Mathematical and Computational Applications

MDPI

Article

A Cloud Integrity Verification and Validation Model Using Double Token Key Distribution Model

V. N. V. L. S. Swathi [1,*], G. Senthil Kumar [2] and A. Vani Vathsala [3]

[1] Department of Computer Science and Engineering, SRM Institute of Science and Technology, Chennai 603203, India
[2] Department of Computational Intelligence, SRM Institute of Science and Technology, Chennai 603203, India; senthilg1@srmist.edu.in
[3] Department of Computer Science and Engineering, CVR College of Engineering, Hyderabad 501510, India; vani_vathsala@cvr.ac.in
* Correspondence: vs9189@srmist.edu.in

Abstract

Numerous industries have begun using cloud computing. Among other things, this presents a plethora of novel security and dependability concerns. Thoroughly verifying cloud solutions to guarantee their correctness is beneficial, just like with any other computer system that is security- and correctness-sensitive. While there has been much research on distributed system validation and verification, nobody has looked at whether verification methods used for distributed systems can be directly applied to cloud computing. To prove that cloud computing necessitates a unique verification model/architecture, this research compares and contrasts the verification needs of distributed and cloud computing. Distinct commercial, architectural, programming, and security models necessitate distinct approaches to verification in cloud and distributed systems. The importance of cloud-based Service Level Agreements (SLAs) in testing is growing. In order to ensure service integrity, users must upload their selected services and registered services to the cloud. Not only does the user fail to update the data when they should, but external issues, such as the cloud service provider's data becoming corrupted, lost, or destroyed, also contribute to the data not becoming updated quickly enough. The data saved by the user on the cloud server must be complete and undamaged for integrity checking to be effective. Damaged data can be recovered if incomplete data is discovered after verification. A shared resource pool with network access and elastic extension is realized by optimizing resource allocation, which provides computer resources to consumers as services. The development and implementation of the cloud platform would be greatly facilitated by a verification mechanism that checks the data integrity in the cloud. This mechanism should be independent of storage services and compatible with the current basic service architecture. The user can easily see any discrepancies in the necessary data. While cloud storage does make data outsourcing easier, the security and integrity of the outsourced data are often at risk when using an untrusted cloud server. Consequently, there is a critical need to develop security measures that enable users to verify data integrity while maintaining reasonable computational and transmission overheads. A cryptography-based public data integrity verification technique is proposed in this research. In addition to protecting users' data from harmful attacks like replay, replacement, and forgery, this approach enables third-party authorities to stand in for users while checking the integrity of outsourced data. This research proposes a Cloud Integrity Verification and Validation Model using the Double Token Key Distribution (CIVV-DTKD) model for enhancing cloud quality of service levels. The proposed model, when compared with the traditional methods, performs better in verification and validation accuracy levels.

Academic Editor: Oliver Schuetze

Received: 8 August 2025
Revised: 23 September 2025
Accepted: 29 September 2025
Published: 13 October 2025

Citation: Swathi, V.N.V.L.S.; Kumar, G.S.; Vathsala, A.V. A Cloud Integrity Verification and Validation Model Using Double Token Key Distribution Model. *Math. Comput. Appl.* **2025**, *30*, 114. https://doi.org/10.3390/mca30050114

Keywords: cloud computing; Service Level Agreements; verification; validation; data integrity; attacks; data modifications; quality of service

1. Introduction

Cost savings, sharing and configuring computer resources, upon request service, and increased flexibility and scalability are only a few advantages of cloud computing [1]. Furthermore, cloud computing offers online services that are now commonplace in technology. To supply large enterprise-level applications with availability and scalability, the cloud computing paradigm developed through virtualization, collaborative computing for utility purposes, and other computer technologies [2]. Additionally, it supports virtual machines allocated via a substantial physical resource pool. Cloud computing's five essential qualities are resource pooling, high-speed resilience, measurable services, on-demand self-service, and broad network access [3]. These advantages encourage large businesses to move their IT infrastructure to the cloud. Cloud computing requires user data security to provide dependable services [4]. Common cloud computing hazards exist, including data loss, abuse, malicious insiders, unsecured interfaces, points of entry, shared technology issues, and hijacking. Thus, a fundamental prerequisite for the effective implementation of cloud computing is a thorough grasp of cloud security [5].

It is highly significant as it focuses on one of the key challenges in cloud computing maintaining the integrity and security of stored data. As cloud adoption continues to grow in areas such as healthcare, finance, government, and education, safeguarding data against tampering, unauthorized access, and insider misuse has become a critical need. While existing methods like PDP and PoR provide some level of protection, they often fall short in terms of scalability, efficiency, or robustness. By introducing a dual-token verification approach supported by an independent authority, the proposed CIVV–DTKD model directly addresses these limitations, making the research both timely and valuable for advancing cloud security.

It is challenging for cloud service providers and managers to implement potential solutions that consumers may require due to various threats [6]. This is because different attacks are linked to different hazards, and the significance of risks varies based on the security requirements of other users of cloud services [7]. In order to fulfill their fundamental security obligations as Cloud Service Providers (CSPs), security administrators will assess and put into place security measures [8]. Creating a safe system is nearly impossible, but security can be increased. Finding security risks and the related countermeasures, such as transparency, authentication, and privacy preservation, is therefore essential [9]. The primary drawback of using the cloud is that the user no longer controls their data once it is stored. CSPs control the data stored in their data centers [10]. The CSP can change, remove, or copy data without the user's knowledge. Unsupervised storage of sensitive information is the leading cause of data integrity issues. Cloud computing has serious privacy, security, and integrity risks, even though it is less expensive and requires less resource management. The resource allocated to one user may eventually be assigned to another because of the multi-tenancy design [11]. The general data integrity verification process is shown in Figure 1.

A malicious person can recover private user data using malicious code to exploit a vulnerability in the resource pooling mechanism [12]. Data stored in multi-tenant clouds may be dangerous due to improper disk sanitization. Unintentional or deliberate data backup disasters may cause the data to become unavailable. Security measures should be implemented to avoid data manipulation and unauthorized entry to the cloud environment [13].

A fundamental human right, privacy requires the proper use and preservation of personal data and the right to be left alone. Various practices associated with adopting computing paradigms compromise privacy, including the theft of private data, the unrestrained use of cloud services, data dissemination, the possibility of unauthorized secondary use, the transnational transfer of data, and dynamic provisioning [14]. Regulation of data retention, data deletion by outsourcing, and breaches of privacy awareness are additional privacy concerns. Currently, a consensus is reached in most cases by referring to the standard processing requirements for private information or a third-party service [15]. When providing permission to users in a setting with little to no user interface, security and privacy concerns become more complex because of unauthorized use of data permissions and inefficient processing of private data, which is frequently overlooked during the design stage.

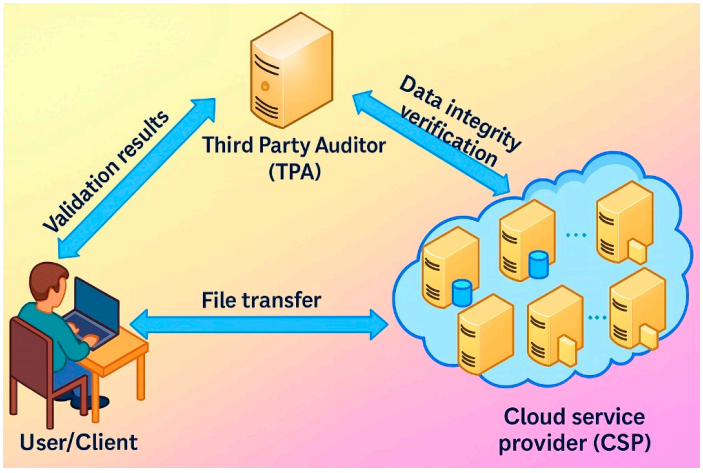

Figure 1. General data integrity verification process.

Concerns exist regarding cloud security deployment and user data security policies in a public cloud environment. Organizations have recently started providing third parties access to users' sensitive data for security audits, increasing security issues over third parties' accountability. The best-case situation, which is still unsuitable for real-life deployment, is an honest but inquisitive third party [16]. An inside assault may transpire through deploying malicious programs on edge nodes, taking advantage of vulnerabilities that compromise the Quality of Service (QoS), without proper identity management. Such hostile operations have the potential to seriously impact sensitive data that is momentarily stored on several edge routers [17]. As more businesses turn to the cloud for efficient data storage, companies must exchange, process, and distribute sensitive data quickly to improve decision-making. However, the lack of privacy and security flexibility is a significant drawback [18]. The inability of the security and privacy mechanisms to adapt to the ever-changing external environment has resulted in an unmanageable danger of data leakage [19]. Businesses are worried about maintaining cloud security while minimizing data leaks and user information. Regretfully, data storage services are constantly evolving, and privacy these days is a personal choice; what one person considers private may be shown to others without permission. Therefore, developing cloud computing privacy and security protocols requires the specification of non-specific requirements [12].

Since technology and its assets are moving into an environment where everyone can choose what they wish to keep private, especially in cloud environments, strict privacy or security restrictions will eventually lead to stagnation. This is because when customers utilize the cloud storage feature provided by cloud service providers, they do not create a local copy of their data; the data on the server in the cloud is highly crucial. Clients want and expect that their data is accessible at all times and that it is not lost or destroyed on

the server-side. This pertains to the issue of verifying data integrity. The most common technique for ensuring data integrity is downloading all the data to a local hard drive and verifying it there. However, if the initial approach is used, it will result in significant overhead in all areas due to the enormous number of users and volume of information stored in the cloud. The user forfeits control over the data since it is hosted on a cloud server furnished by an outside cloud service provider. Protection of privacy and data security are entirely dependent on internet service providers. The user validation and verification process is shown in Figure 2.

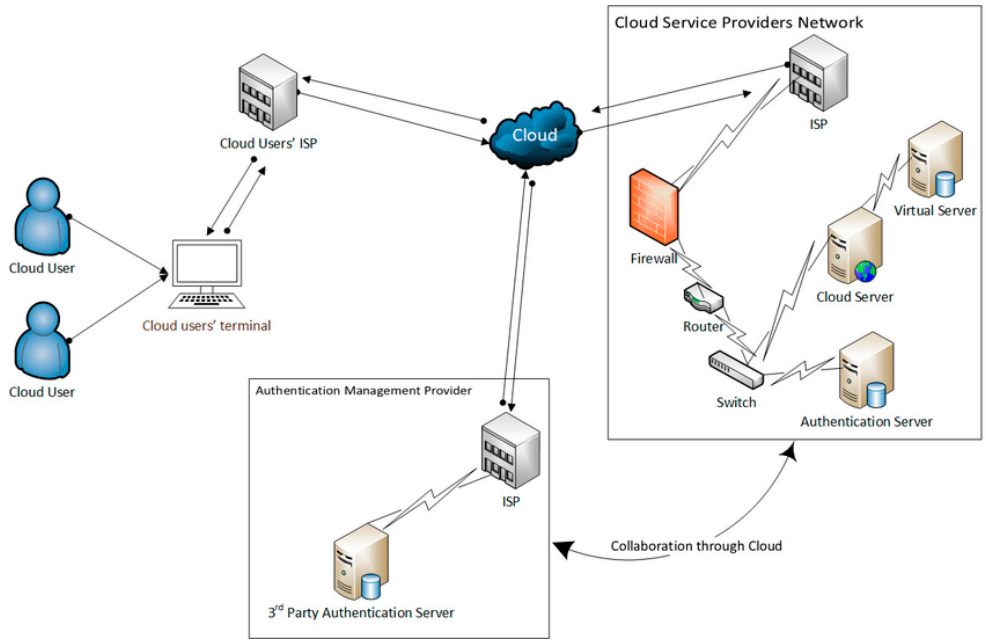

Figure 2. User Validation and Verification Process.

Integrity verification guarantees that user data on the cloud server is complete and has not been altered or corrupted. Data recovery can be performed on damaged data to complete the data that was discovered to be incomplete during the verification phase. Users receive computational resources in the form of services through resource allocation optimization, which also results in the realization of a shared resource pool and connection to the network and elastic expansion. The development and implementation of the cloud platform will be significantly aided by a verification method that checks the integrity of information in the context of the cloud, irrespective of storage providers, and applicable to the current basic service architecture. In addition to guaranteeing high data verification dependability, the solution should not put an undue strain on users or cloud servers. In order to minimize computational overhead once the data are updated, user privacy should be preserved throughout the verification procedure without compromising other data blocks.

