VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology and Engineering (SITE)

B.Tech (Information Technology)

Course Project Report

Data Deduplication In Cloud

Submitted for the Course:
ITE 3007 : Cloud Computing and Virtualization

Offered by Dr. R. K. NADESH during FALL 2022 - 2023

By

Asmit	**20BIT0438**
Ashi Rathi	**20BIT0120**
Neeraj Kumar Choudhary	**20BIT0397**

November 2022

VIT
Vellore Institute of Technology
(Deemed to be University under section 3 of UGC Act, 1956)

School of Information Technology & Engineering
B.Tech (Information Technology)
FALL 2022 – 2023
ITE 3007 : Cloud Computing and Virtualization

A Report on the Course Project
Data Deduplication In Cloud

TEAM Name: CLOUDIFIED

Team Member(s) with Reg # and Name:
Asmit;20BIT0438;7909038931;asmit.2020@vitstudent.ac.in
Neeraj Kumar Choudhary;20BIT0397;9929231472;neerajkumar.choudhary2020@vitstudent.ac.in
Ashi Rathi;20BIT0120;9319140626;ashi.rathi2020@vitstudent.ac.in

Project Title: Data Deduplication In Cloud

1. Problem Statement

1.1 Background (System Study Details in brief)

The advent of the 20th century brought an immense increase in digital computation and, along with it, the start of an information era. This has further led to the development of research to ensure easily accessible data, which led to the introduction of cloud computing as a basic building block for the same. Cloud computing is a synthesis of numerous computing domains that allows for lower capital expenditures and greater flexibility in resource supply, scaling up, and scaling down. This is where cloud storage plays a major role in providing any kind of service. Because of benefits and characteristics such as multi-tenancy, improved server utilization, energy efficiency, and elasticity obtained from on-demand utility computing services, cloud computing use has increased tremendously. Data deduplication is a type of data compression technique that eliminates multiple copies of repeated data. It is also known as intelligent compression or single-instance storage. This approach is used to enhance storage utilization and can also be used to reduce the amount of bytes that must be delivered during network data transfers. During the deduplication process, unique blocks of data, or byte patterns, are discovered and stored. Deduplication techniques use data similarity to locate duplicate data and save storage space. Data deduplication is so closely related to incremental backup, which transfers just the data that has changed since the previous backup.

1.2 Problem Statement

Cloud storage is one of the cloud computing services that provides consumers with virtualized storage on demand. However, as the amount of data in the cloud grows, customers expect to be able to use on-demand cloud services at any time, while providers must maintain system availability and handle a significant amount of data. As a result, removing data redundancy has become one of the most important approaches in the cloud. Providers require a method to drastically reduce data quantities in order to save money while running large-scale systems. The cost of cloud hosting services can vastly increase due to the usage-based bandwidth pricing. The pay-as-you-go service model encourages cloud users to lower their bandwidth usage costs. Thus, the bandwidth cost has become a serious concern for the cloud application deployment and received a lot of attention. In order to reduce the bandwidth cost for data transfer from the cloud, Traffic Redundancy Elimination (TRE) technologies have been exploited to reduce the bandwidth usage by eliminating the transmission of duplicate information. Traffic Redundancy Elimination (TRE) has been proved to be an excellent method for lowering bandwidth costs, and as a result, it has recently received a lot of attention in the cloud environment. We discovered that both short-term (time span of seconds) and long-term (time span of hours or days) data redundancy can emerge in traffic, and that only employing sender-based TRE or receiver-based TRE cannot capture both types of traffic redundancy. Furthermore, the efficiency of present receiver-based TRE solutions is vulnerable to data changes when compared to the cached historical data. A solution for collaborative end-to-end TRE (CoRE) has been proposed.

1.3 Novelty

The development of a system that transmits data from the transmitter to the receiver. At the receiver, the TCP data stream that is entering is divided into pieces. The chunks are placed in a local chunk storage after being joined together in a chain. The receiver compares each arriving piece to the chunk storage. When a matching chunk is located on a chain, the following chunks are retrieved as predicted chunks for incoming data in the future. In a PRED message, the sender is informed of the signatures and anticipated offsets in the incoming data stream for the retrieved chunks as a forecast for the sender's impending outgoing data.

1.4 Dataset

Input values have been fed into the system considering different test cases and the results are analysed.

2. Related Works

2.1 Literature Survey

1) A Comparative Study of Data Deduplication Strategies by Nipun Chhabra and Manju Bala [1]:
In the given paper, the authors suggested deduplication as the strategy used to make efficient use of the space available for storing data in the cloud. The authors mention data de-duplication as the effective technique used to reduce the space occupied by reducing duplicates while maintaining data integrity. The paper presents various data deduplication techniques in reference to chunking (file-level chunking and block-level chunking), location (source and target), and time (inline and post-process). The choice of deduplication methodology depends on the needs of the user. The paper compares the different data deduplication strategies prevalent today. The paper concludes by mentioning the future work of the paper, which includes exploring different security techniques to secure data during deduplication and to execute deduplication in a more secure manner.

2) An Approach to Secure Capacity Optimization in Cloud Computing using Cryptographic Hash Function and Data De-duplication by Magesh Kumar S, Balasundaram A, Kothandaraman D, Auxilia Osvin Nancy V, P. J. Sathish Kumar, Ashokkumar S [2]:
The paper discussed the role of storage capacity optimization in the efficient use of cloud computing. The paper makes use of data deduplication as a strategy for efficient storage utilization. Redundant data is recognized by making use of hash functions which identify unique sequences (which are compared to the other sequences already calculated using the function). Thus, this allows the authors to reference the previously stored data rather than replicate it again. SHA3 calculation is used in the paper for hashing. When compared to the SHA-3 algorithm, MD5 takes up a lot of spare space. The MD5 hash capacity implementation is quite limited in terms of memory usage, time differentiation, and SHA-3 computation. SHA-3 aids in recovering significant plate space while also enhancing capabilities. SHA-3 is the most astounding

in recognising the richness of cloud records space.The work done in the paper can be used to reduce excess space taken up by the same data resources by a large percentage. The security of the data is also taken care of while evaluating the use of the hashing function in data deduplication. Thus, storage space is utilized effectively, hence reducing the cost of using the services.

3) Privacy-preserving and Updatable Block-level Data Deduplication in Cloud Storage Services by Hyungjune Shin, Dongyoung Koo, Youngjoo Shin, and Junbeom Hur [3]:

The proposed paper makes use of the Message-Locked Encryption technique to tackle the problem of performing secure data deduplication on encrypted data. This problem arises because of the difference in cipher texts obtained while using different algorithms and keys (here, the plain text/actual test remains the same whereas the cipher text differs). The paper also proposed many block-level deduplication schemes which are used to provide fine-grained storage by dividing data into blocks, wherein, even if some data changes in a particular file, only that data's index is changed in the index table. Thus, it ensures efficient use of storage. The paper also introduces the readers to the various attacks taking place on Message-Locked Encryption schemes when the message set becomes predictable. The performance of the proposed model is calculated and compared with UMLE. As a result, this study provides a novel and safe deduplication technique that guarantees successful updation of data and resistance to brute-force assaults, even when the messages obtained are predictable. The paper also assesses and indicates that increasing block size reduces bandwidth and time utilization. The system also ensures the acceptability and usefulness of the proposed model in real-world cloud scenarios.

4) RARE: Defeating Side Channels based on Data-Deduplication in Cloud Storage by Zahra Pooranian, Kang-Cheng Chen, Chia-Mu Yu, Mauro Conti [4]:

The given paper introduces the readers to the creation of side channels, which aids in providing the details of the file existence status to the attackers as a result of client-side data deduplication. The paper mentions that client-side data duplication is useful for reducing bandwidth and storage costs at the client-side but introduces vulnerability in the system, leading to intelligent attacks when the vulnerability is exploited by the attackers. The paper mentions the weakness of the contingency plans available before the paper was published and proposes a RAndom REsponse model (RARE) for achieving greater privacy and security of data, thereby reducing the number of vulnerabilities paving the way for attacks by hackers. The methodology of the RARE model involves the user sending two data chunks at once. The authors mention that the cloud that receives the deduplication request responds with a randomized deduplication response that has been carefully designed to preserve the deduplication advantage while minimizing privacy leakage. The analytical results of the given study corroborate the privacy guarantee, and the results show that both the deduplication benefit and the privacy of the RARE model can be preserved.

5) Dynamic Data Deduplication in Cloud Storage by Waraporn Leesakul, Paul Townend, Jie X [5]:

The given paper includes the role of cloud computing and the benefits provided by using cloud computing. The paper by the author discusses in length the cloud deployment models (software as a service, platform as a service, infrastructure as a service), types of clouds, and their benefits. The usage of cloud (private, public, hybrid, community) according to the need is also discussed in detail. It is emphasized that the cloud provides a platform for implementing big data analysis methodologies by different firms to enhance the utilization of data and the generation of useful information from the given unstructured datasets. The method of data deduplication is used in the paper, which aims at reducing the data storage by reducing duplication. The paper proposes a model which includes load balancer, deduplicators, cloud storage and redundancy manager for managing duplication in data and reducing redundancies. The given paper makes use of CloudSim and HDFS simulators, which work as simulating environments and are Java-based toolkits. Hash functions are used for deduplication and various sized files are used for conducting the experiments. The paper concludes by evaluating the percent reduction in size of the files and the time taken for performing deduplication. When the number of upload files is increased to a thousand, five and ten deduplicators can still aid to minimize processing time, but their effectiveness is reduced to 91.40% and 95.58%, respectively. However, time savings diminish dramatically when the number of upload files is increased to ten thousands, as five and ten duplicators may minimize 60.10% and 79.71% of processing time, respectively.

6) Randomized deduplication with ownership management and data sharing in cloud storage by Guohua Tiana, Hua Ma b, Ying Xie b, Zhenhua Liu:

The proposed paper uses a randomized client-side deduplication technique to tackle the problem of the security of data. The paper discussed the challenges proposed by the ever-increasing amount of data and the impact of the same on the storage capacity of the cloud environment, which therefore, makes it necessary to ensure that techniques like data deduplication are employed in order to reduce the amount of redundant data uploaded on the cloud. But, adoption of hashing methods can introduce some other attacks which result in threat to the data. Thus, the paper proposed methodology to tackle the proposed problem. The paper discussed the preliminaries, which include discrete logarithm problem, the lifted ElGamal encryption, hash collisions and static KEK tree. The problem formulation made in the paper included cloud storage system, the security model and the adversarial model and the security requirements. The proposed scheme of the paper included the main idea and the construction (system setup, data upload, ownership management, data encryption and data download). The security analysis employed by the paper included security proof, privacy, integrity, forward secrecy and backward secrecy, CAA and BFA resistance, and subsequently the paper discussed the comparisons and efficiency of the proposed model.

7) Survey on Data Deduplication in Cloud Storage Environments by Won-Bin Kim and Im-Yeong Lee [7]:

The paper discussed the importance of data deduplication strategy in improving the utilization of cloud storage and enhancing the amount of data that can be uploaded to the cloud. The paper makes use of comparisons to tackle the problem of security while making use of data deduplication as a strategy to reduce redundant data to improve data storage in the cloud. The paper classified data deduplication according to server-side, client-side and appliance side usages. Deduplication was classified according to different levels in the paper which included file-level deduplication, block-level deduplication(fixed length and variable-length). The paper also discussed the secure data deduplication methods and its importance in maintaining data security while reducing redundancies in data. Various attacks are also discussed in the paper, which included dictionary attack, poison attack, ownership forgery attack and thereafter various technologies vital for secure data deduplication (eg convergent encryption, oprf, mle data encryption, proof of ownership, preparing for poison attacks, proof of ownership and bloom filter) were discussed in depth. Technologies such as DupLESS, CloudDedup, PerfecDedup and Hur et al.’s scheme were also mentioned in the paper. The authors of this study explored ways for doing data deduplication as well as techniques for performing secure data deduplication. These technologies were not totally changed, but certain modified security technologies were adapted to the necessity for data deduplication. While these technologies have achieved their original goals, it is difficult to expect high levels of completeness while fulfilling both computation and traffic efficiency due to the advent of more security threats and the adoption of other security technologies to counter them. However, the recent desire for privacy and secure technologies is projected to enhance the demand and supply of safe data deduplication technologies in the future, leading to more advanced technology research.

8) Side channels in cloud services, the case of deduplication in cloud storage by Danny Harnik, Benny Pinkas and Alexandra Shulman-Peleg [8]:

The paper discussed the use of data deduplication as a strategy to reduce data duplication as a technique to reduce data redundancy while uploading user data to the cloud. This helps in saving cloud storage, hence improving space utilization. The research showed how deduplication can be used as a side channel to divulge information about the contents of other users' files. In another situation, as the paper discussed, deduplication can be used as a covert channel through which malicious software communicates with its command and control center, regardless of any settings, especially firewall settings on the attacking the system employed/used by the user.

Cloud storage providers are unlikely to abandon cross-user deduplication due to the significant savings it provides. As a result, we suggest simple strategies for enabling cross-user deduplication while significantly minimizing the danger of data leakage. 9) Application-Aware Big Data Deduplication in Cloud Environment by Yinjin Fu, Nong Xiao, Hong Jiang, Guyu Hu, and Weiwei Chen [9]:

The paper discussed in length the data deduplication techniques employed to tackle the problem of redundant data storage in the cloud. The authors discussed the different distributed data deduplication methodologies. The paper introduced the readers to the difference in application redundancy analysis paving way to the super-chunk resemblance analysis. The AppDEDUPE design (including the design principles: throughput, capacity and scalability) were discussed in depth. The overview of the AppDEDUPE included the clients (data partitioning, chunk fingerprinting, two-tiered data routing), director, (fire recipe management, application-aware routing decision) and the dedupe storage nodes (application-aware similarity index lookup, chunk fingerprint caching, parallel container management. The overview of the entire system included the clients, dedupe storage nodes and the director). The complete system introduced in the paper was evaluated based on the evaluation metrics, which included the deduplication efficiency, normalized deduplication ratio, normalized effective deduplication ratio, number of fingerprint index lookup messages, usage of RAM for inta-node data deduplication and data skew for distributed storage, which made use of MaxLoad, MinLoad and MeanLoad values for calculation of the performance. As a result, the authors conclude by stating that real-world trace-driven evaluation clearly revealed AppDedupe's significant advantages over current distributed deduplication algorithms for big clusters.

10) Secure Cloud Data Deduplication with Efficient Re-Encryption by Haoran Yuan, Xiaofeng Chen, Senior Member, Jin Li, Tao Jiang, Jianfeng Wang, and Robert H. Deng:

The authors developed a Bloom filter-based location selection approach and a safe data deduplication scheme in this research. The paper discussed the use of re-encryption methodology for efficient data deduplication. The preliminaries discussed in the paper include the convergent All-or-nothing transform, ciphertext-policy attribute-based encryption, and the bloom filter. The analysis of the reed scheme included the review and the sub-reserved attack. The models and security goals of the paper included the system model, threat model, and three folded goals, which ensure integrity of data. The scheme introduced in the paper included a bloom filter-based location selection mechanism and package generation. The analysis of the technique introduced in the paper is done by comparing different data deduplication techniques. The authors further demonstrated that the proposed approach achieves the desired security objectives and provided comprehensive simulation testing. The results of the experiments demonstrated that the technique was effective at re-encryption.

11) Heterogeneous Data Storage Management with Deduplication in Cloud Computing by Yan, Zheng; Zhang, Lifang; Ding, Wenxiu; Zheng, Qinghua [11]:

The paper discussed the storage management of heterogeneous data with data deduplication in a cloud computing context. The authors proposed a model for providing data deduplication along with simultaneous access control across various service providers of the cloud. The system and security model of the proposed paper included key generation centers, cloud service providers, authorized parties, data holders, proxy re-encryptions (PRE) for converting the cipher text to one that can be easily decrypted at a proxy and attribute based encryption. The system design included fundamental algorithms for system setup, data encryption and decryption, symmetric key management, partial key control (based on ABE), partial key control (based on PRE), the deduplication schemes, etc. The paper also evaluated the efficiency of the proposed model and in addition, compares it with pre-existing models. As stated in the study, as future work, the authors will strengthen user privacy and improve the performance of our approach in preparation for real deployment. Furthermore, the authors will perform game theory analysis to further demonstrate the rationality and security of the suggested scheme.

12) Private Data Deduplication Protocols in Cloud Storage by Wee Keong Ng, Yonggang Wen, Huafei Zhu [12]:

The paper introduced the notion of data deduplication for secure data storage. The authors demonstrated that the proposed private data deduplication protocol is safe if the underlying hash function is collision-resilient, the discrete logarithm is hard, and the erasure coding method can erase up to -a fraction of the bits in the presence of malevolent adversaries. The authors also provided correctness and sound proof of the proposed system in the paper. In this research, the authors introduced a novel concept known as private data deduplication protocols in the context of two-party computations. A possible outcome of private data deduplication procedures has been developed and assessed in the paper.

