. [CrossRef]

11. Islam, S.; Ouedraogo, M.; Kalloniatis, C.; Mouratidis, H.; Gritzalis, S. Assurance of Security and Privacy Requirements for Cloud Deployment Models. *IEEE Trans. Cloud Comput.* **2015**, *6*, 387–400. [CrossRef]
12. Cui, J.; Zhang, J.; Zhong, H.; Xu, Y. SPACF: A Secure Privacy-Preserving Authentication Scheme for VANET with Cuckoo Filter. *IEEE Trans. Veh. Technol.* **2017**, *66*, 10283–10295. [CrossRef]
13. Urban, T.; Tatang, D.; Holz, T.; Pohlmann, N. Towards Understanding Privacy Implications of Adware and Potentially Unwanted Programs. In *Computer Security*; Springer International Publishing: Cham, Switzerland, 2018; pp. 449–469.
14. Zawoad, S.; Hasan, R. I Have the Proof: Providing Proofs of Past Data Possession in Cloud Forensics. In Proceedings of the 2012 International Conference on Cyber Security, Alexandria, VA, USA, 14–16 December 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 75–82.
15. Jiang, M.; Zhao, C.; Mo, Z.; Wen, J. An improved algorithm based on Bloom filter and its application in bar code recognition and processing. *EURASIP J. Image Video Process.* **2018**, *2018*, 139. [CrossRef]
16. Najafimehr, M.; Ahmadi, M. SLCF: Single-hash lookup cuckoo filter. *J. High Speed Netw.* **2019**, *25*, 413–424. [CrossRef]
17. Huang, K.; Zhang, J.; Zhang, D.; Xie, G.; Salamatian, K.; Liu, A.X.; Li, W. A Multi-partitioning Approach to Building Fast and Accurate Counting Bloom Filters. In Proceedings of the 2013 IEEE 27th International Symposium on Parallel and Distributed Processing, Cambridge, MA, USA, 20–24 May 2013; IEEE: Piscataway, NJ, USA, 2013; pp. 1159–1170.
18. Sivan, R.; Zukarnain, Z. Security and Privacy in Cloud-Based E-Health System. *Symmetry* **2021**, *13*, 742. [CrossRef]
19. Butt, U.A.; Mehmood, M.; Shah, S.B.H.; Amin, R.; Shaukat, M.W.; Raza, S.M.; Suh, D.Y.; Piran, J. A Review of Machine Learning Algorithms for Cloud Computing Security. *Electronics* **2020**, *9*, 1379. [CrossRef]
20. Wu, Z.-Y. A Secure and Efficient Digital-Data-Sharing System for Cloud Environments. *Sensors* **2019**, *19*, 2817. [CrossRef]
21. Amanowicz, M.; Jankowski, D. Detection and Classification of Malicious Flows in Software-Defined Networks Using Data Mining Techniques. *Sensors* **2021**, *21*, 2972. [CrossRef]
22. Wang, Q.; Su, M.; Zhang, M.; Li, R. Integrating Digital Technologies and Public Health to Fight COVID-19 Pandemic: Key Technologies, Applications, Challenges and Outlook of Digital Healthcare. *Int. J. Environ. Res. Public Heal.* **2021**, *18*, 6053. [CrossRef]
23. Waseem, Q.; Alshamrani, S.; Nisar, K.; Din, W.W.; Alghamdi, A. Future Technology: Software-Defined Network (SDN) Forensic. *Symmetry* **2021**, *13*, 767. [CrossRef]
24. Han, S.; Wu, Q.; Zhang, H.; Qin, B.; Hu, J.; Shi, X.; Liu, L.; Yin, X. Log-Based Anomaly Detection with Robust Feature Extraction and Online Learning. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 2300–2311. [CrossRef]
25. Tahirkheli, A.; Shiraz, M.; Hayat, B.; Idrees, M.; Sajid, A.; Ullah, R.; Ayub, N.; Kim, K.-I. A Survey on Modern Cloud Computing Security over Smart City Networks: Threats, Vulnerabilities, Consequences, Countermeasures, and Challenges. *Electronics* **2021**, *10*, 1811. [CrossRef]
26. Chenthara, S.; Ahmed, K.; Wang, H.; Whittaker, F. Security and Privacy-Preserving Challenges of e-Health Solutions in Cloud Computing. *IEEE Access* **2019**, *7*, 74361–74382. [CrossRef]
27. Ogiela, L.; Ogiela, M.R.; Ko, H. Intelligent Data Management and Security in Cloud Computing. *Sensors* **2020**, *20*, 3458. [CrossRef] [PubMed]