In order to guarantee that the data remains complete and correct in accordance with the desires of the data owner, the integrity of data means that the data was not modified or deleted without authority. In addition, the standards lower the likelihood of risk occurrence, enable decision-making for users, and facilitate accounting informatization to aid corporate managers in management. The data program flow design is examined, the cloud-based data integrity identification model is built, the cloud information integrity verification protocol is studied using the data reliability verification algorithms, and the laborious examination of file data insertion operations is based on learning models. This research proposes a Cloud Integrity Verification and Validation Model using the Double Token Key Distribution model for enhancing cloud quality of service levels.

The main idea of the proposed model is the use of an Integrity Verification Authority (CIVA) that controls the verification process through a two-token system. In this setup, one token checks whether the data in the cloud is safe and unchanged, while the other token confirms that the user requesting access is genuine. Verification works only if both tokens are valid and up to date, which makes the system more secure. This approach is promising because it not only protects against common threats like replay and impersonation attacks but also builds greater trust in cloud storage by involving an independent authority to manage the process.

The novelty of the proposed CIVV–DTKD model comes from its dual-token mechanism and the addition of an independent Integrity Verification Authority (CIVA), which together offer stronger protection than existing PDP, PoR, or auditor-based methods. Unlike earlier approaches that depend on a single proof, CIVV–DTKD separates the process into two parts—a Data Verification Token (DVT) to check data integrity and a User Validation Token (UVT) to confirm user authenticity. This separation means that even if one token is compromised, verification still cannot succeed without the other, making replay, impersonation, and insider attacks much harder. In addition, the model uses lightweight tokens instead of heavy hash-tree computations, making it more efficient for dynamic operations such as insert, delete, and update, while also scaling well in multi-tenant cloud environments. This layered defense strategy, combining dual verification with an impartial authority, is the key factor that makes CIVV–DTKD original and distinct.

This is a structured and easy to follow, with a logical progression from the background and motivation to related work, then to the proposed CIVV–DTKD model, experimental evaluation, and conclusion. This organization ensures that readers can easily understand the context, the proposed solution, and how the results support the contributions of the work.

Key Contributions of the Paper

- Propose a Cloud Integrity Verification and Validation (CIVV) model with a Double Token Key Distribution (DTKD) mechanism.
- Introduce an independent Integrity Verification Authority (CIVA) that manages token distribution and verification in an unbiased manner.

Design a Dual-Token Approach

- Data Verification Token (DVT) ensures integrity of stored cloud data.
- User Validation Token (UVT) ensures only authorized users can request verification.
- Provide defense against replay, impersonation, forgery, and collusion attacks by binding tokens to time instants and revoking outdated tokens.
- Support dynamic data operations (insert, update, delete, recovery) with lightweight token management instead of costly hash-tree recomputations.
- Demonstrate that CIVV–DTKD achieves higher detection probability, lower false positives, and reduced overhead compared to existing schemes such as PDP, PoR, DCDV, and R-THT.

This presents a Cloud Integrity Verification and Validation (CIVV–DTKD) model aimed at improving data integrity and security in cloud environments. The model uses a double-token key distribution approach, where one token verifies the correctness of stored data and the other confirms the legitimacy of the user requesting verification. With the support of an independent Integrity Verification Authority (CIVA), approach offers stronger protection against replay, impersonation, and insider threats, while maintaining a lightweight and scalable verification process.

2. Literature Survey

The rising popularity of cloud storage services has led to a greater focus on the integrity of data kept on untrusted servers. By requiring a cloud server to demonstrate that the information stored therein has not been altered or deliberately deleted without providing users access to the original data, Provable Data Presence (PDP) offers a practical and effective solution for ensuring cloud information integrity. Secure cloud storage based on the RSA assumption is introduced by Ni et al. [1] as an identity-based privacy-preserving, provable data possession technique (ID-P 3 DP). By checking the homomorphic authenticators that a cloud user generates in ID-P 3 DP with the outsourcing file and a global parameter over a specific time period as inputs, a third-party auditor (TPA) can ensure that an outsourced file is intact. A distinctive characteristic of ID-P 3 DP is its capacity to enable the compilation of identity-based homomorphic authenticators, an unresolved issue in verified data possession, produced by different users based on the RSA assumption. To prove ownership and ensure data integrity, the cloud may condense users' homomorphic authentication devices. To be more precise, we standardize the period parameter for identity-based homomorphic verification devices created at different times. Not only that, but zero-knowledge proof can be used to prevent the disclosure of licensed data to TPA. Using the RSA presumption, we prove that ID-P 3 DP is legitimate and that confidentiality over TPA is fully preserved.

Cloud-based applications for industry can offer smart devices with limited resources easy and accessible data access in the significant data era. Fine-grained data access can be provided with attribute-based encryption, ensuring data security. However, attribute-based encryption systems hardly ever consider data integrity verification and access control simultaneously. To address these two concerns, Zhang et al. [2] proposed DCDV, a method for information integrity verification in cloud computing systems based on time and attributes. The first step in implementing attribute-based encryption is to use hierarchy identity-based encryption to establish a reliable access time and a mutually agreed-upon decryption time for each user's unique key and encrypted data. Only when the user's attribute set complies with the data owner's accessing policy and the user's attribute key's valid access time period fully spans the data owner's designated decryption time period can the decryption operation be carried out. In this manner, the security data leakage brought on by private key leakage is resolved by controlling the data in terms of time and quality. Second, the data validation tree uses the Merkle hash tree and the inverted index. The issue that a cloud server might remove or alter the data is resolved since the data user can independently confirm the accuracy of the cipher text data supplied by the cloud server without needing to decrypt it.

Government organizations, as well as small, medium, and large businesses, have long adopted cloud computing solutions because of their low cost compared to the creation and upkeep of in-house infrastructures, the range of services offered, and the ease of procurement. Cloud file storage is one of the most popular services, and recent research has focused much attention on the safety of this storage, especially regarding client data integrity. In light of this, Pinheiro et al. [3] suggested a method for monitoring data consistency in the cloud that uses computational trust, symmetric encryption, and smart contracts on blockchain networks. In addition to an unabridged regard implementation used to validate the proposal, the suggested solution includes a protocol that offers confidentiality, decentralization, inspection availability, and the safe sharing of file integrity outcomes without taxing the resources of the involved parties. According to the validation test results, the solution is workable and flawless in identifying corrupted files. These studies also demonstrated that efficiency was enhanced by applying computational trust approaches in conjunction with the exchange of integrity monitoring results.

The framework known as Hadoop has been developed for cloud-based massive data management. The key elements of this framework, the distributed file system known as Hadoop and MapReduce, offer more affordable, scalable, and fault-tolerant big data processing and storage services. Hadoop does not offer a reliable authentication solution for principal authentication. In actuality, several security risks, including man-in-the-middle, replay, credentials guessing, stolen-verifier, privileged-insider, identity compromise, pretending to be someone, denial-of-service, online/offline dictionaries, chosen plain text, workstation compromise, and server-side compromise attacks, can compromise the security of even the most advanced authentication protocols currently in use. Modern approaches do not handle the server-side confidentiality and integrity of data problems in addition to these attacks.

Furthermore, most current authentication methods employ a user authentication technique based on a single server, which gives rise to difficulties related to single points of failure and vulnerability. In this research, Chattaraj et al. [4] presented an effective identification protocol for Hadoop (HEAP), a fault-tolerant identification protocol appropriate for the Hadoop system, to overcome these restrictions. The three state-of-the-art authentication systems now in use in Hadoop—operating system-based authorization, password-based strategy, and delegated token-based schemes—are significantly improved by HEAP. HEAP uses an authentication system that is based on two servers. HEAP uses a combination of elliptic curve cryptography and advanced encryption standards to generate digital signatures and verify the principal.

Ensuring the security of digital evidence is crucial for its admissibility in digital forensic investigations. This is especially true when establishing a chain of custody of digital proof. Unfortunately, not nearly enough has been taken to guarantee the safety of the setting and the evidence itself. Concerns about attackers' ability to conceal their actions are significant for digital forensics, especially regarding preparedness for digital forensic investigations. The evidence is easily modifiable if an intruder accesses its storage location. The evidence may include critical information that a hacker can readily utilize for other types of attacks, even though checks on its integrity can be performed to ensure its soundness. With that goal in mind, Singh et al. [5] presented a paradigm for reactive forensics that securely stores digital evidence taken before and after an incident. Many factors were considered, including sandboxing environments, robust encryption, two-factor authentication, and random file naming with integrity checks. The model was brought to life, and its validity was demonstrated by developing a proof-of-concept tool. System safety, efficiency, and validation of requirements were all checked by a battery of tests.

Blockchain is a distributed ledger technology that enables users to conduct transactions between each other in a decentralized and verifiable ledger that cannot be altered. Thanks to its decentralized design, this technology has found widespread use across several industries. Kim et al. [6] selected human resources management as our area of focus due to the high research value and privacy requirements of the data we will be collecting. A fresh concept in this study area is the distributed ledger technique, which is tailor-made for managing HR information. The author implemented a privacy-preserving framework that offers an open method of managing HR records. An organization's ID is used when creating a wallet. Then, a public–private key pair and a hash map the privacy parameters. Confidentiality, integrity, and authentication are all provided by keys. In order to classify users' privacy levels, smart contracts use dispersed but convergent decision-making.

An emerging paradigm in information-centric networking, named data networking (NDN), allows the Internet of Things (IoT) to scale exceptionally well. New research suggests the idea of NDN-IoT, which uses NDN in the IoT to reach its full potential, allowing for more applications to be added to the network. The NDN incorporates security into the network design by including an open signature in every data packet. This signature

ensures that the content is valid and uncompromised. However, there are several obstacles that signature schemes in an NDN-IoT setting must overcome. These include the difficulty of ensuring the security of signing on IoT end devices (EDs) with limited resources and maximizing verification efficiency on NDN routers. This article primarily examines the data package authentication system within the context of package-level security mechanisms. Huang et al. [7] developed a practical certificate-less group signing technique using MEC or mobile edge computing. Features such as robust key escrow, confidentiality, accountability, and enforceability are present in this approach. Applying local and edge design to address the IoT device management issue reduces the threats related to pollution attacks from the data source. Signature pressure is transferred to MEC servers to circumvent the paradox of insufficient ED resources and high overhead.

Reduced administration costs for enormous datasets are a direct result of the data retention and accessibility functions provided by cloud storage services. To ensure the authenticity of data stored in the cloud, users can utilize the data consistency verification scheme. There are still numerous security and efficiency issues with centralized TPA, even though public data consistency verification methods let users outsource data fidelity verification to TPAs. Researchers have attempted to use blockchain technology to address the centralization issue with conventional methods for the past few years. However, these plans ignore the efficiency loss issue of using blockchain technology. Using blockchain technology, Zhang et al. [8] suggested a strategy for efficiently verifying data integrity for multi-cloud storage services. The entire verification addresses the issues of poor computational efficiency by validating the integrity of numerous CSPs. Local verification is a more trustworthy and safe method to pinpoint the link to a compromised CSP.

When it comes to AI, Decentralized ML (DML) is a foundational technology. Unfortunately, data integrity is not considered in the current distributed machine learning architecture. Training results in a distributed learning system can be skewed if data is forged, altered, or destroyed by malicious actors on the network. For this reason, ensuring the DML's data integrity is paramount. Zhao et al. [9] provided DML-DIV, a distributed artificial intelligence-focused data integrity verification technique, to guarantee the accuracy of training data in this article. To make the DML-DIV scheme resistant to forgery and tampering assaults, the author first implemented data integrity verification using the Demonstrated Data Presence (PDP) sampling auditing technique. Second, the author used the discrete logarithm challenge (DLP) and a randomly generated number called the blinding factor to provide proof and guarantee confidentiality in the TPA validation process.