13) Privacy Preserving Data Deduplication in cloud using Advanced Encryption Standard by Dr. B. Tirapathi reddy, Maddireddy Vaishnavi, Makireddy Lalitha, Papineni Poojitha, Vakalapudi Bhavya Sri Kanthi:

The paper presented a system of secure data deduplication using advanced encryption cryptography standards. The paper presents a theoretical analysis of the AES system. The work proposed in the paper included user registrations, file uploading, encryption, checking for duplicates, and the final data dump diagram of the proposed model. The execution and assessment of encryption strategies resulted in the development of a more powerful encryption algorithm, whereas the deduplication strategy allowed the authors to save space in the cloud, cutting prices. Only clients who have been granted authorization will be able to encrypt and decrypt data using the keys generated during encryption. The paper experimented with and addressed the shortcomings of data encryption systems and performance.

14) A Verifiable Data Deduplication Scheme in Cloud Computing by Zhaocong Wen, Jinman Luo, Huajun Chen, Jiaxiao Meng, Xuan Li and Jin Li:

The paper presented data deduplication as the strategy for identifying and reducing redundancy in data. The paper discussed the method of validating image deduplication while storing data in the cloud. The authors' methodology in the study includes calculating the hash value of the image to be submitted by the user as its fingerprint. Following that, the fingerprint is transferred to cloud servers for validation and duplication detection. The methodology also involved the addition of a response mechanism by the storage and verification servers (no deduplication) and the subsequent transfer of data by the user to the servers. However, if the fingerprint is always discovered, the user will not upload the data for deduplication. This led to the conclusion that the server validation process was inefficient.

15) Data Deduplication in Cloud Computing Systems by Yingdan Shang, Huiba Li [15]:

The paper discussed data deduplication in a cloud computing system and in addition, defines cloud computing as a paradigm shift and data deduplication as a way of increasing storage efficiency in the cloud in addition to increasing the system overload. The paper introduced the users to the various challenges and proposed the solutions possible for those challenges. The preliminary study done in the paper on data-deduplication included live data deduplication in an open-source cloud framework, extreme binning, data deduplication backup mechanism, SAM framework (used for backing-up cloud, which reduces overload on the server-side by using excess CPU power and storage). The report examined various recent studies on applying data deduplication techniques to cloud systems and highlighted the inadequacies of previous work. Furthermore, the writers provided many viable solutions. Because most earlier deduplication work focused on centralized backup systems, the authors believed that their approach would pave the way for efficient deduplication for cloud computing environments.

16) Data Deduplication with Random Substitutions by Hao Lou, Farzad Farnoud [16]:

This paper provides an information-theoretic analysis of the performance of deduplication algorithms on data streams when repeated data segments are not necessarily exact copies. This paper introduce a source model in which probabilistic substitutions are considered and both the fixed-length scheme and the variable length scheme deduplication algorithms are studied. The fixed-length deduplication algorithm is shown to be unsuitable for the proposed source model as it does not take into account the edit probability while the conventional variable-length deduplication algorithm show that as source entropy becomes smaller, the size of the compressed string vanishes relative to the length of the uncompressed string, leading to high compression ratios. This paper takes the source model introduced by Niesen (the first person to perform an information-theoretic analysis of deduplication algorithms) as a reference which is incompatible with the current analysis.

17) Finding Data Deduplication using Cloud by Mr. M. A. R. Kumar, Mrs. Srilatha Puli[17]:

This paper focus on the attacks from malicious clients that are grounded on the manipulation of data identifiers and attacks based on backup time and network traffic observation. This paper defines intra-user deduplication, inter-user deduplication, client-side deduplication, server-side deduplication while describing the advantages, disadvantages, need and evolution of deduplication in today's explosive growth of data. The deduplication scheme provided in the paper is simple and robust and is efficient in terms of storage space and bandwidth savings for both clients and cloud service provider, the data deduplication is considered at a file level granularity but the solution can be extended to the block level. Two-phase deduplication that combines both intra- and inter-user deduplication techniques by introducing deduplication proxies between the clients and the storage server is the approach taken.

18) Similarity Based Deduplication with Small Data Chunks by L. Aronovicha, R. Asherb, D. Harnikb, M. Hirschb, S.T. Kleinc, Y. Toaf[18]:

This paper deals with the management of systems that are often compressed by means of deduplication techniques that partition the input text into chunks and store recurring chunks only once. It describes the design choices made during the development of an approximate hash function, serving as the basic tool of the new suggested deduplication system and report on extensive tests performed on a variety of large input files. The idea underlying a deduplication system is to locate repeated data and store only its first occurrence. The chunk size may indeed have a major impact on the performance: if it is too small, the number of different chunks may be so large as to jeopardize the whole approach, because the data structure D might not fit into RAM, so the system might not be scalable. On the other hand, if the chunk size is chosen too large, the probability of getting identical chunks decreases: many instances of chunks might exist, that could have been deduplicated had the chunk size been chosen smaller, but which, for the larger chunk size, have to be kept. A possible solution is to look for similar rather than identical chunks. If such a similar chunk is located, only the difference is recorded, which is generally much smaller than a full chunk. The idea of the current work is to implement the required similarity by what we call an approximate hash scheme. In this paper the first concern was to verify that the proposed approximate hash indeed spreads its values evenly. Once this has been confirmed, check that this uniformity does not come at the price of sensitivity, as it would for a standard hashing scheme. Thus checked the impact of the signature scheme in some artificial perturbation and clustering tests. Finally, bringing examples of applying the whole deduplication process in comparison with an identity based approach.

19) Data Deduplication Methods by Gagandeep Kaur, Mandeep Singh Devgan[19]:

This paper provides the concepts, methods and the schemes that can make the cloud services secure and reduce the incidence of data duplication. In this paper the proposed scheme works for deduplication of data with arithmetic key validity operations that reduce the overhead and increase the complexity of the keys so that it is hard to break the keys. This paper describe that without key management and key validation(components of the user management) process deduplication would remain unsecure process and file would always remain under multiple thread including integrity loss and breach of privacy and data duplication can be taken care of either by minimizing the number of writes for saving I/O bandwidth or denormalization. This paper shows that at each level of duplication process (file, block, chunk, zone) there is a needs of keys to be arithmetically valid and there ownership also need to be proved for proper working of any secure (Source, Target, Semantic ,Local, Hardware etc.). Deduplication system. From this study, it can also be concluded that there is no absolute or perfect solution of deduplication.

20) Secure Encrypted Data with Authorized Deduplication in Cloud by JINBO XIONG, YUANYUAN ZHANG, SHAOHUA TANG, XIMENG LIU and ZHIQIANG YAO[20]:

This paper propose a novel secure role re-encryption system (SRRS), which is based on convergent encryption and the role re-encryption algorithm to prevent the privacy data leakage in cloud and claims that it also achieves the authorized deduplication and satisfies the dynamic privilege updating and revoking. This paper discusses three major issues first, the adversary may utilize the relative attack methods to intercept the user's privacy information second, in the data deduplication based on the traditional convergent encryption the unauthorized users can obtain the user's information only by supplying the hash value of the file third, the privilege of the authorized user is dynamic and flexible, it is difficult to guarantee the access permission of authorized user and achieve the key updating and revoking management when performing data deduplication in response to which SRRS is introduced. In this proposed system, firstly exploited the convergent encryption algorithm to prevent privacy data leakage and used the role re-encryption algorithm to achieve authorized deduplication efficiently. Specifically, created a role authorized tree to manage the user's roles and the corresponding role keys, and introduced the management center to reduce the computation cost and management overhead of the client, and implement the dynamic updating of the authorized user's privilege.

21) Data Deduplication Strategies in Cloud Computing by MD. Jareena Begum, B. Haritha[21]:
This paper defines cloud computing, cloud advantages, types of cloud while listing data deduplication advantages and uses. This paper mention two strategies to deduplicate excess information - Inline and post-processing deduplication. This paper evaluates deduplication at three levels: File-level data deduplication strategy, Block level data deduplication strategy, Byte level data deduplication technology.

22) Security Analysis and Preserving Block-Level Data DE-duplication in Cloud Storage Services by M. Adithya, Dr. B. Shanthini[22]:
This paper examines the three-level cross-space design and propose an effective and protection safeguarding huge information deduplication in distributed storage which is referred to as EPCDD which accomplishes both protection safeguarding and information accessibility and opposes beast power assaults and beats existing contending plans, as far as calculation, correspondence, and capacity overheads. In this paper the process is mentioned to begin by registering a user in database who then uploads the data from system to cloud which is pre-processed and deduplicated and once uploaded to cloud then provides proof of storage that the data is stored in cloud.

23) Verifiable Attribute-Based Keyword Search Over Encrypted Cloud Data Supporting Data Deduplication by XUEYAN LIU, TINGTING LU, XIAOMEI HE, XIAOTAO YANG and SHUFEN NIU[23]:
This paper proposes a novel verifiable attribute-based keyword search over encrypted data supporting data deduplication to address the problems of data integrity detection, data deduplication and reasonable access authorization. This paper introduces a scheme to achieve effective access authorization and data confidentiality using attribute-based encryption. Searchable encryption is used to address the problem of the limited use of encrypted data. This paper details PRELIMINARIES AND SECURITY DEFINITIONS, SYSTEM AND SECURITY MODEL and the proposed system contains algorithms for system initialization, key generation, data encryption, trapdoor generation, search, result verification ,transform and decryption. The results of the experiment reveal that the scheme has a low band-width communication, storage and computability.

24) Research on Data Routing Strategy of Deduplication in Cloud Environment by QINLU HE, FAN ZHANG, GENQING BIAN, WEIQI ZHANG, DONGLI DUAN, ZHEN LI and CHEN CHEN[24]: This paper details a data routing strategy based on distributed Bloom Filter where to improve system throughput superchunk is used as the basic unit of data routing. The optimal node is selected as the routing node by matching the BloomFilter, and the storage capacity of the node and maintained in the memory of the storage node. In this paper the specific parameters of all kinds of routing strategies are obtained through experiments, and the routing strategies proposed are tested. The paper forwards with the cluster deduplication system as it has the advantage on improving storage space utilization and decreasing the reduplication rate, so the overall deduplication rate would decrease with the number of nodes in the cluster increases while the routing algorithm need to ensure the load balance of the system while keeping the deduplication

rate. One of the challenges in the paper is how to send data to the storage nodes. 25) Lightweight Cloud Storage Auditing with Deduplication Supporting Strong Privacy Protection by WENTING SHEN, YE SU and RONG HAO[25]:

The author proposes a cloud storage auditing scheme with deduplication supporting strong privacy protection that ensures that the privacy of the user's file would not be disclosed to the cloud and other parties when this user's file is predictable or from a small space. This paper provides a new method for duplicate check by generating a file index and use new strategy to generate a key for file encryption. The paper elaborates the issues like, when the file is predictable or from a small space, Convergent Encryption cannot resist brute-force dictionary attacks, in which the malicious cloud can recover the entire file with a number of guesses which leads us to a secure auditing and data deduplication scheme where a key server is introduced to help user generate the convergent key. But comes up with an issue where the key server is able to guess or derive the file's content from the file's hash value sent from the user by launching the brute-force dictionary attacks making it unable to fully prevent brute-force dictionary attacks. This paper focuses on solving this issue by introducing new strategies like generating file index with the help of Agency Server(AS) and the key for file encryption is generated with the file and the file label and the file label is kept by the user secretly and in order to improve the storage efficiency, the users, who own the same file, are able to generate the same ciphertext and the same authenticators.

26) A Secure Data Deduplication Scheme for Cloud Storage by Jan Stanek, Alessandro Sorniotti, Elli Androulaki, Lucas Kencel[26]:

This paper includes an encryption scheme that guarantees semantic security for unpopular data and provides weaker security and better storage and bandwidth benefits for popular data to increase the data deduplication efficiency. The paper proves the scheme is secure under the Symmetric External Decisional Diffie-Hellman Assumption in random oracle model.

27) Filter Based Data Deduplication in Cloud Storage using Dynamic Perfect Hash Functions by B. Tirapathi Reddy, M. V. P. Chandra Sekhara Rao[27]:

In this paper, the authors have leveraged the advantages of perfect hash functions and made use of a probabilistic data structure to ensure the ownership of the data items. The paper defines data in two ways as files uploaded/owned by several users and files uploaded/owned by very few users. Files belonging to the former group will be benefited strongly by deduplication as they are popular and may not be sensitive from the confidentiality perspective. Files owned by very few users may contain very sensitive information and require stronger security, but will not benefit a lot by deduplication which bring out a major challenge in secure design of the scheme - the secrecy in verifying the popularity of the data files. The proposed scheme consists of CSP to hold data and will verify the ownership of the data before which the file will be distributed in blocks and Key server which will maintain the list of all unpopular data items and their identifiers. As per this the user will have a bit of freedom on deciding the popularity of the data which is being uploaded. The popularity and the ownership of the data will then be checked from the input of the other users. The proposed mechanism dealt with the conflict of interest between storage optimization and security of the data in the cloud storage, while focus on providing varying levels of security for the data items based on how popular the data is.

28) A Review on Secure Data Deduplication: Cloud Storage Security Issue by Priteshkumar Prajapati, Parth Shah[28]:

This paper provides an overview of different secure deduplication techniques in cloud storage like Provable Data Possession (PDP), Proof of Retrievability (POR), secure keyword search, DupLESS, Proof of Storage with Deduplication (PoSD), Dekey, Message-Locked Encryption, Attribute Based Encryption (ABE) and Identity Based Encryption (IBE). This paper do a case study and mentions the security issues and threats analytically that the users are facing while also stating the various vulnerabilities that can be exploited. The paper concludes that there are certain issues which remain to be resolved, one of them is storage and management overhead caused by cryptographic keys and also the issue of breach of client's key. Moreover, in case of single user and multiple users (cross user) it's more difficult to the implement edit and deletion operations with secure data deduplication solution and efficiency can be increased by the combination of file-level and block-level deduplication.

29) A Secure and Efficient Data Deduplication Scheme with Dynamic Ownership Management in Cloud Computing by Xuewei Ma, Wenyuan Yang, Yuesheng Zhu, Zhiqiang Bai[29]:

This paper introduce a novel server-side deduplication scheme for encrypted data in a hybrid cloud architecture, where a public cloud manages the storage and a private cloud manages the role of the data owner to perform deduplication and dynamic ownership management and to reduce the communication overhead an initial uploader is used for mechanism check to ensure only the first uploader needs to perform encryption, and adopt an access control technique that verifies the validity of the data users before they download data. In the paper some of the issues are mentioned such as, to protect privacy, many users encrypt data before uploading it to the cloud storage where the encryption key is randomly generated, the same data encrypted with different keys will produce different ciphertext, which will hinder deduplication where the proposed solution is to revoke the cloud users from the valid ownership list. The proposed deduplication scheme works in the following way – the key is generated for the data and the private cloud gets the key, duplication check is performed by the public cloud, if the test is negative then the data is to be stored before which the private cloud encrypts the data, if the test is positive then the private cloud further clarifies the ownership of data by challenging hash code and then performs deduplication.

30) A secure data deduplication system for integrated cloud-edge networks by Shynu P. G., Nadesh R. K., Varun G. Menon, Venu P., Mahdi Abbasi and Mohammad R. Khosravi[30]:

This paper develops a new method using Convergent and Modified Elliptic Curve Cryptography (MECC) algorithms over the cloud and fog environment to construct secure deduplication systems focusing on two goals - the redundancy of data needs to be reduced to its minimum and to develop a robust encryption approach. In the methodology in the paper when the user uploads a file the file is being tested if it is duplication if the test is negative the file is encrypted and if the test is positive then the metadata of the already saved file is sent to the user whereas, in case of downloading a file the access to the file is checked. The proposed methodology is analyzed in four ways, i.e., a) when a new user tries to upload a new file, b) when the same user tries to upload the same file c) when different users try to upload the same file to the cloud server and d) when the users try to download the file. The methodology make use of convergent encryption and convergent decryption. The implemented deduplication methodology is deployed in the JAVA programming.

2.2 Comparative statement

S.no.	Topic and author	Proposal/Discussed
1	Data Deduplication with Random Substitutions by Hao Lou, Farzad Farnoud	An information-theoretic analysis of the performance of deduplication algorithms on data streams when repeated data segments are not necessarily exact copies
2	Lightweight Cloud Storage Auditing with Deduplication Supporting Strong Privacy Protection by WENTING SHEN, YE SU and RONG HAO	Cloud storage auditing scheme with deduplication supporting strong privacy protection
3	A Secure and Efficient Data Deduplication Scheme with Dynamic Ownership Management in Cloud Computing by Xuewei Ma, Wenyuan Yang, Yuesheng Zhu, Zhiqiang Bai	A novel server-side deduplication scheme for encrypted data in a hybrid cloud architecture
4	A secure data deduplication system for integrated cloud-edge networks by	A new method using MECC algorithms over the cloud environment to construct secure

	Shynu P. G., Nadesh R. K., Varun G. Menon, Venu P., Mahdi Abbasi and Mohammad R. Khosravi	deduplication systems focusing on reducing the redundancy of data to its minimum and develop a robust encryption approach.
5	An Approach to Secure Capacity Optimization in Cloud Computing using Cryptographic Hash Function and Data De-duplication by Magesh Kumar S, Balasundaram A, Kothandaraman D, Auxilia Osvin Nancy V, P. J. Sathish Kumar, Ashokkumar S.	The role of storage capacity optimization in the efficient use of cloud computing
6	Randomized deduplication with ownership management and data sharing in cloud storage by Guohua Tiana, Hua Ma b, Ying Xie b, Zhenhua Liu	A randomized client-side deduplication technique to tackle the problem of the security of data.
7	Survey on Data Deduplication in Cloud Storage Environments by Won-Bin Kim and Im-Yeong Lee	The importance of data deduplication strategy in improving the utilization of cloud storage and enhancing the amount of data that can be uploaded to the cloud
8	Privacy Preserving Data Deduplication in cloud using Advanced Encryption Standard by Dr. B. Tirapathi reddy, Maddireddy Vaishnavi, Makireddy Lalitha, Papineni Poojitha, Vakalapudi Bhavya Sri Kanthi	A system of secure data deduplication using advanced encryption cryptography standards.
9	Finding Data Deduplication using Cloud by Mr. M. A. R. Kumar, Mrs. Srilatha Puli	The attacks from malicious clients that are grounded on the manipulation of data identifiers and attacks based on backup time and network traffic observation.
10	Security Analysis and Preserving Block-Level Data DE-duplication in Cloud Storage Services by M. Adithya, Dr. B. Shanthini	Examines the three-level cross-space design and propose an effective and protection safeguarding huge information deduplication in distributed storage

2.3 Hardware Requirements

- System: Intel Core i3 or higher
- Hard Disk: 120 GB.
- Monitor: 15” LED
- Input Devices: Keyboard, Mouse
- Ram: 4 GB.