28. Seh, A.H.; Zarour, M.; Alenezi, M.; Sarkar, A.K.; Agrawal, A.; Kumar, R.; Ahmad Khan, R. Healthcare Data Breaches: Insights and Implications. *Healthcare* **2020**, *8*, 133. [CrossRef] [PubMed]
29. Liu, H.; Crespo, R.G.; Martínez, O. Enhancing Privacy and Data Security across Healthcare Applications Using Blockchain and Distributed Ledger Concepts. *Healthcare* **2020**, *8*, 243. [CrossRef] [PubMed]
30. Chadwick, D.W.; Fan, W.; Costantino, G.; de Lemos, R.; Di Cerbo, F.; Herwono, I.; Manea, M.; Mori, P.; Sajjad, A.; Wang, X.-S. A cloud-edge based data security architecture for sharing and analysing cyber threat information. *Futur. Gener. Comput. Syst.* **2020**, *102*, 710–722. [CrossRef]
31. Ying, Z.; Jiang, W.; Liu, X.; Xu, S.; Deng, R. Reliable Policy Updating under Efficient Policy Hidden Fine-grained Access Control Framework for Cloud Data Sharing. *IEEE Trans. Serv. Comput.* **2021**, 1. [CrossRef]
32. Xie, G.; Liu, Y.; Xin, G.; Yang, Q. Blockchain-Based Cloud Data Integrity Verification Scheme with High Efficiency. *Secur. Commun. Netw.* **2021**, *2021*, 1–15. [CrossRef]
33. Kumar, G.S.; Krishna, A.S. Data Security for Cloud Datasets with Bloom Filters on Ciphertext Policy Attribute Based Encryption. *Res. Anthol. Artif. Intell. Appl. Secur.* **2021**, 1431–1447. [CrossRef]
34. Cano, M.-D.; Cañavate-Sanchez, A. Preserving Data Privacy in the Internet of Medical Things Using Dual Signature ECDSA. *Secur. Commun. Netw.* **2020**, *2020*, 1–9. [CrossRef]
35. Breidenbach, U.; Steinebach, M.; Liu, H. Privacy-Enhanced Robust Image Hashing with Bloom Filters. In Proceedings of the 15th International Conference on Availability, Reliability and Security, Dublin, Ireland, 25–28 August 2020.
36. Shi, S.; He, D.; Li, L.; Kumar, N.; Khan, M.K.; Choo, K.-K.R. Applications of blockchain in ensuring the security and privacy of electronic health record systems: A survey. *Comput. Secur.* **2020**, *97*, 101966. [CrossRef]
37. Adamu, J.; Hamzah, R.; Rosli, M.M. Security issues and framework of electronic medical record: A review. *Bull. Electr. Eng. Inform.* **2020**, *9*, 565–572. [CrossRef]
38. Jeong, J.; Joo, J.W.J.; Lee, Y.; Son, Y. Secure Cloud Storage Service Using Bloom Filters for the Internet of Things. *IEEE Access* **2019**, *7*, 60897–60907. [CrossRef]

39. Patgiri, R.; Nayak, S.; Borgohain, S.K. Hunting the Pertinency of Bloom Filter in Computer Networking and Beyond: A Survey. *J. Comput. Netw. Commun.* **2019**, *2019*, 1–10. [CrossRef] [PubMed]
40. Ming, Y.; Zhang, T. Efficient Privacy-Preserving Access Control Scheme in Electronic Health Records System. *Sensors* **2018**, *18*, 3520. [CrossRef] [PubMed]
41. Decouchant, J.; Fernandes, M.; Völp, M.; Couto, F.M.; Esteves-Veríssimo, P. Accurate filtering of privacy-sensitive information in raw genomic data. *J. Biomed. Inform.* **2018**, *82*, 1–12. [CrossRef] [PubMed]
42. Ramu, G. A secure cloud framework to share EHRs using modified CP-ABE and the attribute bloom filter. *Educ. Inf. Technol.* **2018**, *23*, 2213–2233. [CrossRef]
43. Brown, A.P.; Ferrante, A.M.; Randall, S.M.; Boyd, J.; Semmens, J.B. Ensuring Privacy When Integrating Patient-Based Datasets: New Methods and Developments in Record Linkage. *Front. Public Health* **2017**, *5*, 34. [CrossRef]
44. Vatsalan, D.; Christen, P.; Rahm, E. Scalable Privacy-Preserving Linking of Multiple Databases Using Counting Bloom Filters. In Proceedings of the 2016 IEEE 16th International Conference on Data Mining Workshops (ICDMW), Barcelona, Spain, 12–15 December 2016; IEEE: Piscataway, NJ, USA, 2016; pp. 882–889.