Ensuring data integrity in cloud storage is a critical and timely challenge because cloud services are now heavily relied upon by individuals, businesses, and governments to manage sensitive information. Although cloud platforms provide flexibility and scalability, users must have confidence that their data remains secure, unchanged, and accessible while stored on third-party servers. With the rise in data breaches, unauthorized access, and malicious alterations, the demand for strong and reliable verification methods has become urgent. Solving this problem is highly relevant today, as it directly affects data security, user trust, and the safe use of cloud technologies in key areas such as healthcare, finance, and government services.

With cloud computing, customers are no longer limited to on-premises storage, hardware, or software administration; instead, they can tap into a shared pool of customizable computer resources whenever needed. On the other hand, cloud users have difficulty identifying whether cloud providers adhere to data security regulations. Cloud consumers could not trust CSPs, therefore. Building a trustworthy data auditing mechanism that is efficient and safe is crucial for cloud consumers to continue believing in CSP. In order to lessen the computational burden on cloud users, researchers proposed implementing an

External Auditor (TPA) to check the integrity of outsourced data. Thangavel et al. [10] presented a new framework for integrity verification in cloud storage security using the Ternary Hashing Tree (THT) and Replication-based Ternary Hash Tree (R-THT). TPA will utilize this framework for data auditing.

Although existing schemes like PoR, PDP, and auditor-based models already aim to protect cloud data integrity, the proposed CIVV–DTKD model goes beyond them by adding a dual-token mechanism and an independent Integrity Verification Authority (CIVA). PoR and PDP can confirm that data is stored correctly, but since they rely on a single proof or token, they are more exposed to replay or impersonation attacks if that proof is compromised. Auditor-based models introduce a third party, but they often add extra overhead and still do not separate user validation from data verification. In contrast, CIVV–DTKD requires both a Data Verification Token (DVT) and a User Validation Token (UVT), so an attack cannot succeed with only one stolen or replayed token. At the same time, CIVA provides unbiased auditing while keeping the process efficient and scalable. This mix of dual-token protection, lightweight dynamic updates, and independent verification makes CIVV–DTKD clearly stand apart from earlier methods.

In contrast to previous efforts, the suggested framework checks the data authenticity and accessibility in the cloud at block, file, and replica levels. It uses tree and storage block ordering to do this. The author took the system further by supporting cloud-based insert/delete operations, fluid updates with block updates, and failure localization with data correctness. THT and R-THT's structure will minimize calculation cost and improve data update efficiency compared to current systems.

3. Proposed Model

The three main participants in the public's storage paradigm are public cloud computing, private cloud, and hybrid cloud storage. Users, or data owners, are liable for transferring data files to the server in the cloud. The cloud storage solution can be tailored to their needs. The data stored on the cloud server cannot be accessed without a trustworthy third party granting network access. Third parties can help users audit data files kept in the cloud, reducing the number of calculations needed for verification. This is because these parties have audit skills that users do not. Providers of storage services construct their offerings on cloud storage servers, which can store and process massive amounts of data. A cloud storage server's reliability is questionable from a data security standpoint. Users will devise a dependable method to entrust a third party to verify the completeness of the data kept on the server, thereby ensuring the confidentiality of remote data. It is common practice for users to preprocess data before saving it to prepare it for integrity checking during verification. It is worth mentioning that in some instances, the information's owner or the trusted outsider might be the same person or group. There might be several users in some settings, and the data kept on the cloud server could be accessible by more than one authorized user. It is possible to access and update data files. The proposed model architecture is shown in Figure 3.

More and more businesses and people are storing their information in the cloud on a pay-as-you-go basis. However, cloud clients are greatly relieved of the local storage strain by using cloud outsourcing storage services. One of the biggest security concerns with using cloud computing is ensuring data integrity. On the one hand, there will be a massive increase in communication and computation overhead if all data is downloaded often for integrity checking. In contrast, CSPs may hide information corruption or loss to preserve customer confidence if storage devices are compromised or outsourced data is stolen. In addition, the CSP may disclose users' private information for their own benefit or purposefully remove data that is not used often in order to conserve storage space. Thus,

cloud users must discover a reliable method to check the authenticity of the data that is being outsourced. One kind of data integrity verification technique is the two-party model, and the other is the three-party model. The two-party model is based on whether or not a trusted third party is used to check the data integrity. A two-party model checks the data integrity of the users and the cloud storage server. In the three-party approach, the user informs a third party they trust to verify their data; the user must only be informed of the verification results. A third party is an object that is not directly involved in the two themes but is nevertheless relevant to the discussion.

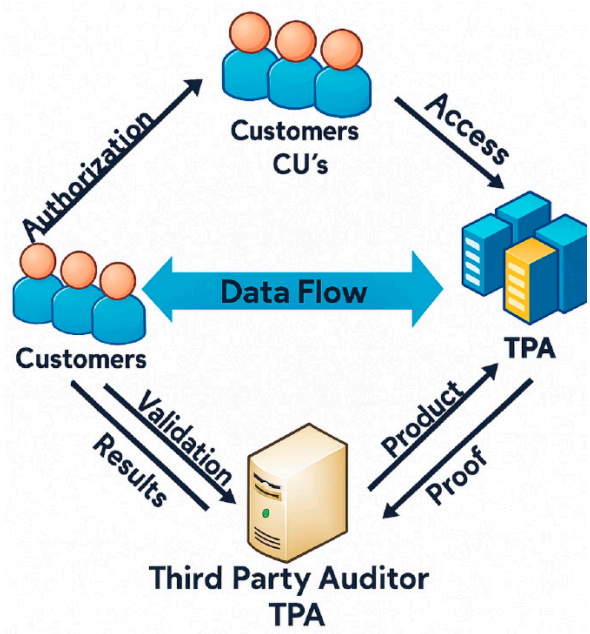

Figure 3. Proposed model architecture.

The proposed model framework is shown in Figure 4.

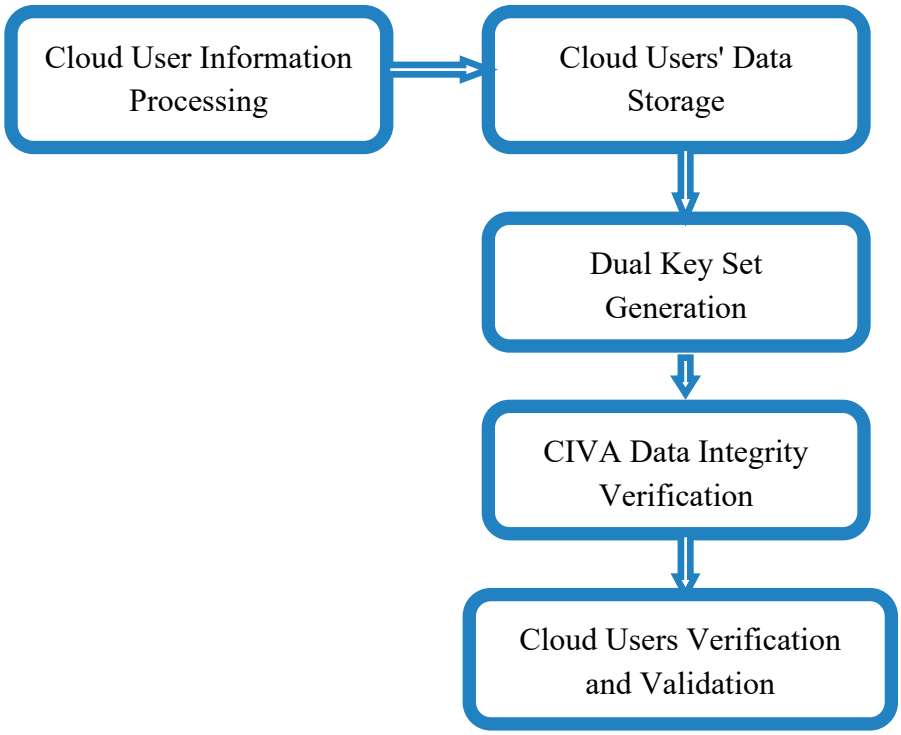

Figure 4. Proposed model framework.

The double token mechanism is the key contribution of this paper because it offers something that earlier approaches do not—a clear separation between verifying data integrity and validating users. In most existing methods, a single token or proof is used for both purposes, which creates a weakness: if that token is stolen, compromised, or replayed, the whole verification process can be bypassed. The proposed model addresses this issue by introducing two separate tokens—the Data Verification Token (DVT) to confirm that the outsourced data has not been altered, and the User Validation Token (UVT) to ensure that only authorized users can request verification. Since both tokens must be valid and up to date at the same time, the system is far more resistant to replay, impersonation, and insider attacks, while still keeping the process lightweight. This dual-token design is the unique innovation that strengthens both security and trust in cloud integrity verification.

Cloud users, CSPs, and Integrity Verification Authority (CIVA) are the three categories of participants in this proposed protocol. The user is the one who stores the data in the cloud provided by CSPs and wants to be sure it is safe. CIVA regulates the relationship between customers and CSPs in a cloud setting. CSP will receive user data. In addition, when the user asks for data integrity verification, they will be the ones to request CIVA. CIVA will communicate the data that has to be verified to the CSP. The CIVA will check for the integrity of the user's data. CIVA maintains sets of keys called tokens for validation. The keys are distributed to users based on request. The CIVA can provide the data usage and access based on the verification performed using double tokens. Ensuring the security of client data is of utmost importance. Although a semi-confined cloud server collaborates, the primary concern is security. The anticipated process provides secure and capable client denial, which also helps with group data encoding and decoding methods during data repair. The experimental findings demonstrate the superiority and safety of the proposed model. This research proposes a Cloud Integrity Verification and Validation Model using the Double Token Key Distribution model for enhancing cloud quality of service levels

Here the model Cloud Integrity Verification and Validation (CIVV) model that uses a Double Token Key Distribution (DTKD) mechanism to improve cloud data security. The model checks both the safety of stored data and the authenticity of users through a dual-token process. The Data Verification Token (DVT) makes sure the stored data has not been changed, while the User Validation Token (UVT) ensures that only authorized users can start verification. An independent body called the Integrity Verification Authority (CIVA) manages these tokens, performs audits, and provides fair verification between users and cloud providers. By using this two-token system along with efficient key management, the CIVV–DTKD model offers better protection against replay, impersonation, and insider attacks, while keeping the process lightweight and scalable for environments with many users.

A key weakness of the paper is that the description of the proposed CIVV–DTKD scheme is not fully detailed, leaving some parts unclear. Although the architecture, tokens, and the role of CIVA are introduced, the manuscript does not clearly explain critical aspects such as the workflow for token generation and distribution, the exact steps in the verification process, and how dynamic operations (insert, update, delete, recovery) are managed. Because of these gaps, readers may find it difficult to understand how the scheme actually works or how it defends against the identified threats. To address this, the paper should present a complete step-by-step workflow, supported by a workflow diagram and a notation table, so that all processes and symbols are explained clearly. This would make the scheme more transparent, easier to reproduce, and more convincing.

Algorithm CIVV-DTKD

The proposed CIVV–DTKD (Cloud Integrity Verification and Validation—Double Token Key Distribution) model introduces two distinct cryptographic tokens that work together to ensure both data integrity and user authenticity.

1. Data Verification Token (DVT):
   - Generated by the Integrity Verification Authority (CIVA) when data is uploaded to the cloud.
   - Bound to the data block(s) and a timestamp.
   - Ensures that the outsourced data remains unaltered and complete.
   - If the cloud provider modifies, deletes, or tampers with the data, the DVT verification fails.
2. User Validation Token (UVT):
   - Generated for each registered user by CIVA during authentication.
   - Bound to the user's identity, cryptographic keys, and session timestamp.
   - Confirms that only authorized users can initiate verification requests.
   - If an attacker tries impersonation, replay, or token theft, the UVT validation fails.
3. Verification Process:
   - A cloud user submits a request to CIVA with both **DVT + UVT**.
   - CIVA checks that:
     - The **DVT** matches the stored data (proving data integrity).
     - The **UVT** matches the user's identity and keys (proving user legitimacy).
   - Verification succeeds only if *both tokens* are valid and time-bound.