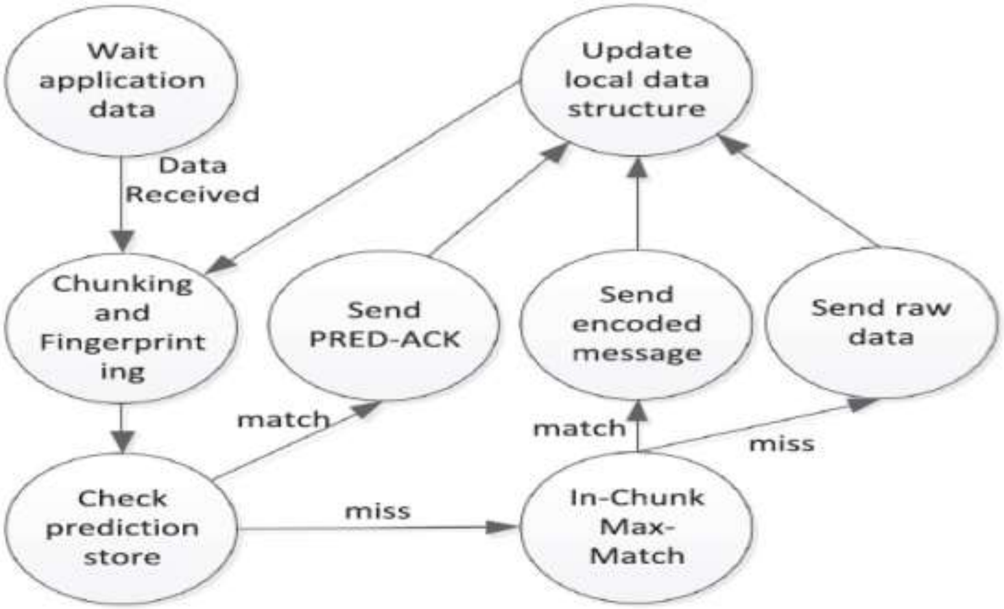
2.4 Software Requirements

- Operating system: Windows 7 or higher
- Coding Language: Java
- Tools: JAVA JDK1.8, Netbeans IDE 8.2

3. System Design

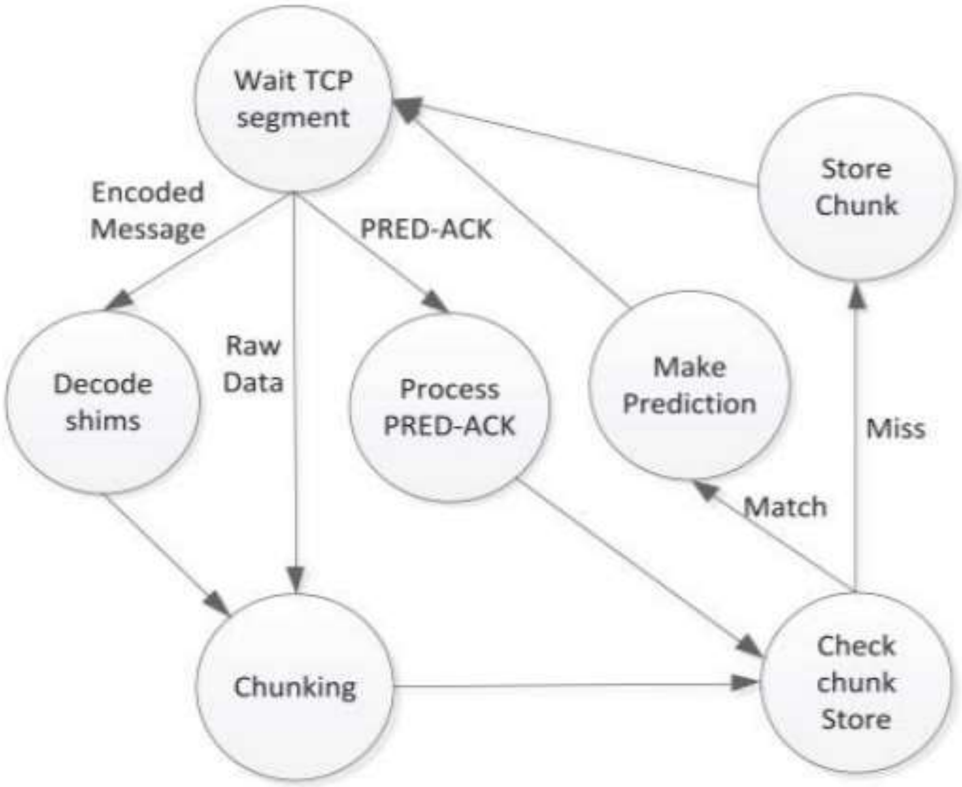
3.1 High-Level Design (Black Box design)

Sender Side



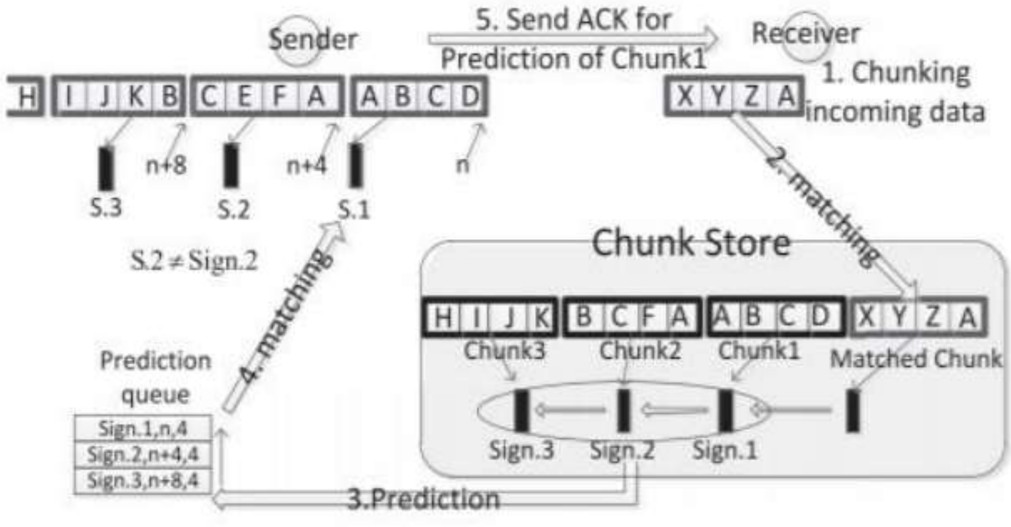
The sender employs a prediction store to cache the most recent predictions received from the receiver. Each prediction comprises the SHA-1 signature of a projected chunk as well as its expected offset, i.e., TCP sequence number in the TCP byte stream. The sender also has a chunk cache, which stores recently delivered chunks. A fingerprint store stores the meta-data of each representative fingerprint for each cached chunk, which contains the fingerprint value, the address of the chunk in the cache addressed by the fingerprint, and the byte offset of the window in the chunk across which the fingerprint is generated.

Receiver Side



The receiver separates the incoming data stream into pieces for each TCP connection and maintains a local prediction store that stores recent predictions for the TCP connection. To reconstitute the original data stream, the receiver sorts incoming TCP segments based on their kind. TCP segments from the sender to the receiver are classified into three types: PRED-ACK messages, shim-encoded messages, and bare data. The chunk predictions are sent in the order of their projected offsets within a PRED message.

3.2 Low-Level Design (Detailed design)



A system has been created that takes data from the sender and forwards it to the recipient. The entering TCP data stream is broken into pieces at the receiver. The chunks are linked in sequence to form a chain and are then stored in a local chunk storage. Each arriving chunk is compared to the chunk storage by the receiver. Once a matched chunk is found on a chain, it retrieves several consecutive chunks down the chain as projected chunks in future incoming data.

The recovered chunks' signatures and predicted offsets in the incoming data stream are provided to the sender in a PRED message as a forecast for the sender's upcoming outgoing data. To match data with a prediction, the sender computes SHA-1 across the outgoing data at the predicted offset with the length specified by the prediction, then compares the result to the signature in the prediction. When a signature match is detected, the sender sends a PRED-ACK message to the receiver, instructing it to transfer the matched data from its local storage. This approach eliminates the extra computational and storage expenses paid by TRE in the cloud, which would otherwise negate the bandwidth savings advantages.

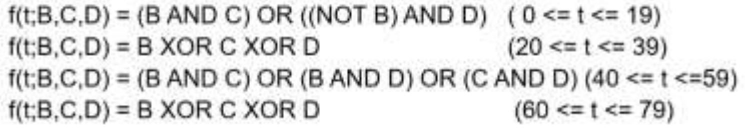
4. System Implementation

4.1 Algorithms (followed, proposed or altered)

SHA (Hashing Algorithm):
To match data with a prediction, the sender computes SHA-1 over the outgoing data at the expected offset with the length specified by the prediction, then compares the result to the signature in the prediction.
Steps:
Step 1: Append Padding Bits: The message is "padded" with a 1 and as many 0s as necessary to make it 64 bits shorter than an even multiple of 512.

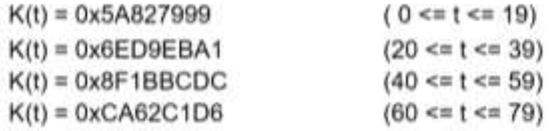
Step 2: Append Length: The padded message has 64 bits appended to the end. These bits contain the binary format of 64 bits, which represents the length of the original message.

Step 3: SHA1 necessitates 80 processing functions, which are as follows:



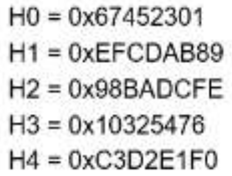
```
f(t;B,C,D) = (B AND C) OR ((NOT B) AND D)   ( 0 <= t <= 19)
f(t;B,C,D) = B XOR C XOR D                  (20 <= t <= 39)
f(t;B,C,D) = (B AND C) OR (B AND D) OR (C AND D) (40 <= t <=59)
f(t;B,C,D) = B XOR C XOR D                  (60 <= t <= 79)
```

Step 4: Prepare processing components: SHA1 necessitates 80 processing functions, which are as follows:



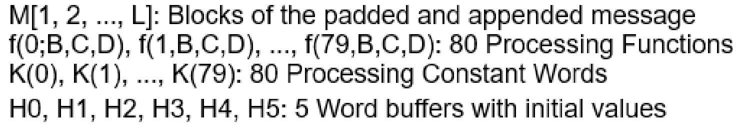
```
K(t) = 0x5A827999        ( 0 <= t <= 19)
K(t) = 0x6ED9EBA1        (20 <= t <= 39)
K(t) = 0x8F1BBCDC        (40 <= t <= 59)
K(t) = 0xCA62C1D6        (60 <= t <= 79)
```

Step 5: Initialize buffers:



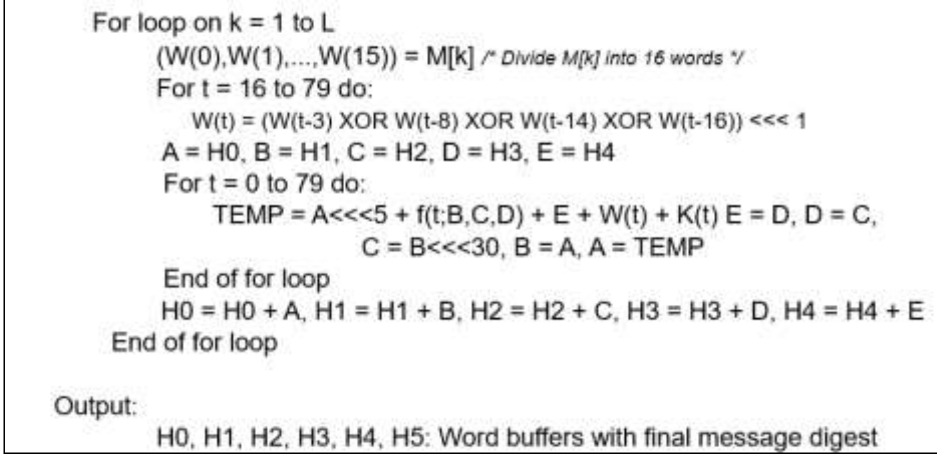
```
H0 = 0x67452301
H1 = 0xEFCDAB89
H2 = 0x98BADCFE
H3 = 0x10325476
H4 = 0xC3D2E1F0
```

Step 6: Processing message in 512-bit blocks (l blocks in total message): The SHA1 algorithm's main task is to loop through the padded and appended message in 512-bit blocks. Predefined functions and input:



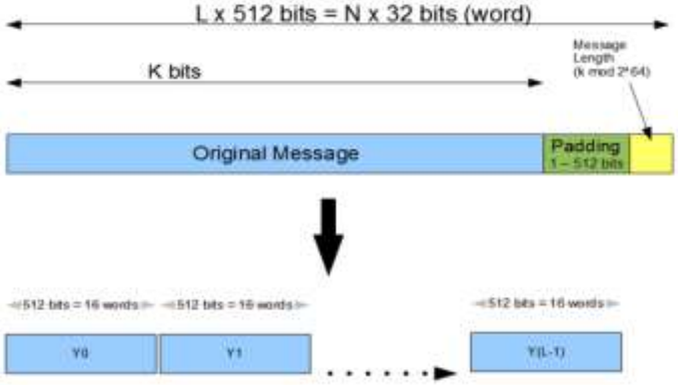
M[1, 2, ..., L]: Blocks of the padded and appended message
f(0;B,C,D), f(1,B,C,D), ..., f(79,B,C,D): 80 Processing Functions
K(0), K(1), ..., K(79): 80 Processing Constant Words
H0, H1, H2, H3, H4, H5: 5 Word buffers with initial values

Pseudo Code:



```
For loop on k = 1 to L
    (W(0),W(1),...,W(15)) = M[k] /* Divide M[k] into 16 words */
    For t = 16 to 79 do:
        W(t) = (W(t-3) XOR W(t-8) XOR W(t-14) XOR W(t-16)) <<< 1
    A = H0, B = H1, C = H2, D = H3, E = H4
    For t = 0 to 79 do:
        TEMP = A<<<5 + f(t;B,C,D) + E + W(t) + K(t) E = D, D = C,
                    C = B<<<30, B = A, A = TEMP
    End of for loop
    H0 = H0 + A, H1 = H1 + B, H2 = H2 + C, H3 = H3 + D, H4 = H4 + E
End of for loop

Output:
    H0, H1, H2, H3, H4, H5: Word buffers with final message digest
```



Cooperative End-to-End Traffic Redundancy Elimination (CoRE):
It ensures low computation and storage costs on servers and ensures that changes in cloud service points have little impact on TRE efficiency.

CoRE has two TRE modules: one for short-term redundancy and one for long-term redundancy. A two-layer redundancy detection system incorporates two TRE modules. The first-layer TRE module detects long-term redundancy in any outbound traffic from the server. If no short-term redundancy is found, it switches to the second-layer TRE module to look for it at a finer granularity.

First-layer TRE module (long-term data redundancy check): In this module, long-term redundancy is detected by using a prediction-based Chunk-Match method. In CoRE, the sender uses the same chunking algorithm as the receiver to divide incoming data into chunks. The sender then computes its signature (e.g., its SHA-1 hash value) for each chunk and looks up the signature in the prediction store, which stores all predictions recently received from the receiver. If a matching signature is found, the sender sends a prediction confirmation PRED-ACK message to the receiver instead of the outgoing chunk, regardless of whether the chunk has the predicted offset in the TCP stream. As a result, our prediction matching does not involve data position in the TCP stream, making CoRE resistant to data changes.

Second-layer TRE module (short-term data redundancy check):
Both the server and the client (present in this module) keep a temporary small local chunk cache to store the majority of recently transmitted and received chunks from their communication The server compares incoming data to the local data. To detect short-term redundancy, cache at fine granularity. The sender stores recently transferred chunks in its local cache. a chunk cache. The sender computes a value for each chunk in the cache. a collection of representative fingerprints, each containing the hash value in the chunk of a data window of size w Each and every representative fingerprint (along with a link to the corresponding chunk in the database). A fingerprint store is used to store cache. To detect redundancy within an outgoing chunk, the sender uses In-Chunk MaxMatch to identify the maximum number of sub-strings in the chunk that have duplicates in the chunk cache. The sender specifically compares each representative fingerprint of the outgoing chunk to the fingerprint store to see if a matching fingerprint exists. A matching fingerprint indicates that the outgoing chunk contains a data window of size w that was also present in a previously transmitted chunk in the cache. If a matching fingerprint is discovered in the fingerprint store, the in-cache chunk that it refers to is retrieved.