45. Sarkar, U.; Chakrabarti, P.; Ghose, S.; De Sarkar, S. Effective use of memory in iterative deepening search. *Inf. Process. Lett.* **1992**, *42*, 47–52. [CrossRef]
46. Roy, S.S.; Basu, A.; Das, M.; Chattopadhyay, A. FPGA implementation of an adaptive LSB replacement based digital watermarking scheme. In Proceedings of the 2018 International Symposium on Devices, Circuits and Systems (ISDCS), Howrah, India, 29–31 March 2018.
47. Kricha, Z.; Kricha, A.; Sakly, A. Accommodative extractor for QIM-based watermarking schemes. *IET Image Process.* **2019**, *13*, 89–97. [CrossRef]
48. Li, M.; Yuan, X.; Chen, H.; Li, J. Quaternion Discrete Fourier Transform-Based Color Image Watermarking Method Using Quaternion QR Decomposition. *IEEE Access* **2020**, *8*, 72308–72315. [CrossRef]
49. Wang, D.; Jiang, Y.; Song, H.; He, F.; Gu, M.; Sun, J. Verification of Implementations of Cryptographic Hash Functions. *IEEE Access* **2017**, *5*, 7816–7825. [CrossRef]
50. Na, D.; Park, S. Fusion Chain: A Decentralized Lightweight Blockchain for IoT Security and Privacy. *Electronics* **2021**, *10*, 391. [CrossRef]
51. Su, G.-D.; Chang, C.-C.; Lin, C.-C. Effective Self-Recovery and Tampering Localization Fragile Watermarking for Medical Images. *IEEE Access* **2020**, *8*, 160840–160857. [CrossRef]
52. Naz, F.; Khan, A.; Ahmed, M.; Khan, M.I.; Din, S.; Ahmad, A.; Jeon, G. Watermarking as a service (WaaS) with anonymity. *Multimedia Tools Appl.* **2019**, *79*, 16051–16075. [CrossRef]
53. Harfoushi, O.; Obiedat, R. Security in Cloud Computing Using Hash Algorithm: A Neural Cloud Data Security Model. *Mod. Appl. Sci.* **2018**, *12*, 143. [CrossRef]
54. Nandhini, K.; Balasubramaniam, R. Malicious Website Detection Using Probabilistic Data Structure Bloom Filter. In Proceedings of the 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 27–29 March 2019; IEEE: Piscataway, NJ, USA, 2019; pp. 311–316.
55. Reviriego, P.; Pontarelli, S.; Maestro, J.A.; Ottavi, M. A Synergetic Use of Bloom Filters for Error Detection and Correction. *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.* **2014**, *23*, 584–587. [CrossRef]
56. Ho, T.; Cho, S.; Oh, S. Parallel multiple pattern matching schemes based on cuckoo filter for deep packet inspection on graphics processing units. *IET Inf. Secur.* **2018**, *12*, 381–388. [CrossRef]
57. Li, G.; Wang, J.; Liang, J.; Yue, C. Application of Sliding Nest Window Control Chart in Data Stream Anomaly Detection. *Symmetry* **2018**, *10*, 113. [CrossRef]
58. Zhang, T.; Zhang, T.; Ji, X.; Xu, W. Cuckoo-RPL: Cuckoo Filter Based RPL for Defending AMI Network from Blackhole Attacks. In Proceedings of the 2019 Chinese Control Conference (CCC), Guangzhou, China, 27–30 July 2019.
59. Bhattacharjee, M.; Dhar, S.K.; Subramanian, S. *Recent Advances in Biostatistics False Discovery Rates, Survival Analysis, and Related Topics*; World Scientific Pub. Co.: Singapore, 2011.
60. Li, D.; Chen, P. Optimized Hash Lookup for Bloom Filter Based Packet Routing. In Proceedings of the 2013 16th International Conference on Network-Based Information Systems, Gwangju, Korea, 4–6 September 2013.
61. Roussev, V.; McCulley, S. Forensic analysis of cloud-native artifacts. *Digit. Investig.* **2016**, *16*, S104–S113. [CrossRef]