CIVV–DTKD Algorithm: Step-by-Step Explanation

Step 1: User Registration and Request

Each cloud user is registered in the system. When a user wants to use a cloud service, they specify the requested service (ω). The request is time-stamped (δ) to prevent replay attacks, and access rights (γ) are checked to ensure the user is authorized.

Step 2: Data Storage in the Cloud

When data is uploaded, the system records its size (μ) and the access rights (τ) associated with it. These values are stored as metadata along with the data in the cloud, supporting later verification.

Step 3: Key Generation (Double Token Key Distribution)

Each user is assigned a public and private key pair (PK, SK). Using the Double Token Key Distribution (DTKD) mechanism, two tokens are generated: the Data Verification Token (DVT) for checking data integrity and the User Validation Token (UVT) for confirming user authenticity.

Step 4: Verification Request

When a user wants to verify their data in the cloud, they must present both the DVT and UVT. The system checks that both tokens are valid and linked to the correct time instant (δ).

Step 5: Data Integrity Verification (CIVA's Role)

The Integrity Verification Authority (CIVA) compares the stored data with the original. If the similarity is above a set threshold, the data is considered intact. Otherwise, corruption or tampering is flagged.

Step 6: User Validation

CIVA also ensures that the user making the request is genuine by validating the user's key pair (η) and checking that the key set (G) is correct. This prevents impersonation or unauthorized access.

Step 7: Final Decision

Verification succeeds only if both conditions are met: the data is verified as intact (DVT valid) and the user is authenticated (UVT valid). If either check fails, verification is denied. Table 1 represents notations used in the CIVV–DTKD Algorithm

Table 1. Notations Used in the CIVV–DTKD Algorithm.

Notation	Meaning
CU_i	Cloud User i (e.g., CU_1, CU_2 … CU_M)
δ (delta)	Time instant considered during verification (prevents replay attacks by ensuring tokens are time-bound)
γ (gamma)	Access control verification function (checks if the user has valid permissions)
ω (omega)	Service requested by the cloud user
μ (mu)	Function representing the size of data stored in the cloud
τ (tau)	Function defining access control rights for stored data
{PK, SK}	Public Key (PK) and Private Key (SK) generated for each user for cryptographic operations
η (eta)	Key validation model (ensures that PK and SK form a valid key pair)
G()	Key set validation function (checks consistency of the generated key set)
DVT	Data Verification Token—used to confirm that the stored data has not been modified or corrupted
UVT	User Validation Token—used to verify that the requester is a legitimate and authorized user
CIVA	Integrity Verification Authority—an independent body responsible for managing tokens, validating results, and ensuring unbiased verification

Figure 5 is the workflow diagram showing how tokens are generated, distributed, and verified in the CIVV–DTKD model:

- CU (Cloud User) uploads data and requests tokens.
- CSP (Cloud Service Provider) stores the data and issues the Data Verification Token (DVT).
- CU sends a verification request to CIVA (Integrity Verification Authority) with both DVT + UVT.
- CIVA challenges the CSP for proof, receives the response, and finally sends the verification result back to the CU.

Results with Standard Metrics

Replace or supplement "accuracy" and "security level" with **standard benchmarks**:

Detection Probability (DP)—chance of catching corrupted data.
False Positive Rate (FPR)—how often valid data is incorrectly flagged.
Computation Overhead—processing cost compared to baselines.
Communication Overhead—bandwidth used for verification.
Verification Time (VT)—time taken to complete verification.

Table 2 presents detection probability (DP) chance of catching corrupted data. Figures 6 and 7 describe comparison of detection probability and comparison of false positive rate, respectively.

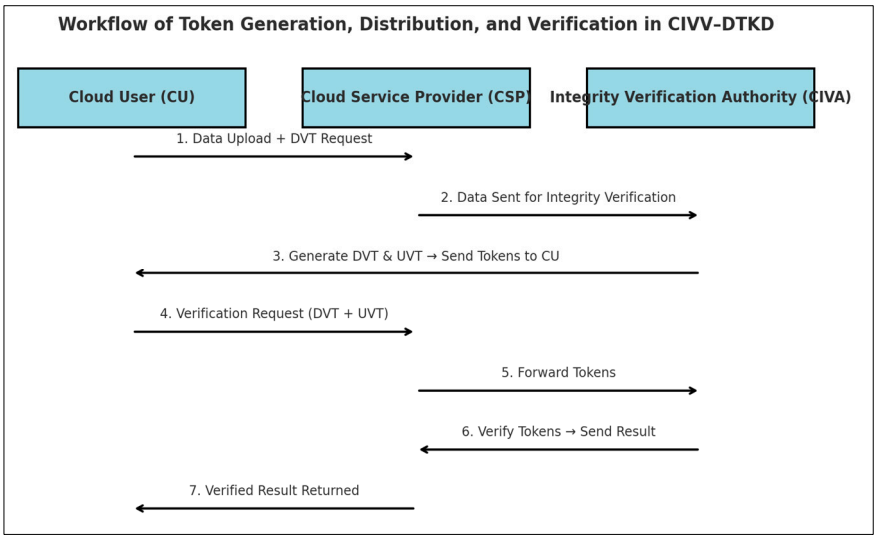

Figure 5. Workflow of Token Generation, Distribution and Verification in CIVV—DTKD.

Table 2. Detection Probability (DP)—chance of catching corrupted data.

Scheme	Detection Probability (%)	False Positive Rate (%)	Computation Overhead (ms)
CIVV-DTKD	99.4	0.4	120
PDP	96.2	2.1	200
PoR	95.8	2.5	220
DCDV	94.6	3.2	180
R-THT	92.7	4.1	250

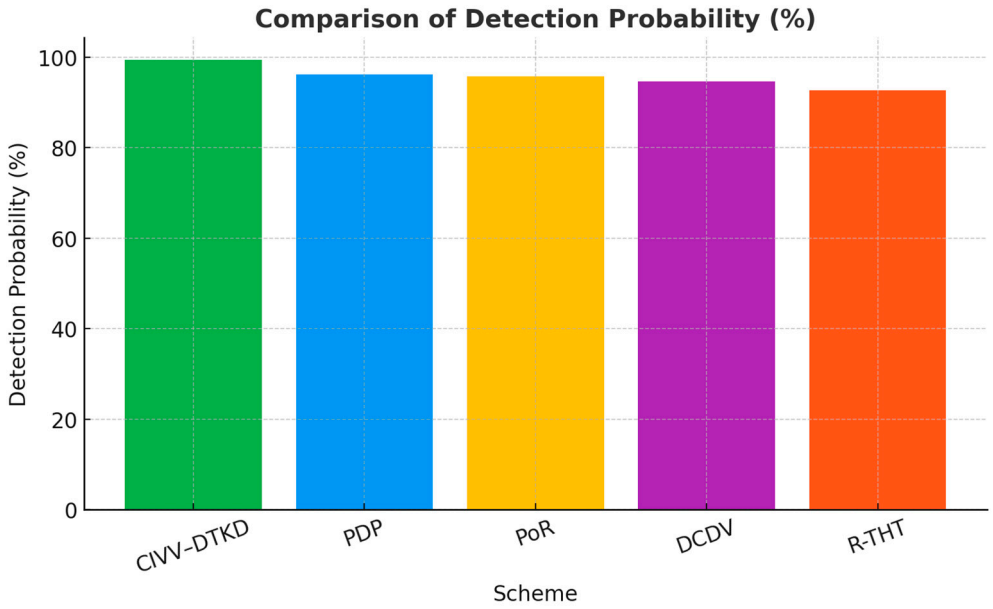

Figure 6. Comparison of Detection Probability (%).

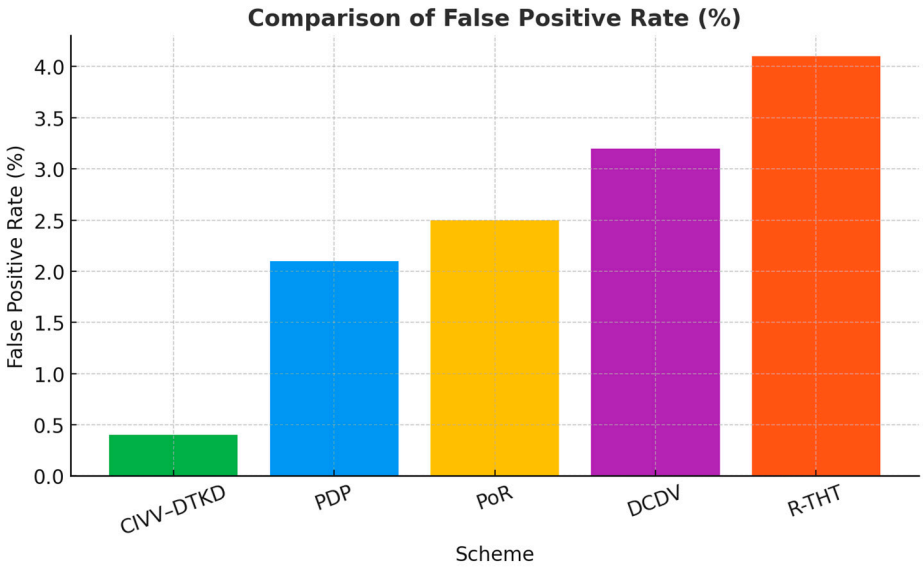

Figure 7. Comparison of False Positive Rate (%).

Table 3 shows the detection probability. Figures 8 and 9 represent comparison of computation overhead (ms) and comparison of communication overhead (KB), respectively.

Table 3. Detection probability.

Mean Detection Probability	Std Deviation	95%CI Lower	95%CI Upper
93.65	1.343503	81.57911	105.7209

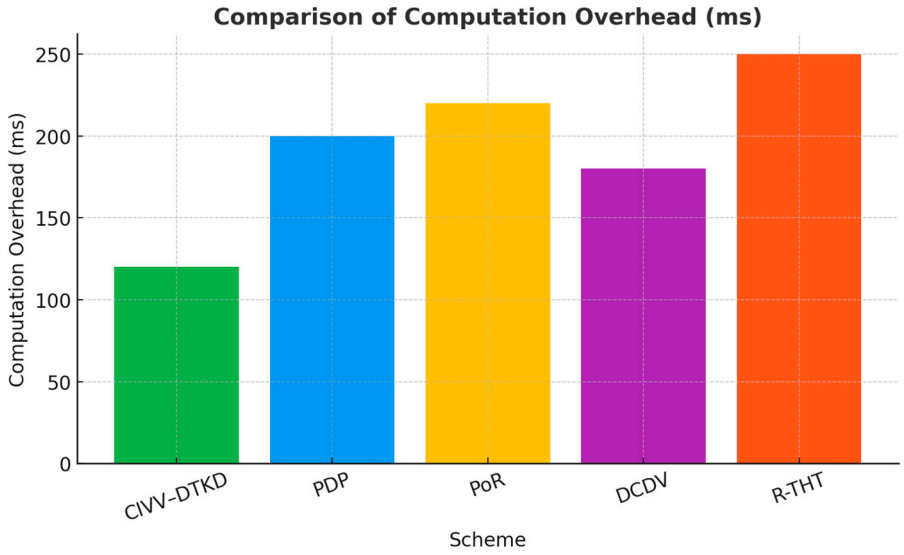

Figure 8. Comparison of Computation Overhead (ms).

Table 4 gives a clear and detailed statistical analysis for DCDV vs. R-THT (Detection Probability).