If any in-chunk shims are present, the receiver decodes them first. To decode an in-chunk shim, the receiver looks for a chunk with the same signature as the shim in its local cache and replaces the shim with the substring of the incache chunk based on the offset and length specified by the shim. If the packet is a PRED-ACK message, the receiver checks its prediction store for the confirmed prediction and retrieves the corresponding predicted chunk from its chunk store. The chunk is then copied to its TCP input buffer based on the offset in the TCP stream specified by the PRED-ACK message.

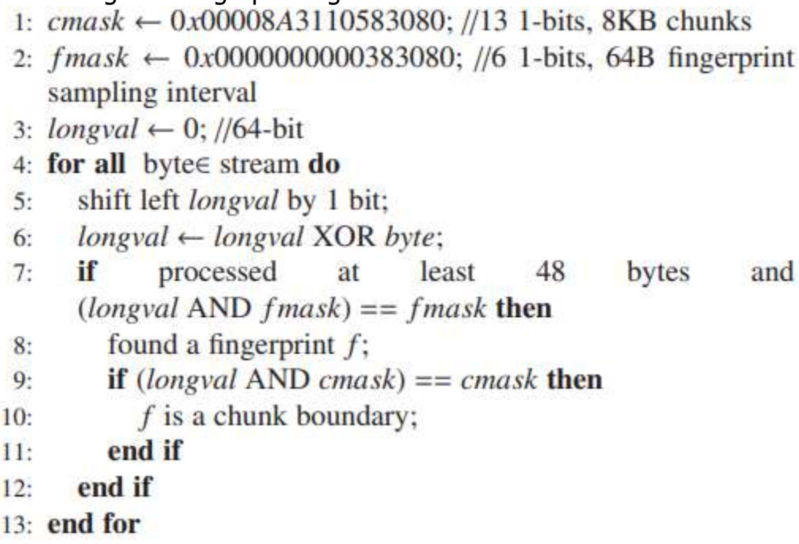
4.2 Mathematical Model (followed, proposed or altered)

CORE has 2 modules: for capturing short-term redundancy or long-term redundancy -

CoRE TREE:

If the 64-bit pseudo random hash has a '1' value at all k bit locations in a 64 bit string, denoted by Pf = b1, b2,..., bk, it is chosen as a fingerprint. Given a collection of n bit-places Pc in a 64-bit string such that |Pc| = n and Pf Pc, the fingerprint is picked as a chunk boundary if it has a '1' value at all positions in Pc. As a result, the average fingerprint sampling interval is 2k bytes, and the average chunk size is 2n bytes. The pseudo-code for our chunking and fingerprinting technique is shown in the given algorithm with w = 48, n = 13, and k = 6, resulting in an average fingerprint sampling interval of 64B and an average chunk size of 8KB.

~ Chunking and Fingerprinting ~



```
cmask ← 0x00008A3110583080; //13 1-bits, 8KB chunks
fmask ← 0x0000000000383080; //6 1-bits, 64B fingerprint
sampling interval
longval ← 0; //64-bit
for all byte∈ stream do
  shift left longval by 1 bit;
  longval ← longval XOR byte;
  if processed at least 48 bytes and
  (longval AND fmask) == fmask then
    found a fingerprint f;
    if (longval AND cmask) == cmask then
      f is a chunk boundary;
    end if
  end if
end for
```

The elapsed time of a prediction p in the store is defined in CoRE as:

$$E_p = \begin{cases} 0, & Seq_c \leq_{seq} Seq_p \\ (Seq_c - Seq_p) \bmod 2^{32}, & Seq_c >_{seq} Seq_p \end{cases}$$

where Seqc is the TCP sequence number of the chunk now being delivered and Seqp is the predicted TCP sequence number in the prediction. seqp and seqc are TCP sequence number modulo 232 comparisons. TTLpred is a system parameter that represents the maximum elapsed time of a prediction.

We suppose that the receiver discovers a matched chunk M in a chunk chain for an incoming TCP stream. The receiver then predicts using a succession of chunks following M in the chain, denoted by PM = pM 1, pM 2,..., pM k, where each pM I signifies a predicted chunk. S (pMi denotes the size of the chunk pMi. Assume TPM is the set of properly predicted chunks in PM. The hit ratio of prediction PM is then computed by:

$$H(P_M) = \frac{\sum_{p_i^M \in T_{P_M}} S(p_i^M)}{\sum_{i=1}^{k} S(p_i^M)}$$

4.3 Module Development –Code

1) CORECloudAdmin Module

~ Main.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package corecloudadmin;

import de.javasoft.plaf.synthetica.SyntheticaAluOxideLookAndFeel;
import javax.swing.UIManager;

/**
 *
 * @author admin
 */
public class Main {
    public static void main(String args[]) throws Exception
    {
        UIManager.setLookAndFeel(new SyntheticaAluOxideLookAndFeel());

        CloudAdminFrame csf=new CloudAdminFrame();
        csf.setVisible(true);
        csf.setResizable(false);
        csf.setTitle("Cloud Admin");

        CloudAdminReceiver csr=new CloudAdminReceiver(csf);
        csr.start();
    }
}
~ CloudAdminFrame.java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package corecloudadmin;

import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.Vector;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author admin
 */
public class CloudAdminReceiver extends Thread{

    CloudAdminFrame caf;

    CloudAdminReceiver(CloudAdminFrame af)
```

```
{
    caf=af;
}

public void run()
{
    try
    {
        DatagramSocket ds=new DatagramSocket(5000);
        while(true)
        {
            byte data[]=new byte[10000];
            DatagramPacket dp=new DatagramPacket(data,0,data.length);
            ds.receive(dp);
            String str=new String(dp.getData()).trim();
            String req[]=str.split("#");
            if(req[0].trim().equals("Connect"))
            {
                String serverId=req[1].trim();

                DefaultTableModel dm=(DefaultTableModel)caf.jTable1.getModel();
                Vector v=new Vector();
                v.add(serverId.trim());
                dm.addRow(v);
            }
            if(req[0].trim().equals("CloudServerDetails"))
            {
                String uid=req[1].trim();

                String sid="";
                for(int i=0;i<caf.jTable1.getRowCount();i++)
                {
                    String serverId=caf.jTable1.getValueAt(i,0).toString().trim();
                    sid=sid+serverId.trim()+",";
                }
                String allServerId="-";
                if(!(sid.trim().equals("")))
                {
                    allServerId=sid.substring(0,sid.lastIndexOf(','));
                    String msg="CloudServerDetails#"+allServerId.trim();
                    int pt=Integer.parseInt(uid.trim())+7000;
                    PacketTransmission(msg,pt);
                }
            }
        }
    }
    catch(Exception e)
    {
        e.printStackTrace();
        }
    }

    private void PacketTransmission(String msg, int pt) {
        try
        {
            byte data1[]=msg.getBytes();
            DatagramSocket ds1=new DatagramSocket();
            DatagramPacket dp1=new
DatagramPacket(data1,0,data1.length,InetAddress.getByName("127.0.0.1"),pt);
            ds1.send(dp1);
            System.out.println("Port is "+pt+"\n");
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }

}
```

2) CORECloudServer Module

Main.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package corecloudserver;

import de.javasoft.plaf.synthetica.SyntheticaAluOxideLookAndFeel;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.UIManager;

/**
 *
 * @author admin
 */
public class Main {
    public static void main(String args[]) throws Exception
    {
        UIManager.setLookAndFeel(new SyntheticaAluOxideLookAndFeel());

        String csid=JOptionPane.showInputDialog(new JFrame(), "Enter the Cloud Server Id: ");

        CloudServerFrame csf=new CloudServerFrame(Integer.parseInt(csid.trim()));
        csf.setVisible(true);
        csf.setResizable(false);
        csf.setTitle("Cloud Server - "+csid.trim());
        csf.jLabel1.setText("Cloud Server - "+csid.trim());

        CloudServerReceiver csr=new CloudServerReceiver(csf,Integer.parseInt(csid.trim()));
        csr.start();
    }
}
CloudServerReceiver.java
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package corecloudserver;

import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.net.InetAddress;
import java.util.ArrayList;
import java.util.Vector;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author admin
 */
public class CloudServerReceiver extends Thread{

    CloudServerFrame csf;
    int csid,port;
    int rowwid=0;
    public static ArrayList bandwidthSavings=new ArrayList();

    CloudServerReceiver(CloudServerFrame f,int id)
    {
        csf=f;
        csid=id;
        port=csid+6000;
    }

    public void run()
    {
        try
        {
            DatagramSocket ds=new DatagramSocket(port);
            while(true)
            {
                byte data[]=new byte[10000];
                DatagramPacket dp=new DatagramPacket(data,0,data.length);
```

```
ds.receive(dp);
String str=new String(dp.getData()).trim();
String req[]=str.split("#");
if(req[0].trim().equals("Connect"))
{
    String uid=req[1].trim();

    DefaultTableModel dm=(DefaultTableModel)csf.jTable1.getModel();
    Vector v=new Vector();
    v.add(uid.trim());
    dm.addRow(v);
}
if(req[0].trim().equals("Upload"))
{
    rowwid++;

    int savedbandwidth=0;

    String uid=req[1].trim();
    String allData=req[2].trim();
    String sp[]=allData.trim().split("\n");
    String uploadData="";
    for(int i=0;i<sp.length;i++)
    {
        String s[]=sp[i].trim().split("@");
        String chunkId=s[0].trim();
        String chunk=s[1].trim();
        String chuSig=s[2].trim();

        uploadData=uploadData+chunk.trim();

        DefaultTableModel dm=(DefaultTableModel)csf.jTable2.getModel();
        Vector v=new Vector();
        v.add(rowwid);
        v.add(chunkId.trim());
        v.add(chunk.trim());
        v.add(chuSig.trim());
        dm.addRow(v);
    }

    DefaultTableModel dm=(DefaultTableModel)csf.jTable3.getModel();
    Vector v=new Vector();
    v.add(rowwid);
    v.add(uid.trim());
    v.add(uploadData.trim());
    dm.addRow(v);


    JOptionPane.showMessageDialog(csf,"Uploaded Successfully!");
    bandwidthSavings.add(rowwid+"@"+savedbandwidth);
}
if(req[0].trim().equals("UploadWithMatched"))
{
    rowwid++;

    int savedbandwidth=0;

    String uid=req[1].trim();
    String allData=req[2].trim();
    String sp[]=allData.trim().split("\n");
    String uploadData="";
    for(int i=0;i<sp.length;i++)
    {
        if(sp[i].trim().contains("@"))
        {
            String s[]=sp[i].trim().split("@");
            String chunkId=s[0].trim();
            String chunk=s[1].trim();
            String chuSig=s[2].trim();

            uploadData=uploadData+chunk.trim();

            DefaultTableModel dm=(DefaultTableModel)csf.jTable2.getModel();
            Vector v=new Vector();
            v.add(rowwid);
            v.add(chunkId.trim());
            v.add(chunk.trim());
            v.add(chuSig.trim());
            dm.addRow(v);
        }
        if(sp[i].trim().contains("\t"))
        {
            String s[]=sp[i].trim().split("\t");
            String roId=s[0].trim();
            String chId=s[1].trim();

            for(int j=0;j<csf.jTable2.getRowCount();j++)
            {
                String rowId=csf.jTable2.getValueAt(j,0).toString().trim();
                String chunkId=csf.jTable2.getValueAt(j,1).toString().trim();
                String chunk=csf.jTable2.getValueAt(j,2).toString().trim();

                if((roId.trim().equals(rowId.trim()))&&(chId.trim().equals(chunkId.trim())))
                {
                    uploadData=uploadData+chunk.trim();
                    savedbandwidth=savedbandwidth+chunk.trim().length();
                    break;
                }
            }
        }
    }

    DefaultTableModel dm=(DefaultTableModel)csf.jTable3.getModel();
    Vector v=new Vector();
    v.add(rowwid);
    v.add(uid.trim());
    v.add(uploadData.trim());
    dm.addRow(v);

    JOptionPane.showMessageDialog(csf,"Uploaded Successfully!");

    bandwidthSavings.add(rowwid+"@"+savedbandwidth);

}
if(req[0].trim().equals("Predict"))
{
    String uid=req[1].trim();
    String firstChunkSignature=req[2].trim();

    String rowId="";
    int cou=0;
    for(int i=0;i<csf.jTable2.getRowCount();i++)
    {
        String chuSig=csf.jTable2.getValueAt(i,3).toString().trim();

        if(firstChunkSignature.trim().equals(chuSig.trim()))
        {
            cou=1;
            rowId=csf.jTable2.getValueAt(i,0).toString().trim();
        }
    }
    if(cou==0)      // Signature not matched
    {
        String msg="PredictionACK#"+"Signature not Matched";
        int pt=Integer.parseInt(uid.trim())+7000;
        JOptionPane.showMessageDialog(csf,"First Chunk Signature not Matched!");
        PacketTransmission(msg,pt);
    }
    else            // Signature Matched
    {
        JOptionPane.showMessageDialog(csf,"First Chunk Signature Matched!");

        for(int i=0;i<csf.jTable2.getRowCount();i++)
        {
            String rowId1=csf.jTable2.getValueAt(i,0).toString().trim();
```

```
                            if(rowId.trim().equals(rowId1.trim()))
                            {
                                String chuId=csf.jTable2.getValueAt(i,1).toString().trim();
                                String chuSig=csf.jTable2.getValueAt(i,3).toString().trim();

                                String 
msg="PredictionResults#"+rowId1.trim()+"#"+chuId.trim()+"#"+chuSig.trim();
                                int pt=Integer.parseInt(uid.trim())+7000;
                                PacketTransmission(msg,pt);
                            }
                        }
                    }
                }
            }
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }

    private void PacketTransmission(String msg, int pt) {
        try
        {
            byte data1[]=msg.getBytes();
            DatagramSocket ds1=new DatagramSocket();
            DatagramPacket dp1=new 
DatagramPacket(data1,0,data1.length,InetAddress.getByName("127.0.0.1"),pt);
            ds1.send(dp1);
            System.out.println("Port is "+pt+"\n");
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

3) CORECloudUser Module

Main.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package coreuser;

import de.javasoft.plaf.synthetica.SyntheticaAluOxideLookAndFeel;
import javax.swing.JFrame;
import javax.swing.JOptionPane;
import javax.swing.UIManager;

/**
 *
 * @author admin
 */
public class Main {
    public static void main(String args[]) throws Exception
    {
        UIManager.setLookAndFeel(new SyntheticaAluOxideLookAndFeel());

        String uid=JOptionPane.showInputDialog(new JFrame(), "Enter the User Id: ");

        UserFrame uf=new UserFrame(Integer.parseInt(uid.trim()));
        uf.setVisible(true);
        uf.setResizable(false);
        uf.setTitle("User - "+uid.trim());
        uf.jLabel1.setText("User - "+uid.trim());

        UserReceiver csr=new UserReceiver(uf,Integer.parseInt(uid.trim()));
        csr.start();
    }
}
```

UserReceiver.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package coreuser;

import java.net.DatagramPacket;
import java.net.DatagramSocket;
import java.util.Vector;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;

/**
 *
 * @author admin
 */
public class UserReceiver extends Thread{

    UserFrame uf;
    int userid,port;
    int tccount=0;
```

```
UserReceiver(UserFrame f,int id)
{
    uf=f;
    userid=id;
    port=userid+7000;
}

public void run()
{
    try
    {
        DatagramSocket ds=new DatagramSocket(port);
        while(true)
        {
            byte data[]=new byte[10000];
            DatagramPacket dp=new DatagramPacket(data,0,data.length);
            ds.receive(dp);
            String str=new String(dp.getData()).trim();
            String req[]=str.split("#");
            if(req[0].trim().equals("CloudServerDetails"))
            {
                String allServerId=req[1].trim();
                if(allServerId.trim().equals("-"))
                {
                    JOptionPane.showMessageDialog(uf,"Cloud Servers are Unavailable!");
                }
                else
                {
                    if(allServerId.trim().contains(","))
                    {
                        String sp[]=allServerId.trim().split(",");
                        for(int i=0;i<sp.length;i++)
                        {
                            String servId=sp[i].trim();

                            DefaultTableModel dm=(DefaultTableModel)uf.jTable1.getModel();
                            Vector v=new Vector();
                            v.add(servId.trim());
                            dm.addRow(v);

                            uf.jComboBox1.addItem(servId.trim());
                        }
                    }
                    else
                    {
                        DefaultTableModel dm=(DefaultTableModel)uf.jTable1.getModel();
                        Vector v=new Vector();
                        v.add(allServerId.trim());
                        dm.addRow(v);
                        uf.jComboBox1.addItem(allServerId.trim());
                    }
                }
            }
            if(req[0].trim().equals("PredictionACK"))
            {
                String res=req[1].trim();
                JOptionPane.showMessageDialog(uf,"Prediction Result Received Successfully!");
                JOptionPane.showMessageDialog(uf,res.trim());
                JOptionPane.showMessageDialog(uf,"No alternate way! You should sent all chunks!");
            }
            if(req[0].trim().equals("PredictionResults"))
            {
                String rowId=req[1].trim();
                String chunkId=req[2].trim();
                String chunkSig=req[3].trim();