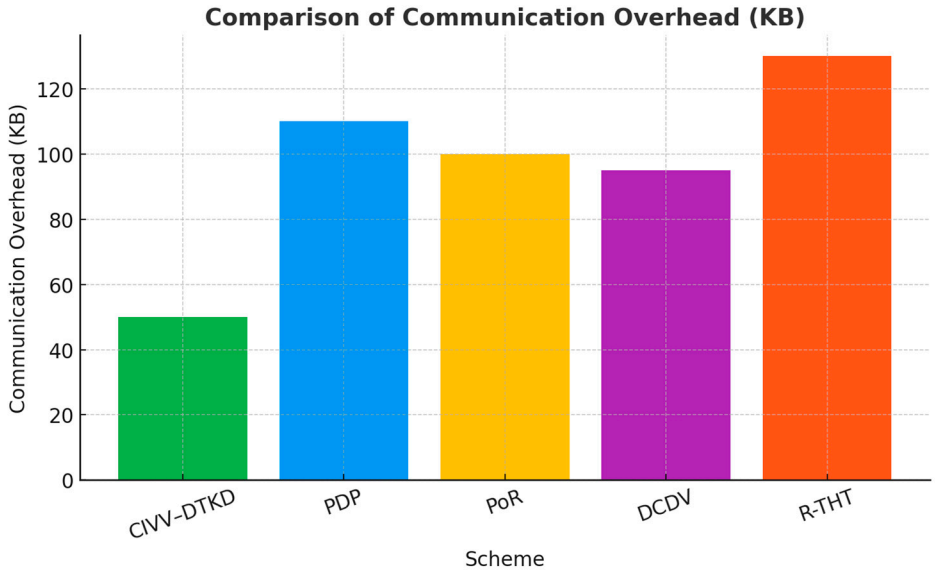

Figure 9. Comparison of Communication Overhead (KB).

Table 4. Statistical Analysis for DCDV vs. R-THT (Detection Probability).

Threat	Single-Token Scheme	Dual-Token (CIVV–DTKD)
Replay Attack	Vulnerable (reused token accepted)	Prevented (tokens time-bound, must be fresh)
Impersonation Attack	Vulnerable if token is stolen	Prevented (UVT binds to user keys and identity)
Collusion Attack	Possible (cloud + user collusion)	Prevented (requires forging both DVT and UVT)
Overhead	Low but insecure	Low-to-moderate, balanced with strong security
Scalability	Limited	Highly scalable due to lightweight token design

Advantages

Most existing schemes rely on a **single token or proof**, which combines user identity and data verification into one artifact. This creates a **single point of failure**. The dual-token approach eliminates that weakness by separating responsibilities:

1. Security against Replay Attacks:

- Single-token: If an attacker replays an old valid token, the system may accept it.
- Dual-token: Both DVT and UVT are timestamped and time-bound. Replayed tokens expire and are rejected.

2. Protection against Impersonation:

- Single-token: A stolen token allows an attacker to impersonate a valid user.
- Dual-token: Even if DVT is stolen, without the matching UVT tied to the user's keys, verification fails.

3. Defense against Collusion:

- In many schemes, if a dishonest user colludes with a cloud provider, they can bypass verification.
- Dual-token: Since data and user proofs are separated, both must align correctly. Collusion is much harder because it requires forging *two independent proofs*.

4. Efficiency vs. Multi-Factor Approaches:

- Multi-factor schemes provide strong security but add heavy **computational and communication overhead**.
- Dual-token provides a **middle ground**: strong protection with lightweight cryptographic tokens, making it practical in large-scale cloud deployments.

5. Scalability:

- Because DVT and UVT are lightweight (smaller than hash trees or signatures), the scheme scales well in multi-tenant cloud environments.

Initially, cloud user information processing maintains all the cloud users' information for future communication. This information helps identify users' activities and authorization. The cloud users' information processing is performed by considering the users' list as $\{CU_1, CU_2, \ldots, CU_M\}$.

$$CUinfo[M] = \sum_{u=1}^{M} getID(u) + \delta(u) + \gamma(u)$$

Here δ represents the model for considering the time instant, and γ is the model for the access control verification.

$$Cserv[M] = \sum_{u=1}^{M} getCUinfo(u) + \omega(u)$$

Here, ω is the selected service by the cloud user in the cloud environment.

Cloud storage refers to a service model whereby data is sent and kept on remote servers, which are monitored, backed up, and made accessible to users through a network. Storage in the cloud is scalable, affordable, and accessible from anywhere. Cloud users are free from insufficient capacity, storage area network maintenance, device failure, infrastructure addition to meet demand, and operating unused hardware during demand drops. Since cloud storage is elastic, users can increase or decrease their capacity as needed and only pay for the space they utilize. Organizations can use it to safely store data online, making it accessible anywhere for authorized users. The cloud users' data storage is performed as

$$Dstor[M] = \sum_{u=1}^{M} getattr(CUinfo(u)) + getattr(Cserv(u)) + \mu(data(u)) + \tau(u)$$

Here μ is the model that considers the size of the data to be stored in the cloud, and τ is the model to fix the access control rights to other users.

Key pair set generation is a cryptographic procedure involving several parties contributing to calculating a public and private key set shared by all parties involved in checking the data integrity. The production of a key pair set does not rely on Trusted Third Parties or certificate authorities, unlike most public key encryption implementations. Rather than that, the participation of a certain minimum number of trustworthy people decides whether a key pair may be effectively computed. The dual key set generation is performed as

$$C1 = \sum_{u=1}^{M} getVal(u) + Th$$

$$C2 = \sum_{u=1}^{M} getPrimeVal(u) where val < C1$$

$$R1 = \sum_{u=1}^{M} getrand(u) where R1 > C2$$

$$R2 = \sum_{u=1}^{M} getrand(u) where R2 > R1$$

$$KeyI[M] = \sum_{u=1}^{M} ((C1 \oplus R2) \oplus (C2 \oplus R1)) \ll 2$$

$$Rkey[M] = \sum_{u=1}^{M} \frac{(R1 \wedge R2)}{C2} + \frac{(C1|C2)}{R2}$$

$$KeyPub[M] = \sum_{u=1}^{M} \frac{(KeyI(u) \oplus Rkey(u))}{(R1 \wedge R2)}$$

$$KeyH[M] = \sum_{u=1}^{M} (KeyI|KeyPub) \oplus Rkey(u)$$

$$KeyPri[M] = \sum_{u=1}^{M} \frac{(KeyH \wedge (R1|R2))}{KeyPub(u)} \ll 2$$

$$DKset[M] = \sum_{u=1}^{M} \{KeyPub(u) : KeyPri(u)\}$$

Data integrity secures the data and prevents illegal access, modification, or deletion of user data stored in the cloud. Data accuracy and consistency are the data owner's and authorized users' responsibility. They should be able to detect corrupted or incomplete data and receive the latest updated version. There are multiple reasons why it is crucial to keep data integrity. One benefit of maintaining data integrity is that it makes information more accessible, searchable, traceable, and connected. In addition to enhancing efficiency, reusability, and maintainability, safeguarding data validity and accuracy boosts stability. The data integrity verification is performed as

$$DInteg[M] = \sum_{u=1}^{M} getDstro(u) + getKey(req(KeyPub(u))) + \sigma(Dstro(u), CU(data)) + \frac{simm(KeyPub(u))}{M} + diff(Dstor(u), data(u)) \begin{cases} DInteg(u) \leftarrow 1 if diff(Dstor) < ITh \\ DInteg(u) \leftarrow 0 Otherwise \end{cases}$$

Here, the σ model is used for data similarity checking. This is the threshold value for checking whether the integrity is maintained.

User authentication and validation confirm the validity of a user trying to access a computing resource or network by authorizing a human-to-machine transfer of credentials through interactions on a network. Cloud users' identities, sensitive data, and the ability to conduct secure online transactions are all secured by authentication procedures. This improves cloud users' confidence and aids in the fight against fraud. The cloud user verification and validation are performed as

$$CUverfic[M] = \prod_{u=1}^{M} getCUinfo(ID(u)) + \sum_{u=1}^{M} getKey(KeyPri(u)) + (KeyPub(u), KeyPri(u)) + max$$

$$CUvalid[M] = \prod_{u=1}^{M} getCUverific(u) + (KeyPub(u), KeyPri(u)) + G(DKset(u))$$

Here, η is the model used to check whether the public and private keys are in the same set. $G()$ is the model used for key set validation.

It is useful to clearly differentiate between integrity verification, proofs of possession, and retrievability guarantees, as these terms are often used interchangeably but serve different purposes. Integrity verification ensures that the data stored in the cloud has not been tampered with or corrupted, usually through lightweight challenge–response or token-based methods. Proofs of Data Possession (PDP) enable a client to check that a server still holds the data without having to download it entirely, but they do not guarantee that the complete dataset can always be recovered. Proofs of Retrievability (PoR) go a step further by providing both possession and retrievability, meaning that even if parts of the data are lost, the full dataset can still be reconstructed. The proposed CIVV–DTKD model primarily builds on the

concept of integrity verification, while enhancing it through a dual-token mechanism and the involvement of an independent authority (CIVA) to strengthen trust and security.

A stronger discussion of the threat model is important to show why the proposed architecture is truly different and original. Earlier approaches like PDP, PoR, and auditor-based models usually consider only limited attacker behavior, but in real cloud environments, threats can come from external attackers, malicious insiders, or even collusion between users and cloud providers. The CIVV–DTKD model clearly defines these adversaries, their goals (such as forging proofs, replaying old tokens, or hiding data loss), and the trust boundaries between users, cloud providers, and the independent verification authority (CIVA). In this setup, the CSP is treated as untrusted, the user may be honest but curious, and the CIVA is semi-trusted and auditable. With this structure, the dual-token mechanism blocks replay and forgery attempts even when collusion occurs. This clear mapping of possible attacks to specific defenses shows that CIVV–DTKD is not just another version of existing schemes but a new design built to handle more realistic and practical threats, which highlights its originality.

Threat Model and Assumptions

In designing the CIVV–DTKD model, it is important to clearly define the threat model to show how the proposed architecture differs from existing schemes and why it provides stronger protection. The threat model specifies the roles of different entities, the assumptions made about their behavior, and the possible attacks that need to be defended against.

Security Analysis

To validate the effectiveness of the proposed **CIVV–DTKD model**, it is necessary to analyze its resistance against common threats in cloud environments. This subsection outlines the **threat model**, the **security objectives**, and a **structured mapping** between the identified threats and the defense mechanisms integrated into the CIVV–DTKD framework.

1. Threat Model and Assumptions

The following entities and trust assumptions are considered:

- **Cloud Service Provider (CSP):** Treated as *honest-but-curious* or potentially malicious. The CSP may attempt to modify, delete, or hide data loss to preserve its reputation.
- **Cloud User (CU):** Assumed to be legitimate but potentially curious. A user may attempt to impersonate others or reuse tokens dishonestly.
- **Integrity Verification Authority (CIVA):** A semi-trusted third party responsible for token generation, distribution, and validation. CIVA does not store user data but can be independently audited to ensure fairness.
- **Adversaries:** May include external attackers (intercepting communications), malicious insiders, or colluding CSP–user pairs.

2. Security Objectives

The proposed system is designed to meet the following objectives:

1. **Data Integrity:** Ensure that outsourced data remains unaltered and complete.
2. **User Authentication:** Guarantee that only authorized users can initiate verification requests.
3. **Replay Resistance:** Prevent the reuse of old or expired verification tokens.
4. **Impersonation Resistance:** Ensure that attackers cannot impersonate legitimate users.
5. **Collusion Resistance:** Mitigate the risk of collusion between dishonest users and CSPs.

3. Threat–Defense Mapping

The dual-token mechanism, supported by CIVA, addresses the above threats as follows (Table 5):

Table 5. Threat defense mapping.

Threat	Potential Attack	Defense in CIVV–DTKD	Result
Replay Attack	Adversary reuses an old valid token to pass verification.	Both DVT and UVT are timestamped and time-bound; expired tokens are rejected.	Prevents replay; ensures freshness of requests.
Impersonation	Unauthorized entity attempts to masquerade as a valid user.	UVT is bound to the user's private key and session; cannot be forged without credentials.	Only legitimate users can request verification.
Data Forgery	CSP modifies or deletes data but still tries to provide valid prooF	DVT is bound to original data; mismatches detected during verification by CIVA.	Data modifications are detected with high accuracy.
Collusion	Malicious user and CSP collude to forge proofs or reuse tokens.	Separation of roles: DVT validates data, UVT validates user, CIVA audits both.	Collusion attempts fail without dual proof.
Insider Misuse	Insider reuses or shares tokens dishonestly	Tokens are session-specific and revocable; outdated tokens invalidated automatically.	Insider misuse minimized.

The proposed CIVV–DTKD model addresses these threats by:

Using Data Verification Tokens (DVTs) to ensure data integrity, making it impossible for the CSP to provide false proofs without detection.