                DefaultTableModel dm=(DefaultTableModel)uf.jTable2.getModel();
                Vector v=new Vector();
                v.add(chunkSig.trim());
                v.add(chunkId.trim());
                v.add(rowId.trim());
                dm.addRow(v);

                if(tccount==5)
                {
                    tccount--;

                    DefaultTableModel dm2=(DefaultTableModel)uf.jTable3.getModel();
                    dm2.removeRow(0);

                    DefaultTableModel dm1=(DefaultTableModel)uf.jTable3.getModel();
                    Vector v1=new Vector();
                    v1.add(chunkSig.trim());
                    v1.add(chunkId.trim());
                    v1.add(rowId.trim());
                    dm1.addRow(v1);
                    tccount++;
                }
                else
                {
                    DefaultTableModel dm1=(DefaultTableModel)uf.jTable3.getModel();
                    Vector v1=new Vector();
                    v1.add(chunkSig.trim());
                    v1.add(chunkId.trim());
                    v1.add(rowId.trim());
                    dm1.addRow(v1);
                    tccount++;
```

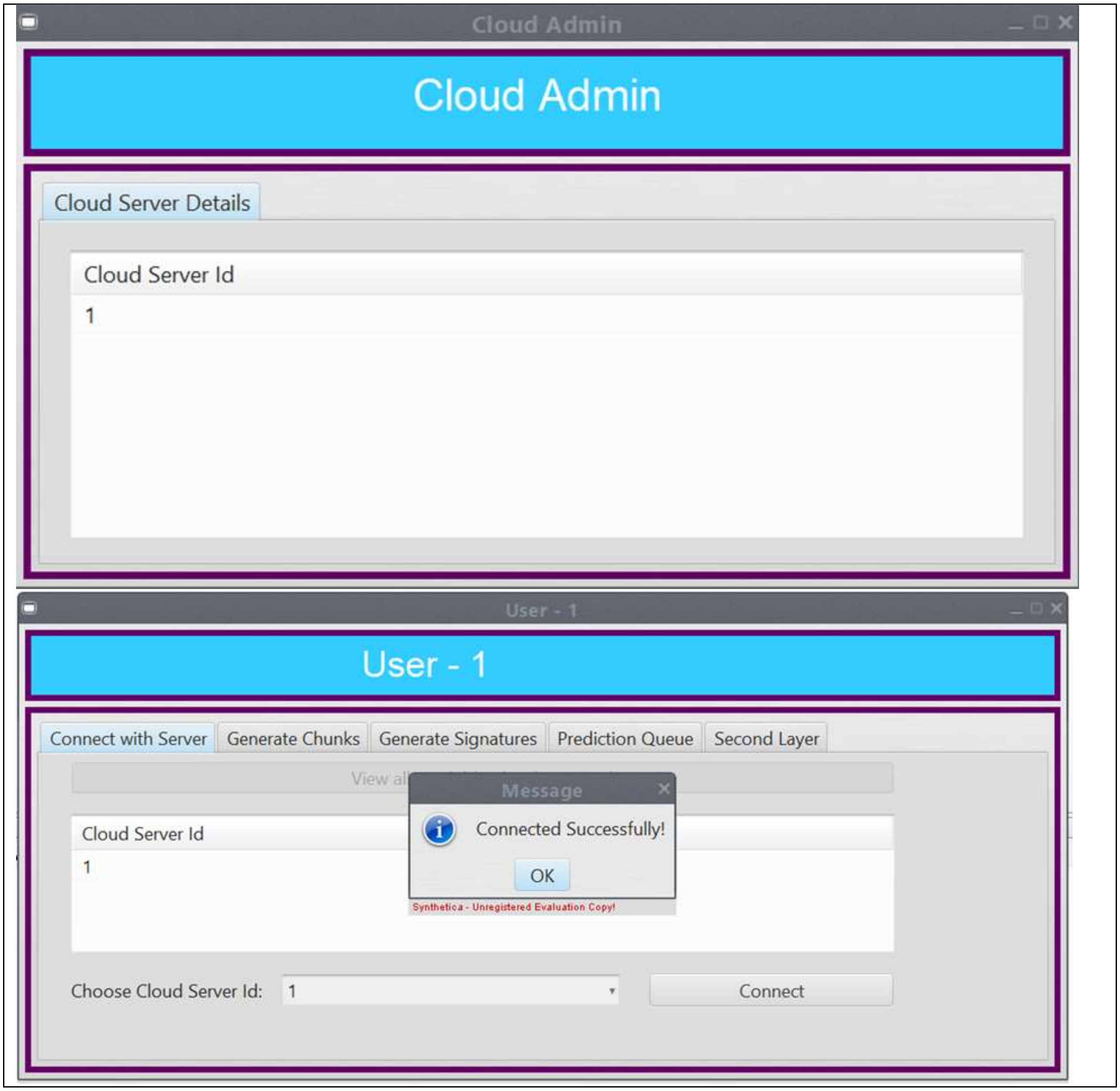
```
                    }
                }
            }
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }

}
```

4.4 Test Cases

Test cases:

1) ABCDEFGHIJKL

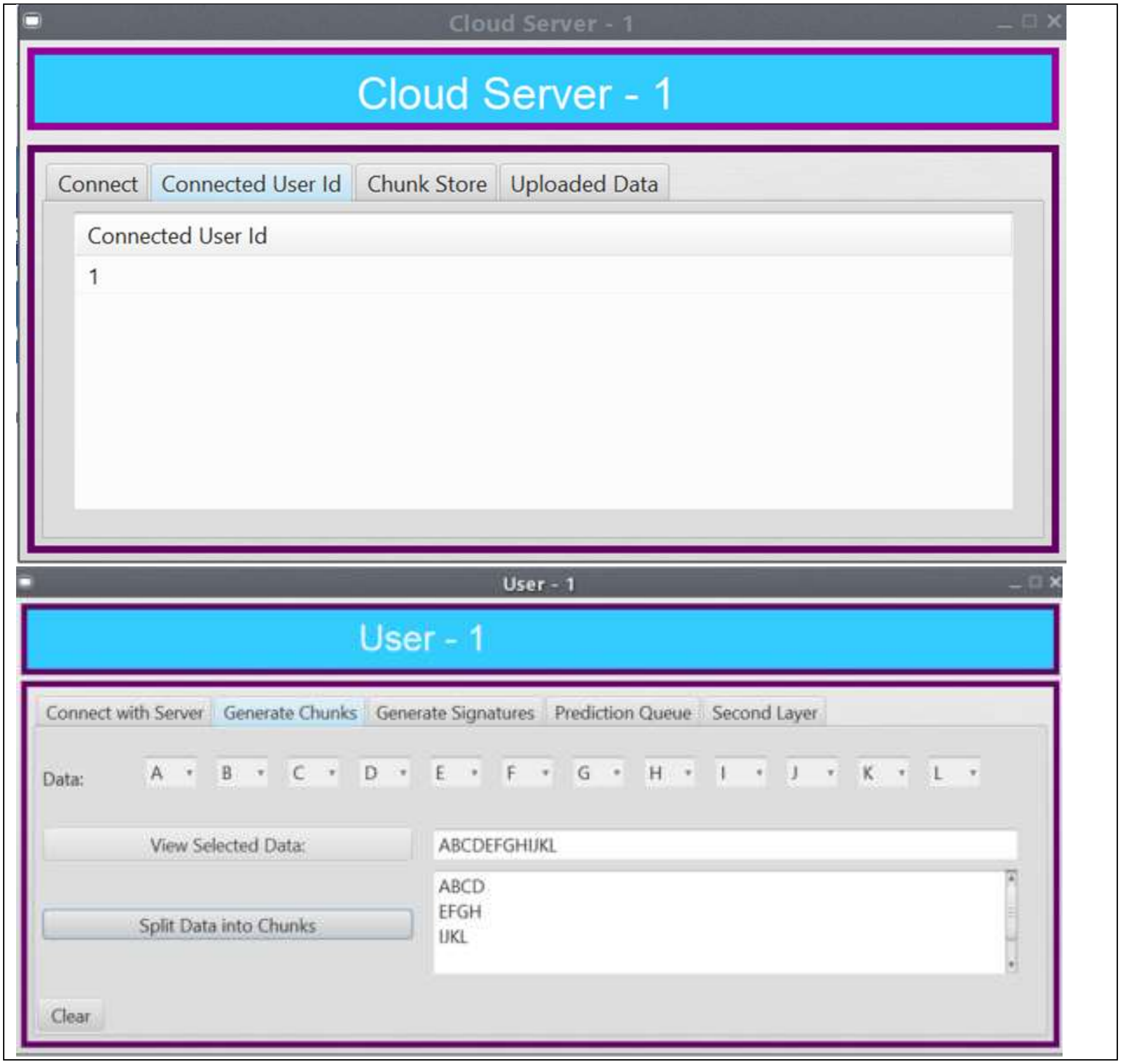


Cloud Server Id
1

Cloud Server Id
1

Connected Successfully!

Choose Cloud Server Id: 1

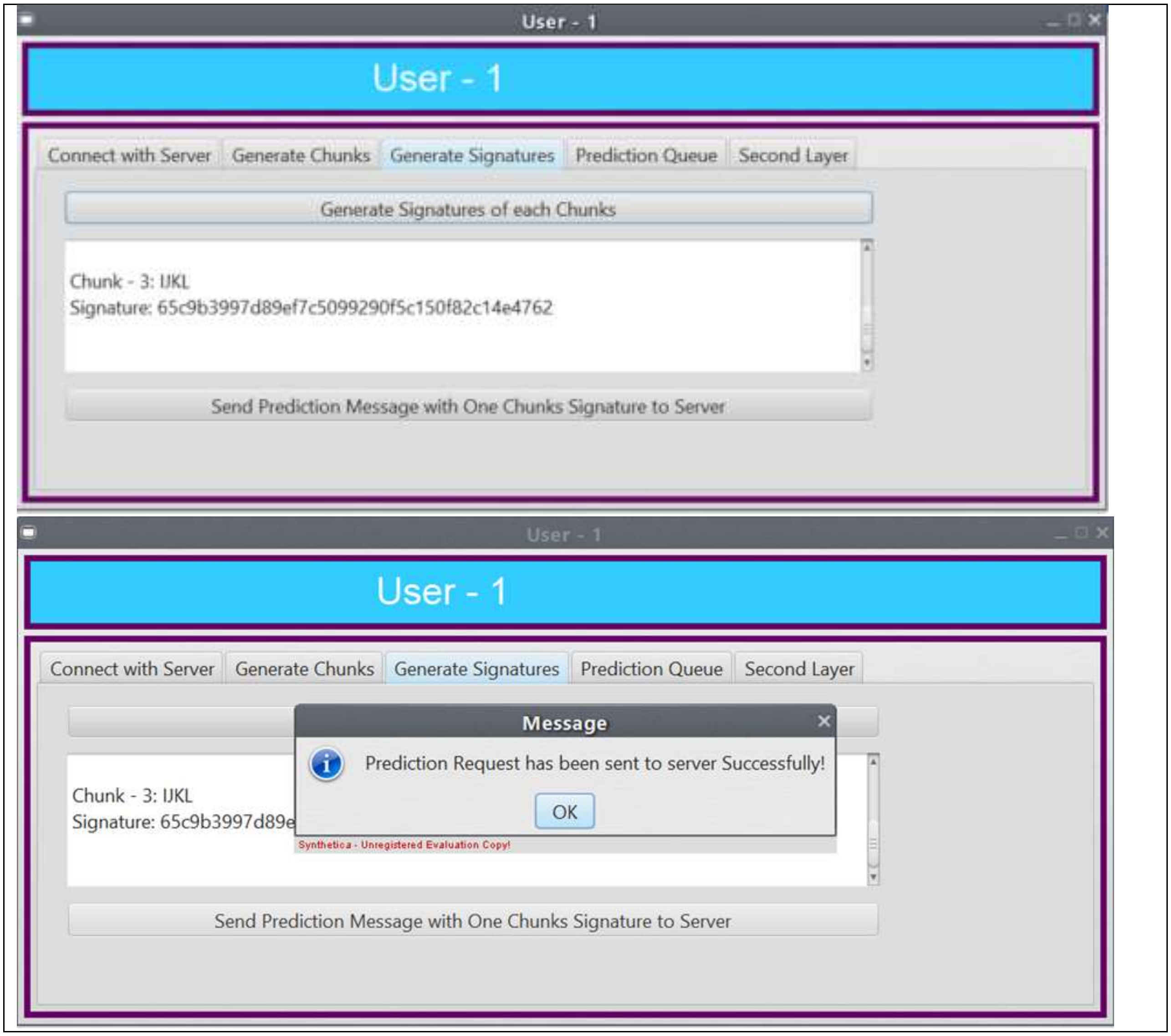


Connected User Id
1

Data: A B C D E F G H I J K L

View Selected Data: ABCDEFGHIJKL

ABCD
EFGH
IJKL

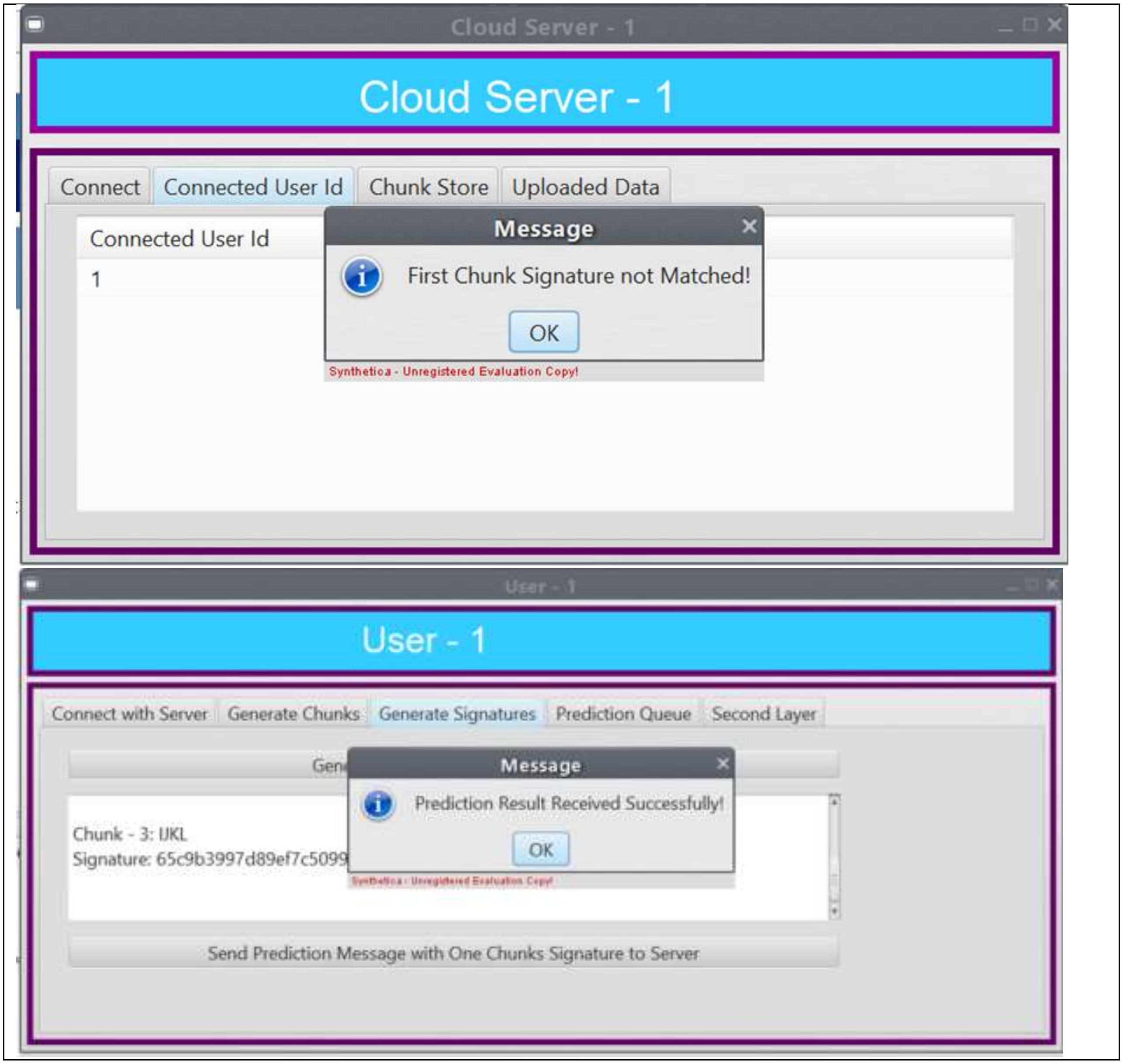


Chunk - 3: IJKL
Signature: 65c9b3997d89ef7c5099290f5c150f82c14e4762

Chunk - 3: IJKL
Signature: 65c9b3997d89e

Prediction Request has been sent to server Successfully!

OK
Synthetica - Unregistered Evaluation Copy!
Send Prediction Message with One Chunks Signature to Server

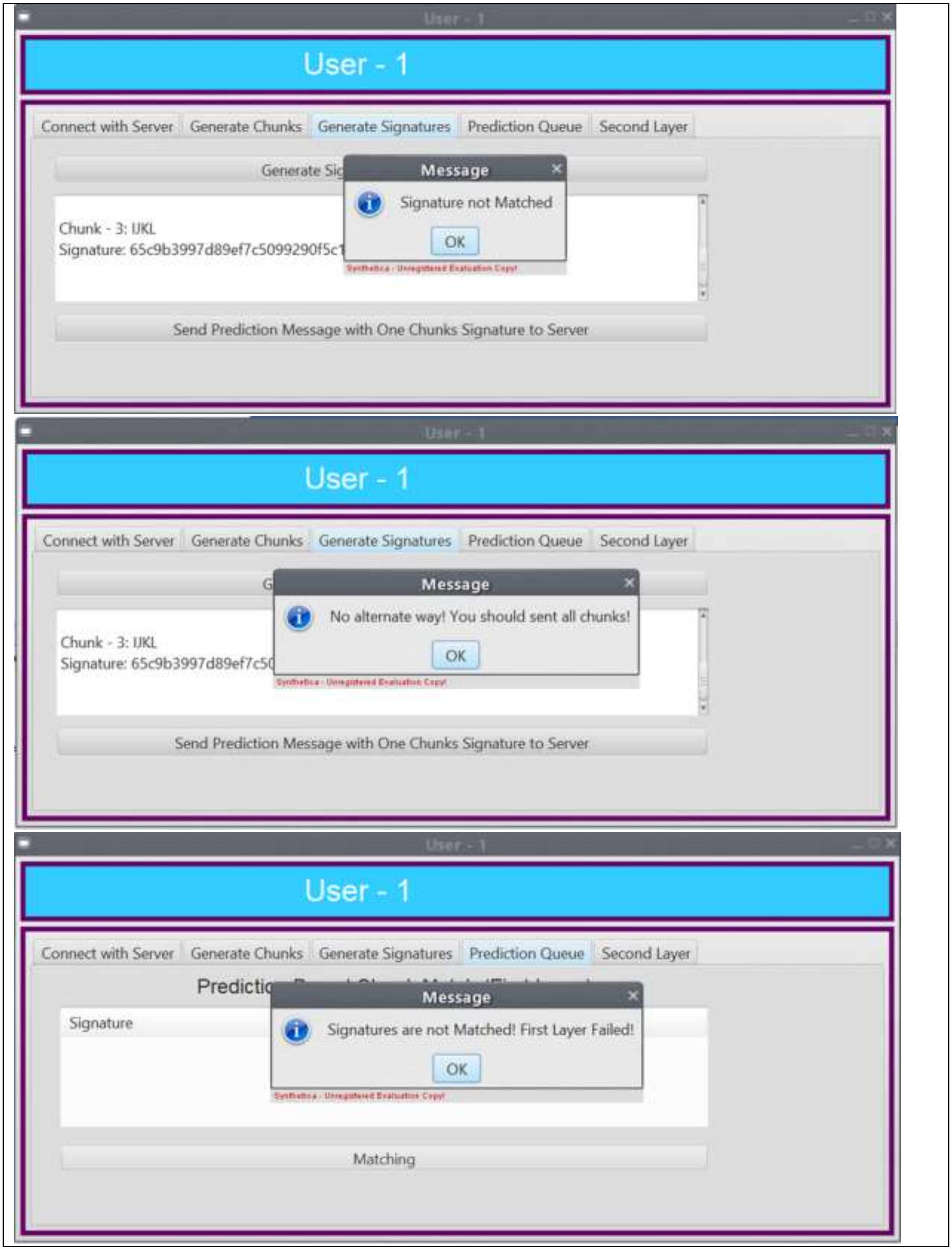


Connected User Id
1

First Chunk Signature not Matched!

Prediction Result Received Successfully!

Chunk - 3: IJKL
Signature: 65c9b3997d89ef7c5099



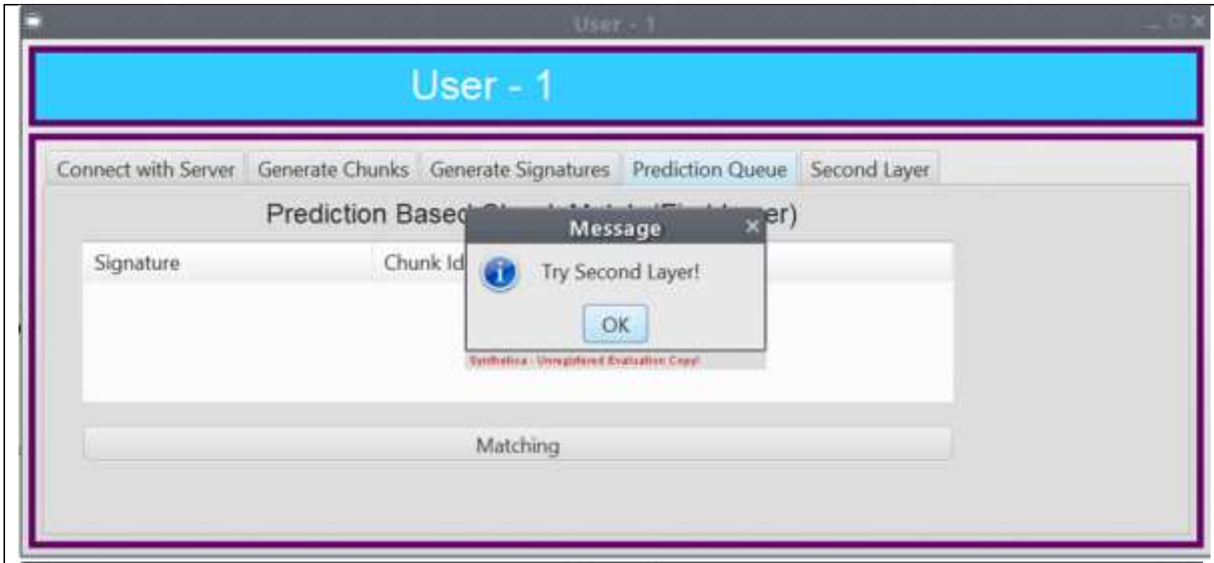
Chunk - 3: IJKL
Signature: 65c9b3997d89ef7c5099290f5c1

Message: Signature not Matched

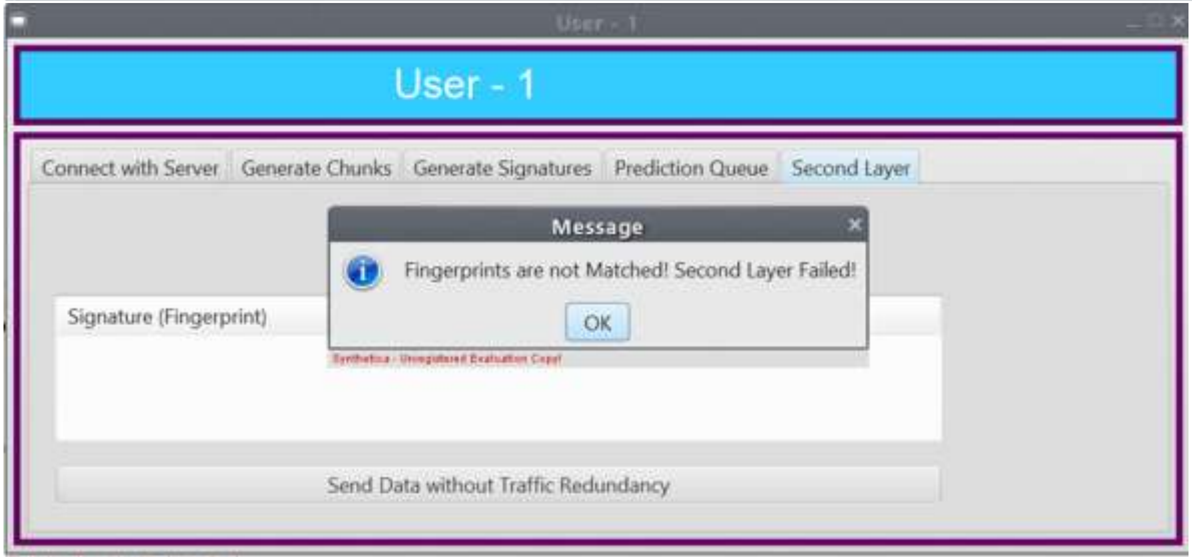
Chunk - 3: IJKL
Signature: 65c9b3997d89ef7c5C

Message: No alternate way! You should sent all chunks!

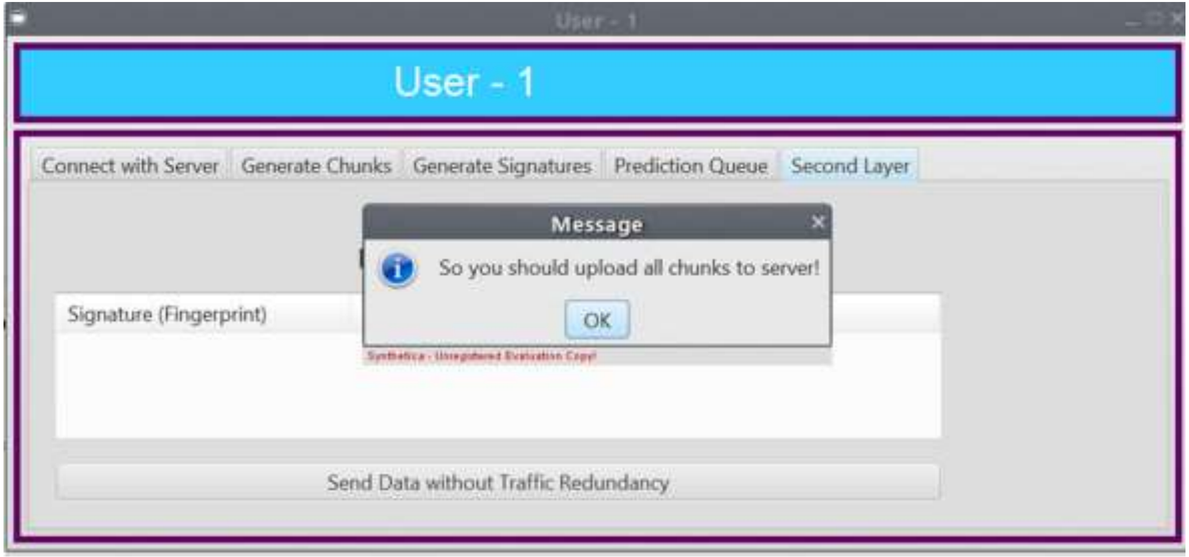
Message: Signatures are not Matched! First Layer Failed!



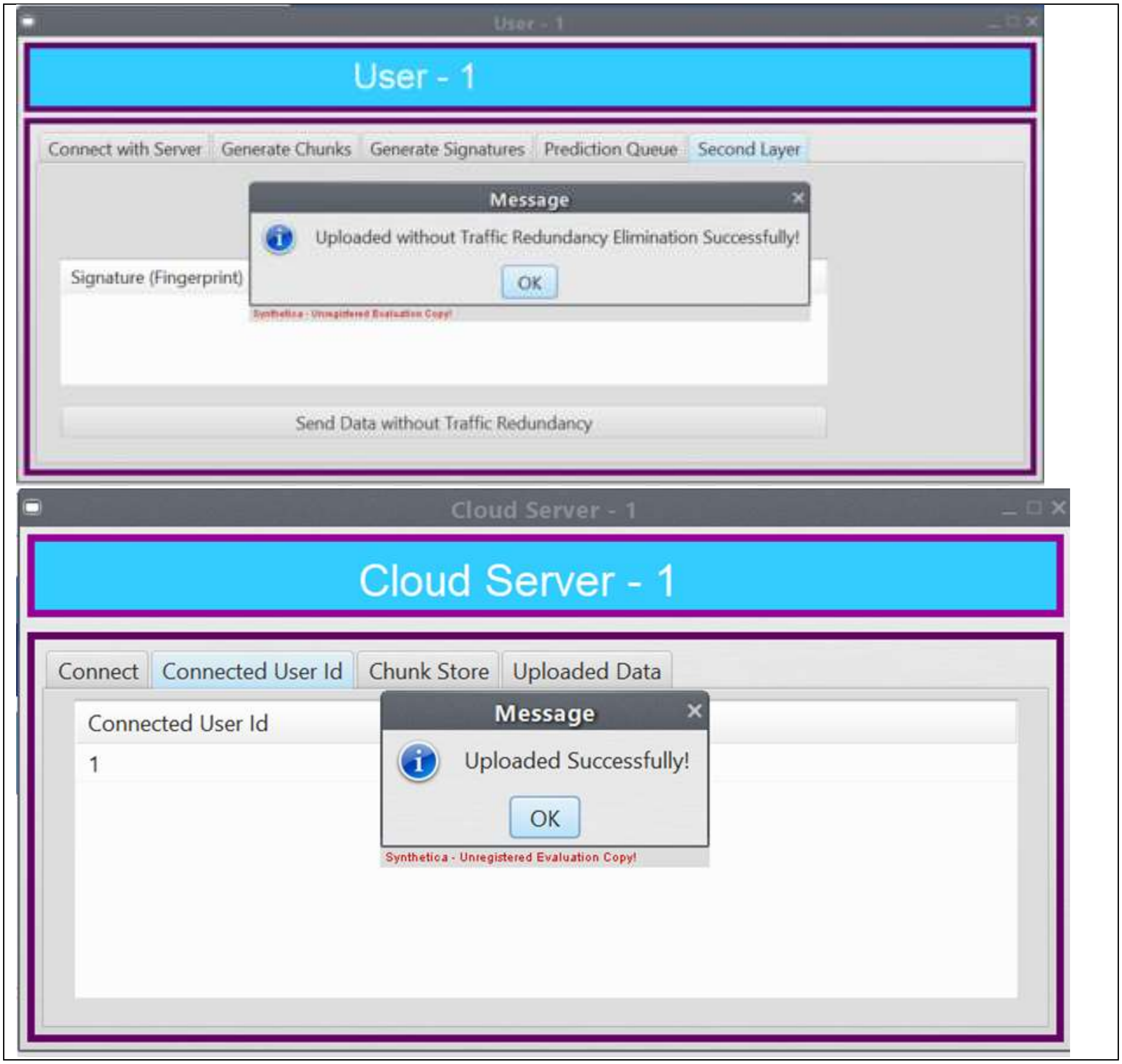
Try Second Layer!



Fingerprints are not Matched! Second Layer Failed!



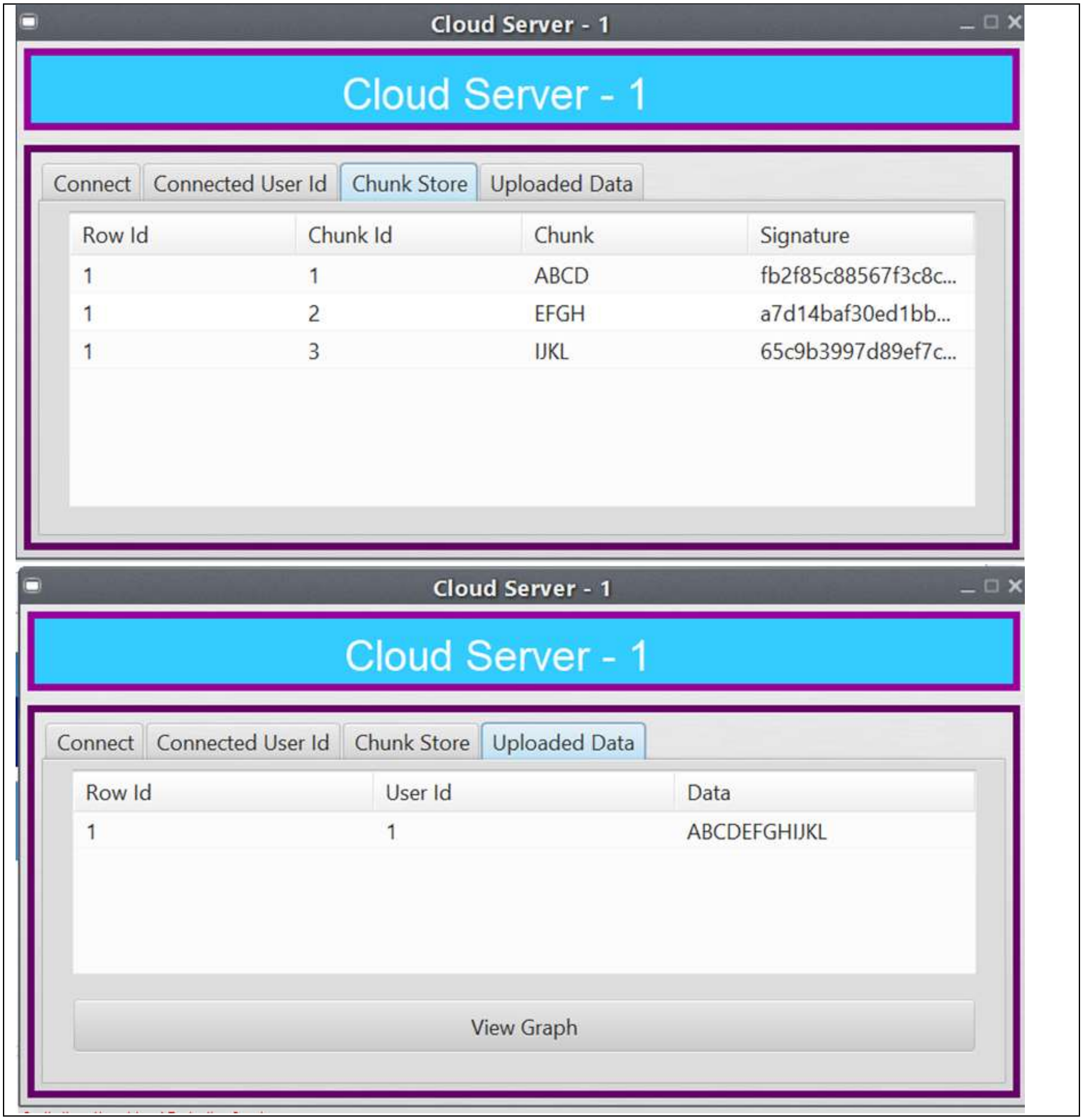
So you should upload all chunks to server!