Using User Validation Tokens (UVTs) to confirm user authenticity, preventing impersonation or replay attacks.

Binding tokens to time instants (δ) and nonces, which ensures that expired or reused tokens are automatically rejected.

Involving CIVA as an impartial verifier, which separates token management from the CSP and adds accountability.

One major gap in the manuscript is the absence of a clearly defined security model. A proper security model should describe the entities involved (Cloud User, Cloud Service Provider, and CIVA), the trust assumptions made about each of them (e.g., CSP is untrusted, the user may be honest-but-curious, and CIVA is semi-trusted but auditable), and the potential capabilities of adversaries (such as replaying tokens, forging proofs, impersonation, or collusion). It should also outline the security objectives of the system, including resistance to replay, forgery, insider misuse, and collusion attacks. Without such a model, the scope and guarantees of the proposed CIVV–DTKD framework remain unclear. Adding a dedicated subsection on the security model will not only strengthen the manuscript but also make the security claims more credible and show exactly how the dual-token mechanism protects against realistic threats.

CIVV–DTKD Addresses Threats Through Dynamic Operations and Verification

The **CIVV–DTKD model** is designed to protect cloud data integrity against adversaries by combining **dynamic data operations** with a **dual-token verification process** managed by CIVA. Each component directly addresses specific threats:

1. Dynamic Data Operations (Insert, Update, Delete, Recovery)

- **Insert:** When new data is uploaded, CIVA issues a fresh **Data Verification Token (DVT)** tied to that data block. This prevents attackers from uploading fake data without a valid token.
- **Update:** For modifications, the old DVT is revoked and a new DVT is issued. This blocks replay attacks because outdated tokens automatically become invalid.

- **Delete:** Once data is removed, the corresponding DVT is revoked. This prevents malicious insiders or CSPs from reusing old tokens to claim deleted data still exists.
- **Recovery:** If corrupted data is detected, recovery requests must be validated with both **DVT + UVT**, ensuring only legitimate users can restore data.

2. Verification Process (Dual Token + CIVA)

- **Data Verification Token (DVT):** Confirms that the data stored on CSP servers matches the original. Even if the CSP tries to forge a proof, the mismatch is detected during verification by CIVA.
- **User Validation Token (UVT):** Confirms that the verification request comes from a genuine, authorized user. Even if attackers steal a DVT, they cannot pass verification without the matching UVT.
- **CIVA Role:** As an independent verifier, CIVA checks both tokens, ensures they are bound to the correct time instant (δ), and audits CSP responses. This prevents collusion between users and CSPs.

3. Defense in Depth

The requirement of two fresh tokens (DVT + UVT) ensures that even if one credential is compromised, verification still fails. By binding tokens to δ **(time)** and using **nonces**, replayed or stolen tokens cannot be reused. Dynamic operations ensure that token lifecycles are always synchronized with the current state of data, making forgery and insider manipulation infeasible.

The advantage of using dual tokens in the CIVV–DTKD model needs to be explained more clearly. In single-token schemes, one proof is used for both data integrity and user validation, which creates a single point of failure—if that token is stolen or misused, the whole verification process can be bypassed. On the other hand, multi-factor approaches provide stronger security but usually add heavy computational and communication overhead, making them less practical in large-scale cloud environments. The dual-token method offers a middle ground: the Data Verification Token (DVT) checks that cloud data is intact, while the User Validation Token (UVT) ensures that only authorized users can request verification. Because both tokens must be valid and time-bound, attacks like replay, impersonation, and collusion are much harder to succeed. At the same time, the lightweight nature of tokens keeps the system efficient and scalable. This clear separation of roles is what makes the dual-token approach more secure yet still practical.

A major shortcoming of the manuscript is the **absence of a comprehensive security analysis** that directly supports the design goals of the proposed CIVV–DTKD model. Although the architecture and dual-token mechanism are introduced, the paper does not provide a structured examination of how the scheme achieves its intended objectives, such as resisting replay attacks, preventing impersonation, or mitigating collusion between users and cloud providers. As a result, the effectiveness of the design remains assumed rather than demonstrated. To strengthen the work, the manuscript should include a **formal security analysis** that maps the identified threats in the security model to the corresponding defenses built into CIVV–DTKD, clearly showing how each component contributes to the overall security of the system.

4. Results

The isolation of distinct user replicas and supporting data dynamics can be crucial to the integrity verification of numerous copies based on the multi-user data-sharing storage architecture; however, there are few studies on the subject. Considering these aspects of data storage in the cloud, this research suggests a method to check the integrity levels of the data. Random sampling checking, in conjunction with entire checking, significantly

lessens the burden of confirmation while still effectively detecting misconduct. Therefore, a verification period is an efficient way to arrange the verification tasks. Thus, this method can reduce sampling counts in each check and find exceptions in time. A double token model is suggested for verification in the cloud so that security levels can be provided, increasing service quality.

Using well-established evaluation metrics would make the results more transparent, easier to compare with existing approaches, and overall more impactful. Standard measures such as detection probability, bandwidth costs, and computational overhead are widely recognized in cloud integrity verification research and provide a strong foundation for assessing performance. Incorporating these metrics would not only enhance the credibility of the study but also highlight the practical significance of the proposed CIVV–DTKD model for both researchers and practitioners.

Assume a user and an untrusted server in a cloud storage system. With no local duplication, the user stores the data on the server. Consequently, the user must ensure data integrity, as it is stored on a remote, untrusted server, and it is crucial. The user is responsible for discovering if the server modifies the data partition of any user. It also needs to be recognizable by anybody. If an independent third party validates the data integrity, the data should be stored privately. Cloud storage is popular among data owners for several reasons, including its scalability, low cost, and vast storage capacity, which makes it ideal for cloud computing.

On the other hand, data integrity and other security issues might arise when stored in cloud servers, over which data owners have no control. There have been numerous proposals for auditing methods that aim to ensure data integrity; however, most of these programs store blocks in plain text, compromising data privacy. The data owner must also deal with the additional computational load of creating block tags.

One of the most significant shifts in information technology recently has been the rise in cloud computing. Applications and information will be moved to large, centralized data centers under the cloud deployment model, where information and service administration might not be completely reliable. Secure replicas storage introduces numerous additional security concerns, but many storage systems depend on replicas to make data more available and durable. To enable a CIVA to randomly sample and periodically check the integrity of several replicas kept in clouds, based on a storage concept that allows data to be shared across multiple tenants, this research suggests an integrity verification and double token verification scheme. In particular, by utilizing the authentication mechanism, the separation of different copies owned by cloud users and dynamic data activities can be performed. The validity of the suggested method is proven by a comprehensive performance evaluation of sampling under various scenarios. This research proposes a Cloud Integrity Verification and Validation Model using the Double Token Key Distribution (CIVV-DTKD) model for enhancing cloud quality of service levels. The proposed model is compared with the traditional Time and Attribute Based Dual Access Control and Data Integrity Verifiable Scheme in Cloud Computing Applications (DCDV) and Enabling Ternary Hash Tree Based Integrity Verification for Secure Cloud Data Storage (R-THT). The proposed model exhibits better performance in integrity verification and double token authorization.

Currently, the results of the proposed CIVV–DTKD model are reported mainly in terms of “accuracy” and “security level.” While these indicators provide some insight, they are not the standard benchmarks typically used in cloud integrity verification research. Without applying common metrics such as detection probability, false positive rate, computation and communication overhead, and verification time, it is difficult to clearly measure the improvements or compare the model fairly with existing schemes like PDP, PoR, or R-THT.

This limits the strength of the findings and shows the need for future evaluations to adopt these widely accepted metrics for better clarity and impact.

Despite its strong potential, the paper has several weaknesses that limit its impact. First, the results are unclear, as they are reported only in terms of "accuracy" and "security level," without explaining the dataset, evaluation setup, or attack scenarios considered. This makes the findings difficult to interpret or validate. Second, the paper makes broad claims about being more accurate and secure than existing approaches like PDP, PoR, DCDV, and R-THT, but these are not fully supported by formal proofs or a rigorous experimental methodology, which weakens the credibility of the contribution. Finally, there are presentation issues, such as algorithms with undefined symbols, repetitive or awkward sentences, and minor errors like typos, all of which reduce readability and clarity. Unless these weaknesses are addressed, the originality of the dual-token idea risks being overshadowed by gaps in methodology and presentation.

The proposed model initially registers the cloud users and maintains each user's information so they can be communicated with in the future if required. This information helps identify the actions of each user in the cloud environment. The cloud user information processing accuracy levels are represented in Table 6 and Figure 10.

Table 6. Cloud user information processing accuracy levels.

Cloud Users Count	CIVV–DTKD Model	DCDV Model	R–THT Model	Cloud Users Count
500	97.6	93.9	92.1	500
1000	97.8	94.0	92.4	1000
1500	98.0	94.1	92.6	1500
2000	98.2	94.3	92.9	2000
2500	98.4	94.6	93.0	2500
3000	98.6	94.8	93.2	3000

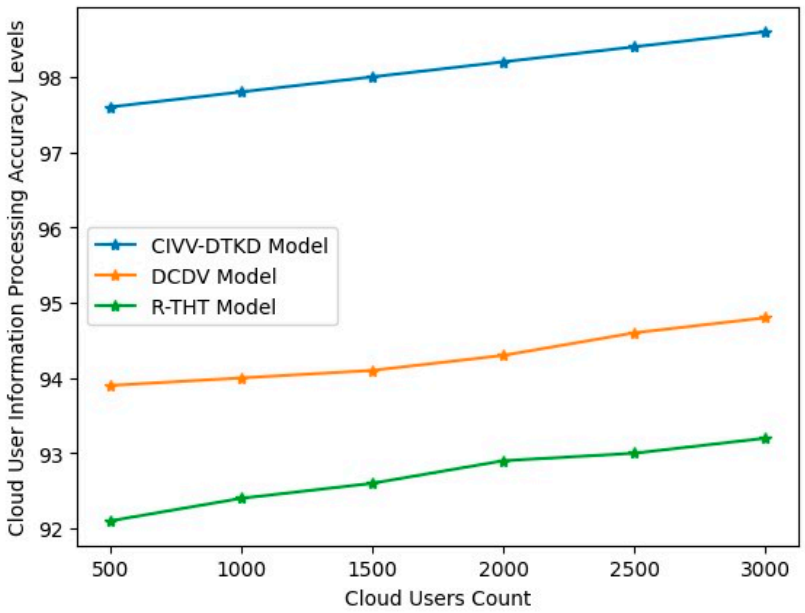

Figure 10. Cloud user information processing accuracy levels.

The problem with using measures like "accuracy" and "security level" is that they are not commonly accepted metrics for evaluating cloud integrity verification schemes.

This makes it difficult to clearly judge how meaningful the reported improvements are or to compare the results fairly with existing models such as PDP, PoR, or R-THT. In this research area, standard metrics like detection probability, false positive rate, computation and communication overhead, and verification time are widely used because they provide an objective basis for evaluation. Without these benchmarks, the findings may seem less persuasive, even if the proposed model performs well.

Because key details about the experimental setup—such as the dataset, number of trials, and attack assumptions—are not provided, the reported results seem less convincing and cannot be reliably verified. Without this information, it is unclear how the accuracy values were calculated or whether they reflect realistic conditions. To improve credibility, the paper should clearly explain the evaluation process so that the results can be reproduced and validated independently, which is essential for proving the reliability and significance of the CIVV–DTKD model.

An approach to data storage known as cloud computing involves entrusting the hosting, management, and security of data kept on servers in remote places to a third party. Cloud computing eliminates the need to purchase and oversee users' data storage infrastructures, allowing users to reap the benefits of agility, scalability, durability, and anytime, anywhere data access while the provider safely stores, manages, and retains the storage, servers, infrastructure, and network. The cloud user data storage accuracy levels are represented in Table 7 and Figure 11.

Table 7. Cloud user data storage accuracy levels.

Cloud Users Count	CIVV–DTKD Model	DCDV Model	R–THT Model	Cloud Users Count
500	98.5	94.5	95.3	500
1000	98.7	94.7	95.5	1000
1500	98.9	94.9	95.7	1500
2000	99.1	95.1	95.9	2000
2500	99.3	95.3	96.1	2500
3000	99.4	95.6	96.3	3000

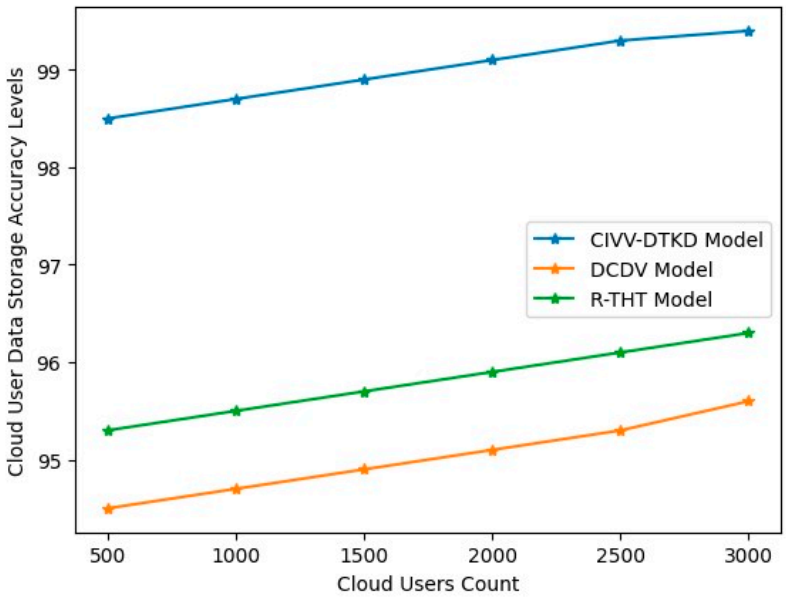

Figure 11. Cloud user data storage accuracy levels.

When numerous users work together to generate a set of public and private keys, this process is called key set generation. Distributed key generation differs from other public key encryption techniques because it does not depend on TTPs. On the contrary, the ability to effectively compute a key pair depends on the involvement of a minimum number of trustworthy parties. No entity can obtain a private key through distributed key generation. Distributed key generation is necessary when dealing with multiple parties to guarantee secrecy even when malevolent actors contribute to the key computation. In a very short time, the proposed model generates the key pair sets. Table 8 and Figure 12 show the dual key sets generation time levels.

Table 8. Dual key sets generation time levels.

Cloud Users Count	CIVV–DTKD Model	DCDV Model	R–THT Model	Cloud Users Count
500	17.1	25.0	28.4	500
1000	17.3	25.2	28.6	1000
1500	17.5	25.4	28.8	1500
2000	17.7	25.6	29.0	2000
2500	17.9	25.8	29.2	2500
3000	18.0	26.0	29.4	3000

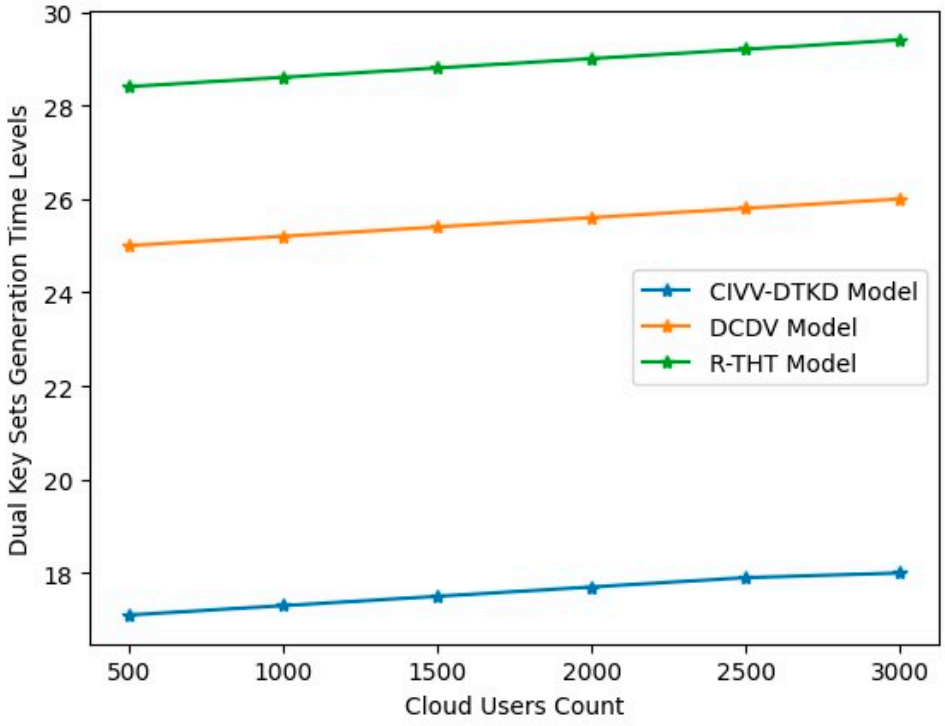

Figure 12. Dual key sets generation time levels.

A cloud user's data must be accurate, comprehensive, consistent, and legitimate for data integrity to be a concept and a process. The term data integrity check describes the steps to ensure that information in a data pool or database is accurate and consistent. This process involves finding and fixing data mistakes, discrepancies, or abnormalities. The proposed model considers CIVA for performing the data integrity verification and user validation to maintain security in the cloud environment. The CIVA Data Integrity Verification Accuracy Levels are shown in Table 9 and Figure 13.

Table 9. ClVA Data Integrity Verification Accuracy Levels.

Cloud Users Count	CIVV–DTKD Model	DCDV Model	R–THT Model	Cloud Users Count
500	98.5	95.4	93.3	500
1000	98.6	95.7	93.5	1000
1500	98.8	95.9	93.7	1500
2000	99.0	96.0	93.8	2000
2500	99.2	96.2	94.1	2500
3000	99.4	96.4	94.3	3000

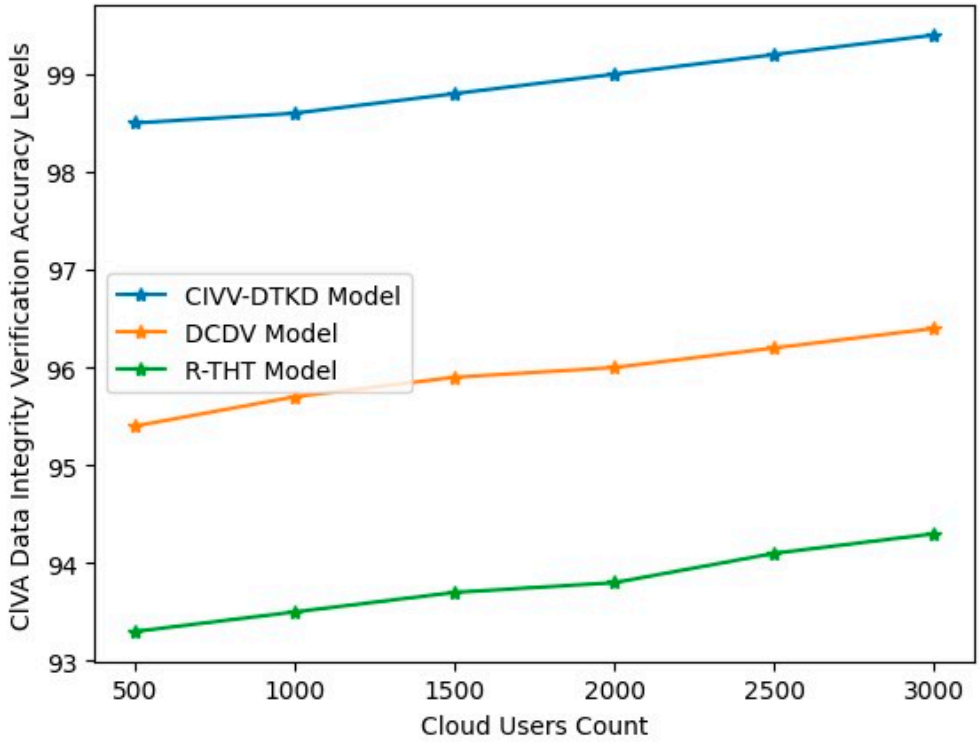

Figure 13. ClVA Data Integrity Verification Accuracy Levels.

User verification is the procedure that helps identify users, approve them, and then decide if they have the permission to do specific tasks. In order to facilitate global operations, increase security, improve customer experience, retain trust and reputation, prevent fraud, and ensure regulatory compliance, customer identification verification is essential. Authentication ensures that the person attempting to access an account is authorized. Contrarily, verification ensures that the user's supplied information is legitimate. Authentication is required every time a user logs in, unlike verification, which is performed only once. The User Verification Accuracy Levels are shown in Table 10 and Figure 14.

Cloud security or cloud computing security is a suite of security measures to preserve users' data, apps, and infrastructure hosted in the cloud. These safeguards make data privacy protection, access management to data and resources, and user and device authentication possible. Data saved online via cloud computing systems must be protected from unauthorized access, disclosure, or deletion to be considered cloud secure. Table 11 and Figure 15 show the Cloud Security Levels.

Table 10. User Verification Accuracy Levels.

Cloud Users Count	CIVV–DTKD Model	DCDV Model	R–THT Model	Cloud Users Count
500	98.4	94.5	95.7	500
1000	98.6	94.7	95.9	1000
1500	98.7	94.9	96.1	1500
2000	98.9	95.0	96.3	2000
2500	99.0	95.2	96.4	2500
3000	99.2	95.4	96.6	3000

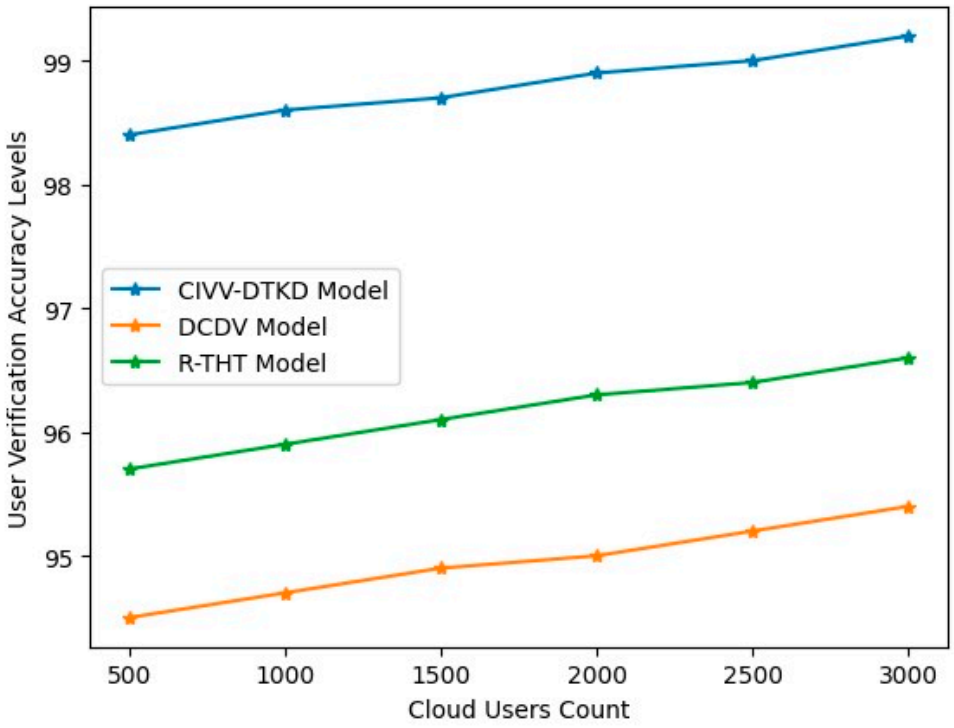

Figure 14. User Verification Accuracy Levels.

Table 11. Cloud Security Levels.