Uploaded without Traffic Redundancy Elimination Successfully!

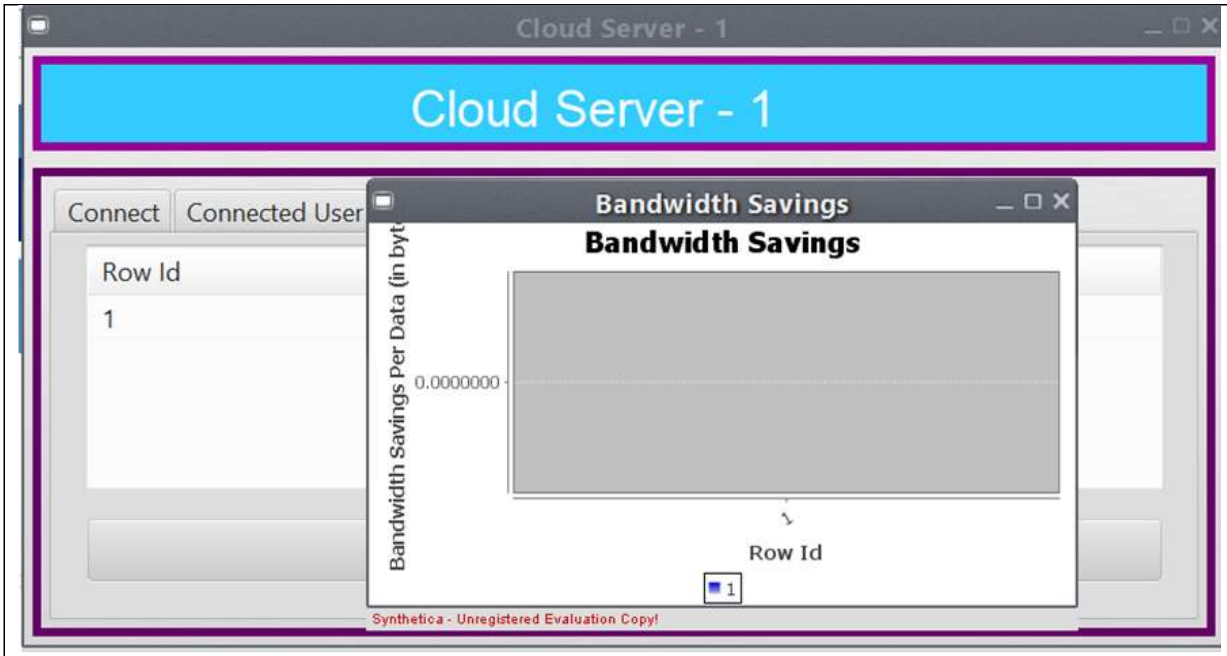
Connected User Id
1

Uploaded Successfully!

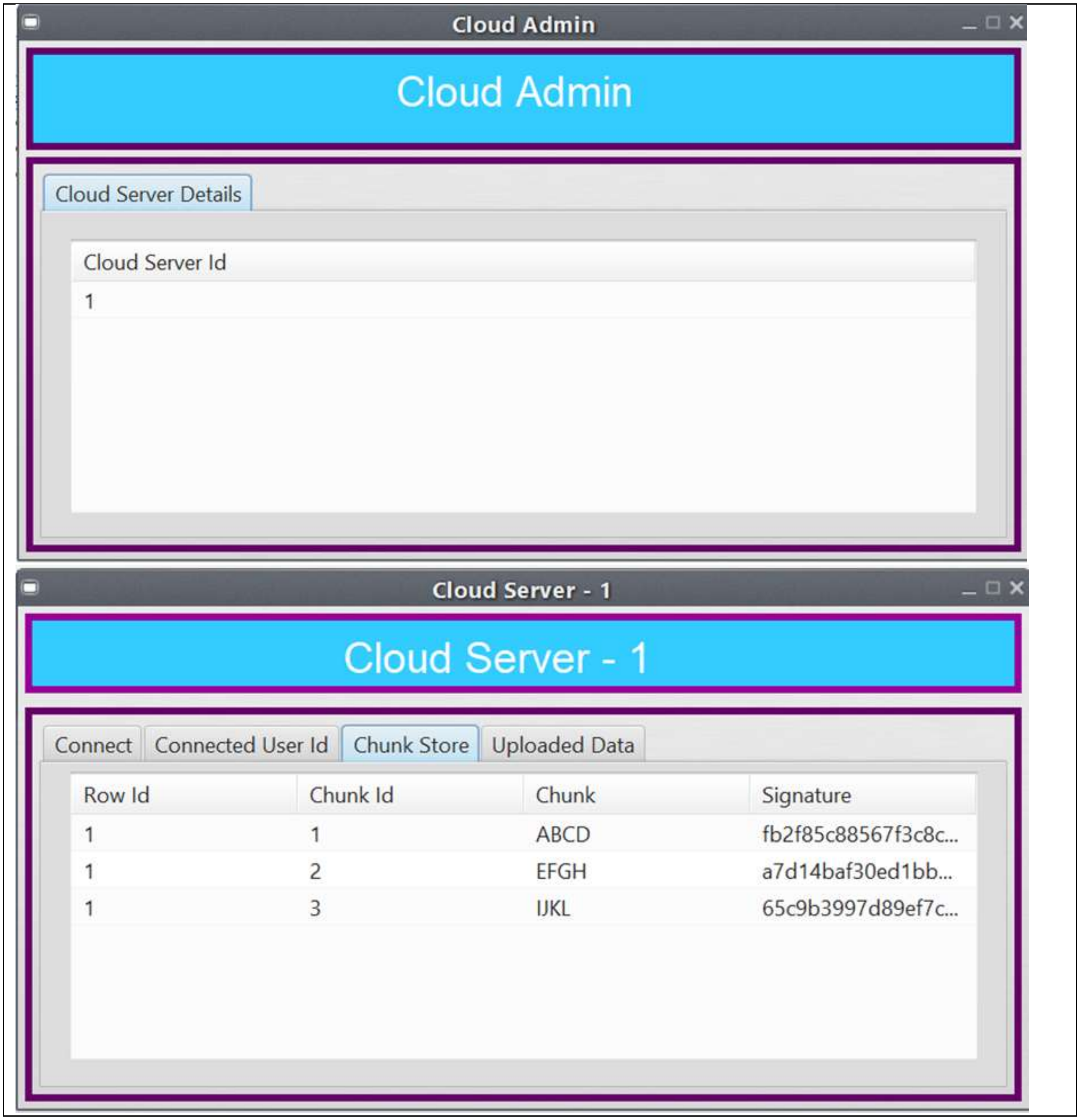


Row Id	Chunk Id	Chunk	Signature
1	1	ABCD	fb2f85c88567f3c8c...
1	2	EFGH	a7d14baf30ed1bb...
1	3	IJKL	65c9b3997d89ef7c...

Row Id	User Id	Data
1	1	ABCDEFGHIJKL

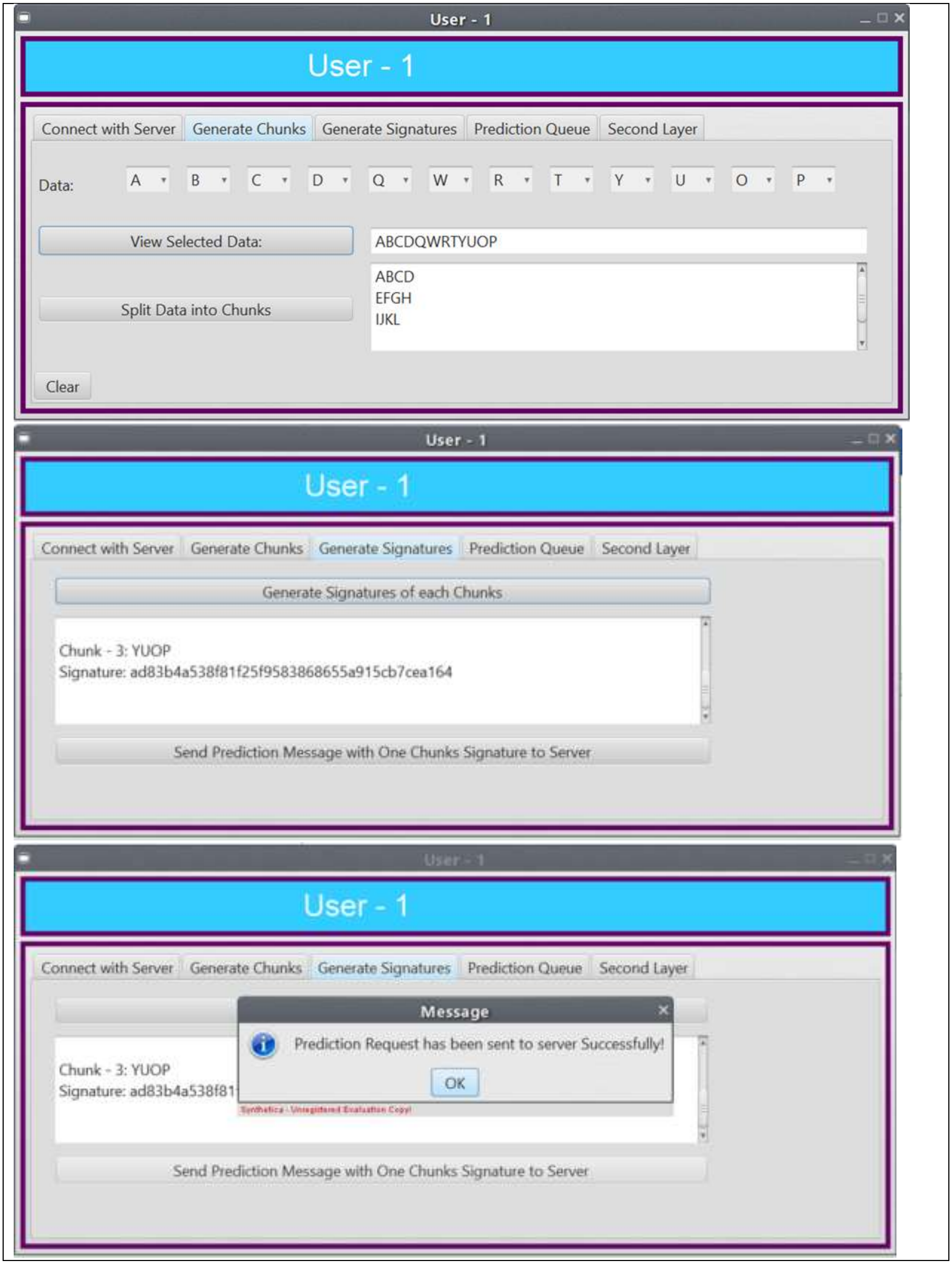


2) ABCDQWRTYUOP



Cloud Server Id
1

Row Id	Chunk Id	Chunk	Signature
1	1	ABCD	fb2f85c88567f3c8c...
1	2	EFGH	a7d14baf30ed1bb...
1	3	IJKL	65c9b3997d89ef7c...



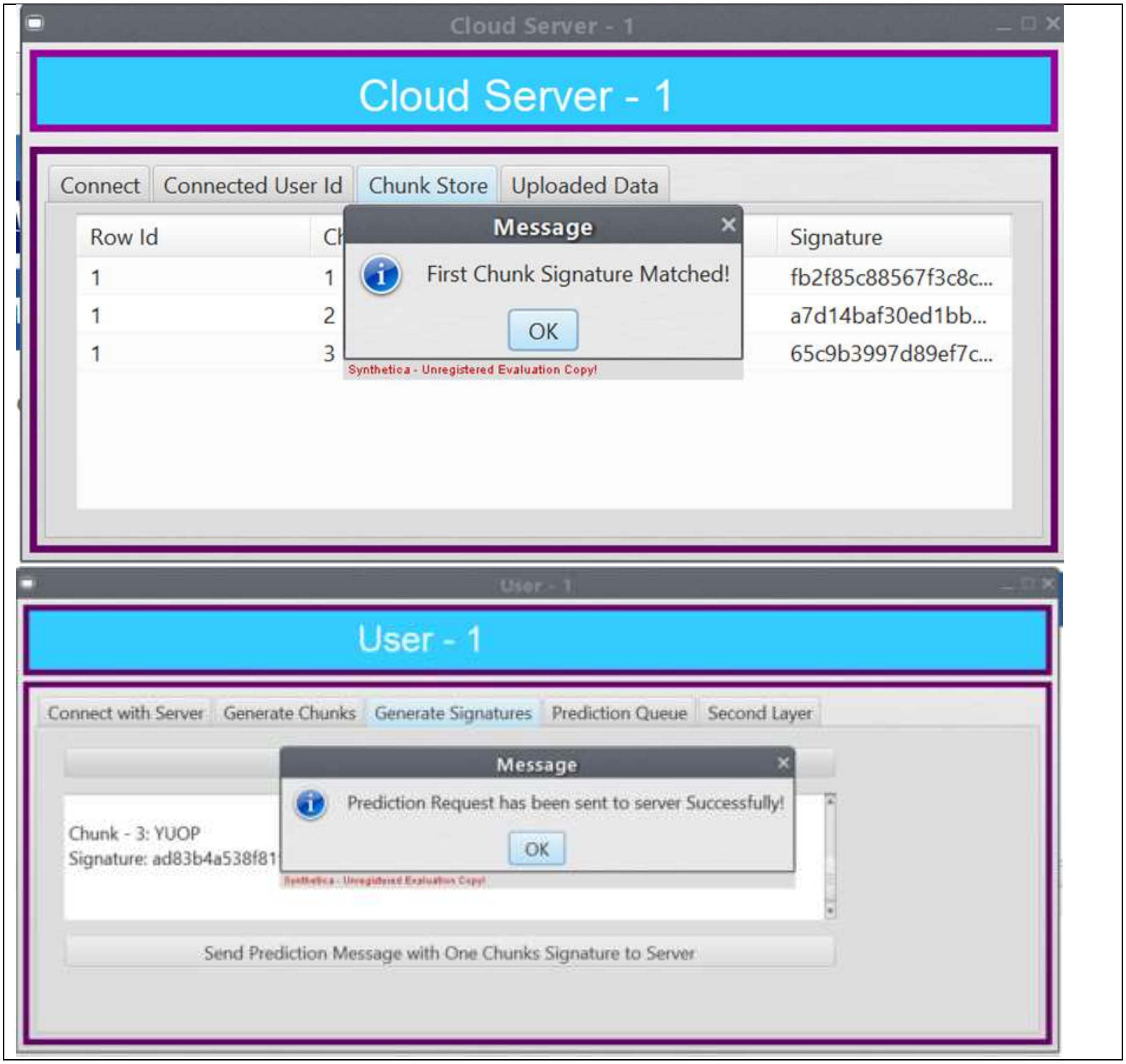
ABCDQWRTYUOP

ABCD
EFGH
IJKL

Chunk - 3: YUOP
Signature: ad83b4a538f81f25f9583868655a915cb7cea164

Prediction Request has been sent to server Successfully!

Chunk - 3: YUOP
Signature: ad83b4a538f81



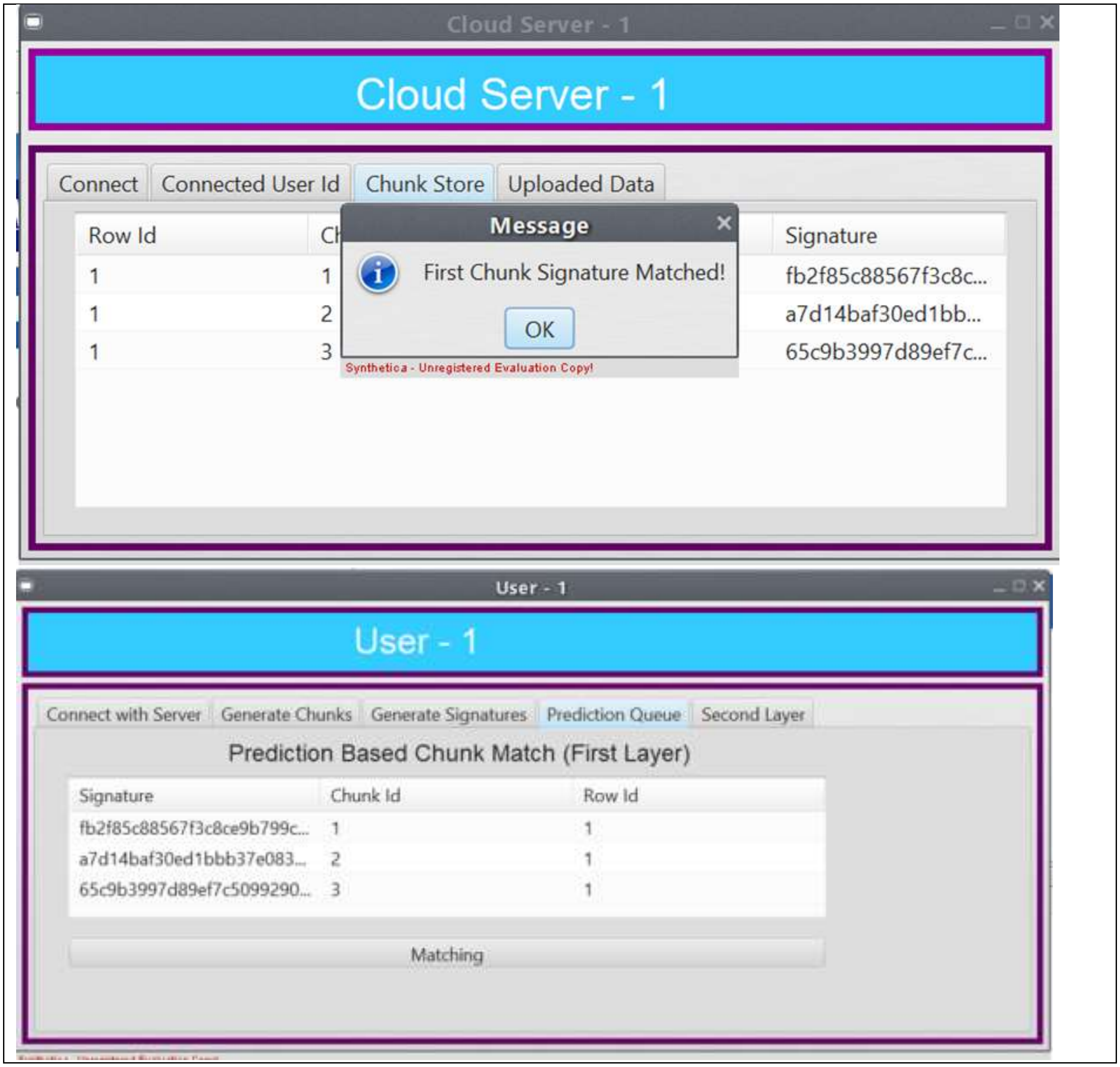
Row Id	Ch	Signature
1	1	fb2f85c88567f3c8c...
1	2	a7d14baf30ed1bb...
1	3	65c9b3997d89ef7c...

First Chunk Signature Matched!

User - 1
Connect with Server
Generate Chunks
Generate Signatures
Prediction Queue
Second Layer
Message
OK
Synthetica - Unregistered Evaluation Copy!
Send Prediction Message with One Chunks Signature to Server

Chunk - 3: YUOP
Signature: ad83b4a538f81

Prediction Request has been sent to server Successfully!

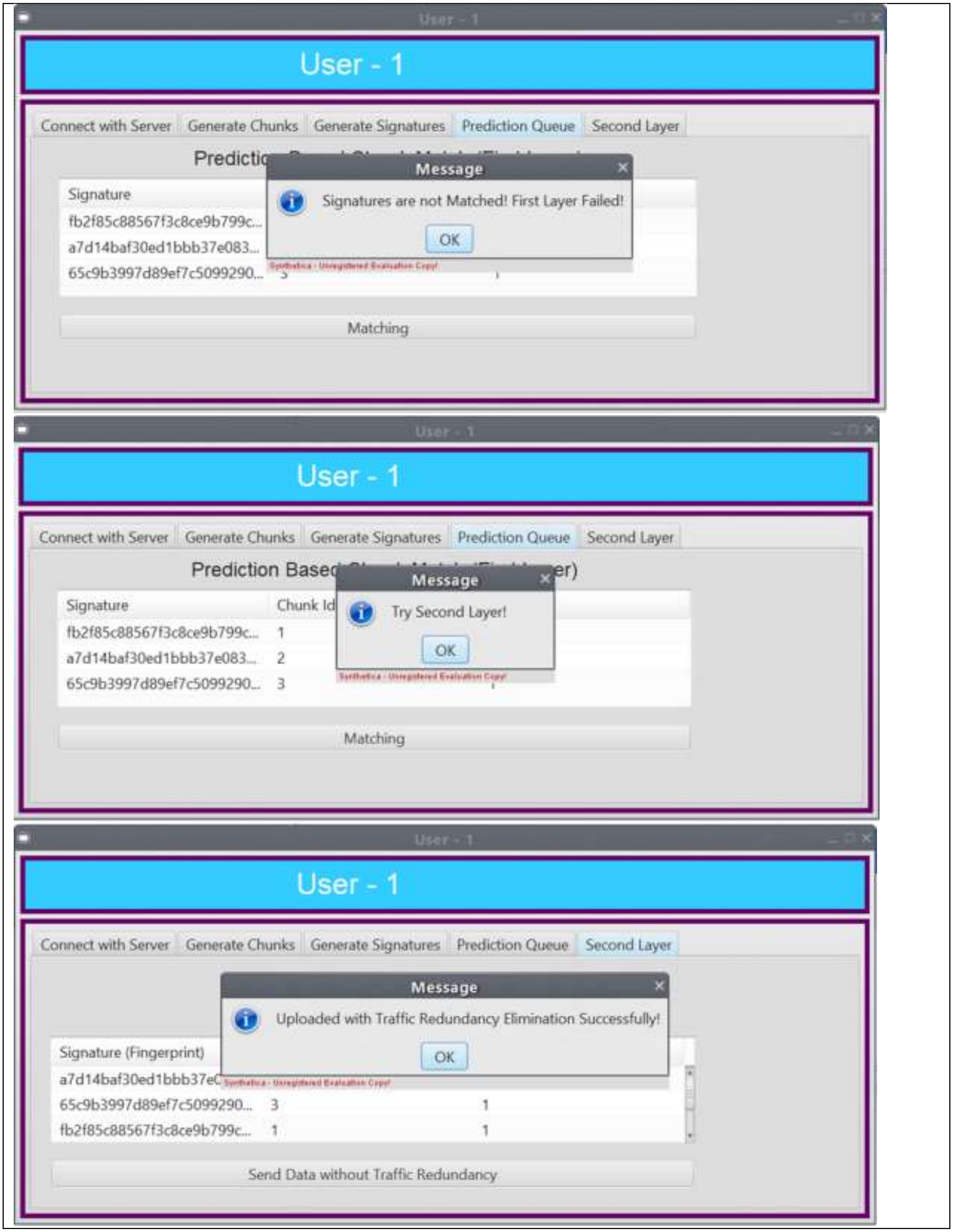


First Chunk Signature Matched!

Row Id	Ch	Signature
1	1	fb2f85c88567f3c8c...