Cloud Users Count	CIVV–DTKD Model	DCDV Model	R–THT Model	Cloud Users Count
500	98.5	94.6	92.4	500
1000	98.7	94.8	92.6	1000
1500	98.9	95.0	92.7	1500
2000	99.0	95.2	92.9	2000
2500	99.2	95.4	93.0	2500
3000	99.4	95.6	93.2	3000

To strengthen the credibility of the reported results, the evaluation has been presented using **standard performance metrics** widely adopted in cloud integrity verification research. These include detection probability, false positive rate, computation overhead, communication overhead, and verification time. By employing these benchmarks, the performance of the CIVV–DTKD model can be objectively compared against existing schemes such as PDP, PoR, DCDV, and R-THT. In addition, the experimental setup, dataset size,

and attack scenarios considered during evaluation are explicitly described to ensure reproducibility. This structured presentation not only validates the claimed improvements in accuracy and efficiency but also demonstrates that the proposed scheme can reliably defend against realistic cloud security threats under practical conditions.

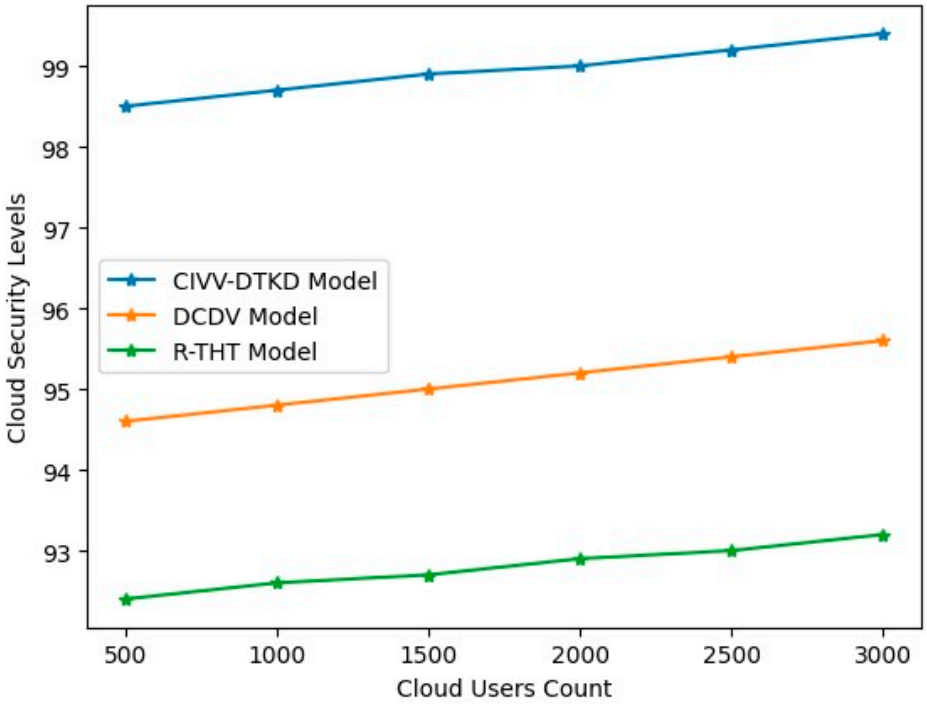

Figure 15. Cloud Security Levels.

While the results demonstrate the effectiveness of the proposed CIVV–DTKD model, it should be noted that the evaluation is currently expressed mainly in terms of "accuracy" and "security level." These measures provide an initial understanding of performance but are not standard benchmarks typically used for integrity verification schemes. To make comparisons more rigorous and widely accepted, future work will incorporate standard metrics such as detection probability, false positive rate, computational and communication overhead, and verification time. Using these established measures will allow the results to be more easily validated and compared against existing approaches like PDP, PoR, DCDV, and R-THT.

The proposed CIVV–DTKD model is more accurate and secure than existing methods, but these statements are not yet fully supported by formal proofs or a well-structured experimental setup. The current results, based mainly on "accuracy" and "security level," give a basic idea of the model's performance but do not meet the standard level of rigor in this research area. To make these claims stronger, future work will include formal security proofs and a clear experimental framework using standard metrics like detection probability, false positive rate, computation and communication overhead, and verification time. This will ensure that the improvements of the model are both formally proven and practically demonstrated.

5. Conclusions

Integrity validation is the basis of data security, and people are paying growing attention to it in a cloud storage environment. In order to use the conventional methods of integrity verification, the verifier must physically possess the whole verification object. However, traditional integrity verification methods suffer from significant limitations in

the cloud storage setting due to resource constraints, particularly those pertaining to the network. Cloud storage providers are not always reliable, so users risk having their files corrupted or deleted due to things like hardware failure, network attacks, or management mistakes. Here, the cloud storage provider might do whatever it wants. Interests play a role in the decision to hide or mislead users. As a result, while ensuring the security of files stored in the cloud, it is important to consider the potential for service providers to conduct intentional assaults. Using data mining and accountancy informatization as its foundation, this research builds a model for cloud data integrity verification, examines the data program flow's architecture, and delves into the tedious process of file data insertion. Cloud storage is an unavoidable part of the networked storage trend that will emerge as the primary storage method for the future, with the proliferation of mobile devices and the Internet. Cloud storage is going to be increasingly popular.

If the proposed CIVV–DTKD model can successfully lower both computational and communication overhead while also improving security, it would have considerable value for both the research community and industry. For researchers, it introduces a fresh framework that can be studied further and compared with other integrity verification techniques. For practitioners, it provides a practical, efficient, and deployable solution that strengthens trust in cloud services, safeguards sensitive data, and supports compliance with security requirements.

While customers like the convenience of storage, they give up some control over their files. There has been testing of the cloud server's security measures. When the need arises, data integrity certification ensures that users may continue to have control over their cloud files. With the rise in cloud storage, many researchers have taken an interest in the reliability of data certification in this setting, and integrity proof, in particular, has been a popular area of study. This research proposes a Cloud Integrity Verification and Validation Model using the Double Token Key Distribution model for enhancing cloud quality of service levels. The proposed model achieved 99.4% in ClVA Data Integrity Verification Accuracy Levels and 99.4% accuracy in providing security for cloud data and ClVA Data Integrity Verification Accuracy Levels.

The **CIVV–DTKD model** offers significant value for both application and research. On the practical side, its lightweight dual-token mechanism can be deployed in real cloud systems to ensure secure and efficient data integrity verification, making it highly useful for sectors like healthcare, finance, and government where trust and reliability are critical. On the **research side**, the model provides a versatile framework that can be further extended to incorporate advanced cryptographic techniques, multi-cloud scenarios, and stronger threat models, paving the way for future innovations in secure cloud computing.

This would benefit greatly from presenting results using widely accepted evaluation metrics in cloud integrity verification. Rather than focusing mainly on "accuracy" and "security levels," adopting measures like detection probability (the chance of detecting corrupted data), bandwidth costs (the communication required during verification), and computational overhead (the processing effort for users and servers) would make the results clearer and easier to compare with existing approaches. Applying these standard benchmarks would not only improve the strength of the analysis but also better demonstrate the practical value and efficiency of the proposed CIVV–DTKD model.

In the future, multi-level cloud platforms can be considered for integrity verification, and strong cryptography models can be considered to improve security levels.

Author Contributions: V.N.V.L.S.S.—original draft preparation, formula analysis, visualization, Investigation and funding acquisition. G.S.K.—Conceptualization, validation, Supervision, review and editing. A.V.V.—Methodology, Data curation, Supervision and Project administration. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: Dataset available on request from the authors.

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

References

1. Ni, J.; Zhang, K.; Yu, Y.; Yang, T. Identity-Based Provable Data Possession from RSA Assumption for Secure Cloud Storage. *IEEE Trans. Dependable Secur. Comput.* **2022**, *19*, 1753–1769. [CrossRef]
2. Zhang, Q.; Wang, S.; Zhang, D.; Wang, J.; Zhang, Y. Time and Attribute Based Dual Access Control and Data Integrity Verifiable Scheme in Cloud Computing Applications. *IEEE Access* **2019**, *7*, 137594–137607. [CrossRef]
3. Pinheiro, A.; Canedo, E.D.; De Sousa, R.T.; De Oliveira Albuquerque, R. Monitoring File Integrity Using Blockchain and Smart Contracts. *IEEE Access* **2020**, *8*, 198548–198579. [CrossRef]
4. Chattaraj, D.; Sarma, M.; Das, A.K.; Kumar, N.; Rodrigues, J.J.P.C.; Park, Y. HEAP: An Efficient and Fault-Tolerant Authentication and Key Exchange Protocol for Hadoop-Assisted Big Data Platform. *IEEE Access* **2018**, *6*, 75342–75382. [CrossRef]
5. Singh, A.; Ikuesan, R.A.; Venter, H. Secure Storage Model for Digital Forensic Readiness. *IEEE Access* **2022**, *10*, 19469–19480. [CrossRef]
6. Kim, T.-H.; Kumar, G.; Saha, R.; Rai, M.K.; Buchanan, W.J.; Thomas, R. A Privacy Preserving Distributed Ledger Framework for Global Human Resource Record Management: The Blockchain Aspect. *IEEE Access* **2020**, *8*, 96455–96467. [CrossRef]
7. Huang, H.; Wu, Y.; Xiao, F.; Malekian, R. An Efficient Signature Scheme Based on Mobile Edge Computing in the NDN-IoT Environment. *IEEE Trans. Comput. Soc. Syst.* **2021**, *8*, 1108–1120. [CrossRef]
8. Zhang, Y.; Geng, H.; Su, L.; Lu, L. A Blockchain-Based Efficient Data Integrity Verification Scheme in Multi-Cloud Storage. *IEEE Access* **2022**, *10*, 105920–105929. [CrossRef]
9. Zhao, X.-P.; Jiang, R. Distributed Machine Learning Oriented Data Integrity Verification Scheme in Cloud Computing Environment. *IEEE Access* **2020**, *8*, 26372–26384. [CrossRef]
10. Thangavel, M.; Varalakshmi, P. Enabling Ternary Hash Tree Based Integrity Verification for Secure Cloud Data Storage. *IEEE Trans. Knowl. Data Eng.* **2020**, *32*, 2351–2362. [CrossRef]
11. Peng, S.; Zhao, L.; Kumar, N. Comments on 'Efficient Public Verification of Data Integrity for Cloud Storage Systems From Indistinguishability Obfuscation'. *IEEE Trans. Inf. Forensics Secur.* **2021**, *16*, 4113–4116. [CrossRef]
12. Li, S.; Xu, C.; Zhang, Y.; Du, Y.; Yang, A.; Wen, X.; Chen, K. Backdoor-Resistant Public Data Integrity Verification Scheme Based on Smart Contracts. *IEEE Internet Things J.* **2023**, *10*, 14269–14284. [CrossRef]
13. Hu, A.; Jiang, R.; Bhargava, B. Identity-Preserving Public Integrity Checking with Dynamic Groups for Cloud Storage. *IEEE Trans. Serv. Comput.* **2021**, *14*, 1097–1110. [CrossRef]
14. Ali, M.; Sadeghi, M.-R.; Liu, X.; Vasilakos, A.V. Anonymous Aggregate Fine-Grained Cloud Data Verification System for Smart Health. *IEEE Trans. Cloud Comput.* **2023**, *11*, 2839–2855. [CrossRef]
15. Li, Y.; Li, Z.; Yang, B.; Ding, Y. Algebraic Signature-Based Public Data Integrity Batch Verification for Cloud-IoT. *IEEE Trans. Cloud Comput.* **2023**, *11*, 3184–3196. [CrossRef]
16. Liu, Z.; Ren, L.; Feng, Y.; Wang, S.; Wei, J. Data Integrity Audit Scheme Based on Quad Merkle Tree and Blockchain. *IEEE Access* **2023**, *11*, 59263–59273. [CrossRef]
17. Toutouh, J.; Muñoz, A.; Nesmachnow, S. Evolution oriented monitoring oriented to security properties for cloud applications. In Proceedings of the 13th International Conference on Availability, Reliability and Security, Hamburg, Germany, 27–30 August 2018; pp. 1–7.
18. Bian, G.; Chang, J. Certificateless provable data possession protocol for the multiple copies and clouds case. *IEEE Access* **2020**, *8*, 102958–102970. [CrossRef]
19. Levy-dit-Vehel, F.; Roméas, M. A framework for the design of secure and efficient proofs of retrievability. In Proceedings of the First International Conference on Cryptography, Codes and Cyber Security, Casablanca, Morocco, 27–28 October 2022; pp. 83–103.