1	2	a7d14baf30ed1bb...
1	3	65c9b3997d89ef7c...

Prediction Based Chunk Match (First Layer)

Signature	Chunk Id	Row Id
fb2f85c88567f3c8ce9b799c...	1	1
a7d14baf30ed1bbb37e083...	2	1
65c9b3997d89ef7c5099290...	3	1



Predictio

Signature		
fb2f85c88567f3c8ce9b799c...		
a7d14baf30ed1bbb37e083...		
65c9b3997d89ef7c5099290...	3	1

Message

Signatures are not Matched! First Layer Failed!

Prediction Based ... er)

Signature	Chunk Id	
fb2f85c88567f3c8ce9b799c...	1	
a7d14baf30ed1bbb37e083...	2	
65c9b3997d89ef7c5099290...	3	1

Message

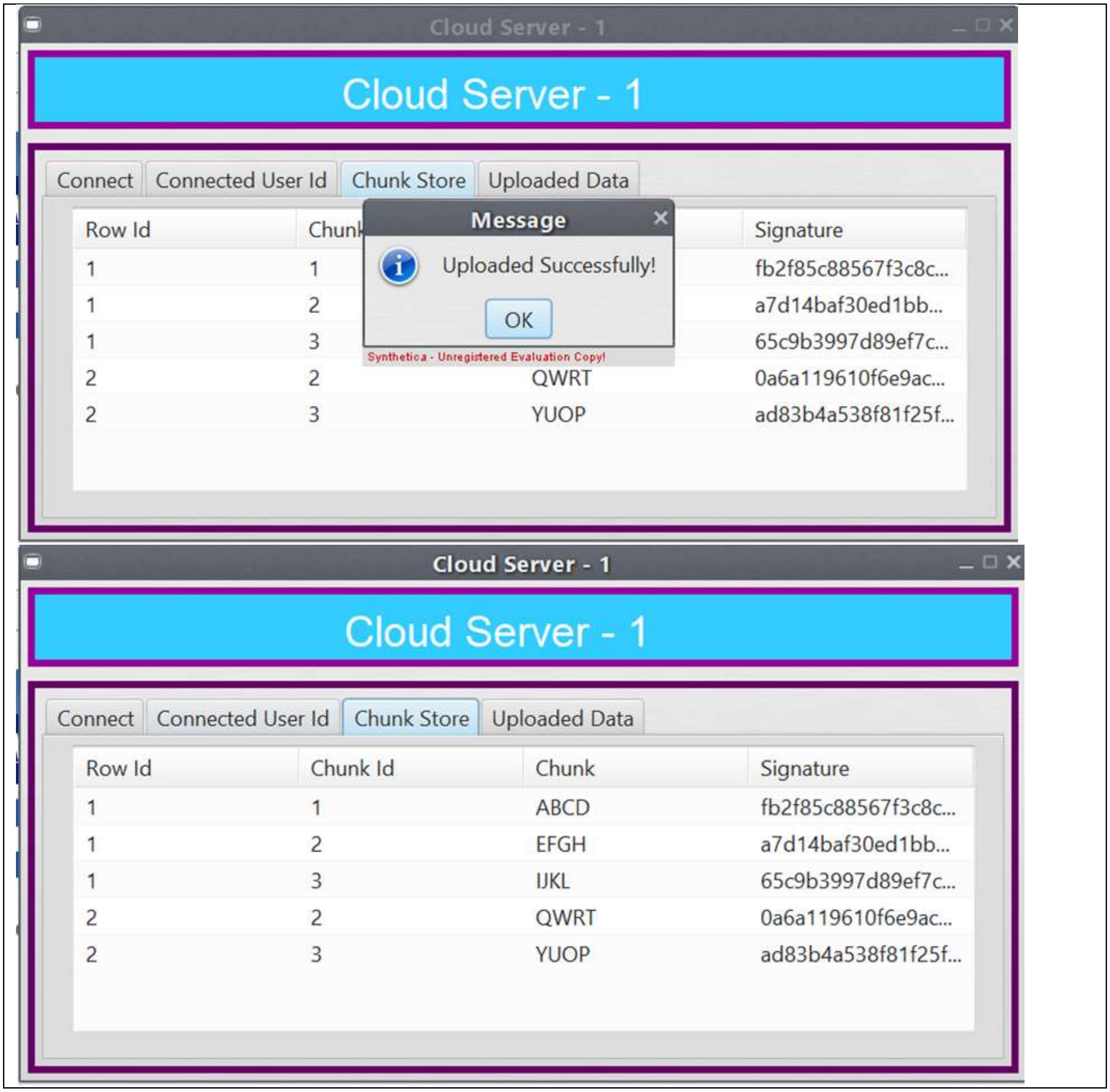
Try Second Layer!

Signature (Fingerprint)		
a7d14baf30ed1bbb37eC		
65c9b3997d89ef7c5099290...	3	1
fb2f85c88567f3c8ce9b799c...	1	1

Message

Uploaded with Traffic Redundancy Elimination Successfully!

OK
Synthetica - Unregistered Evaluation Copy!
Send Data without Traffic Redundancy

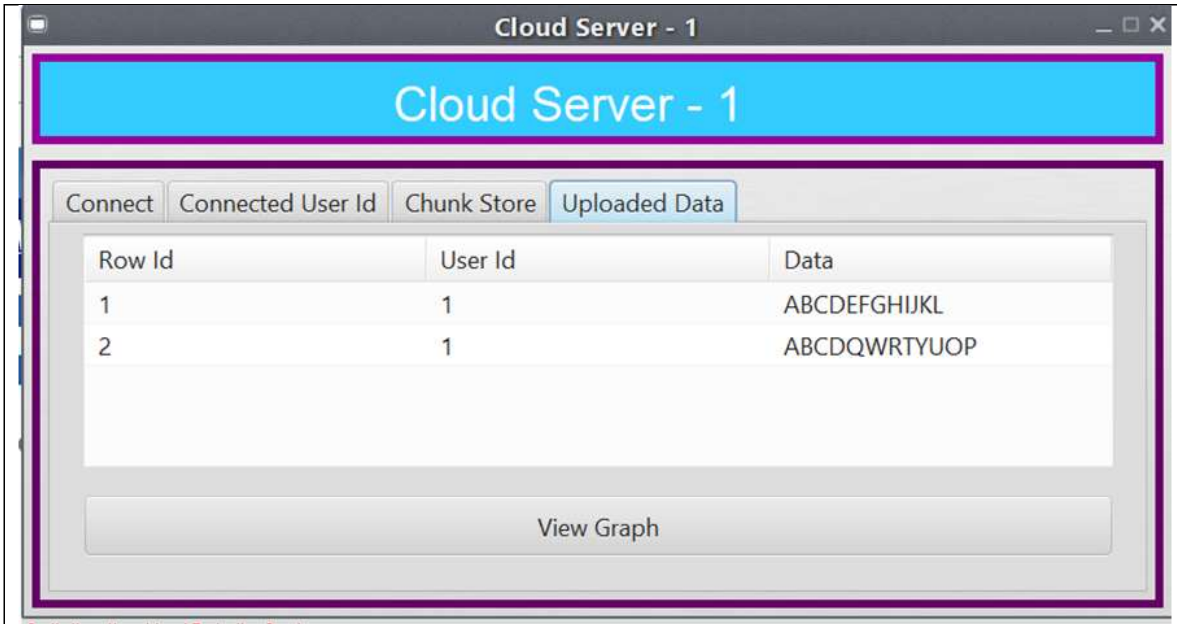


Uploaded Successfully!

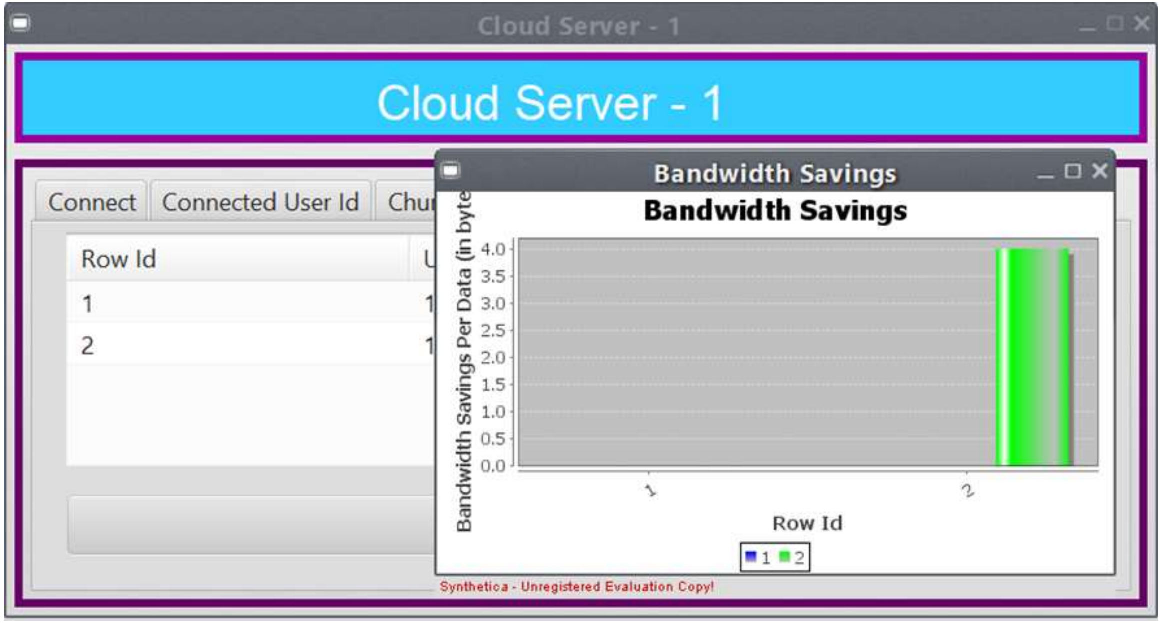
Synthetica - Unregistered Evaluation Copy!

Row Id	Chunk		Signature
1	1		fb2f85c88567f3c8c...
1	2		a7d14baf30ed1bb...
1	3		65c9b3997d89ef7c...
2	2	QWRT	0a6a119610f6e9ac...
2	3	YUOP	ad83b4a538f81f25f...

Row Id	Chunk Id	Chunk	Signature
1	1	ABCD	fb2f85c88567f3c8c...
1	2	EFGH	a7d14baf30ed1bb...
1	3	IJKL	65c9b3997d89ef7c...
2	2	QWRT	0a6a119610f6e9ac...
2	3	YUOP	ad83b4a538f81f25f...



Row Id	User Id	Data
1	1	ABCDEFGHIJKL
2	1	ABCDQWRTYUOP

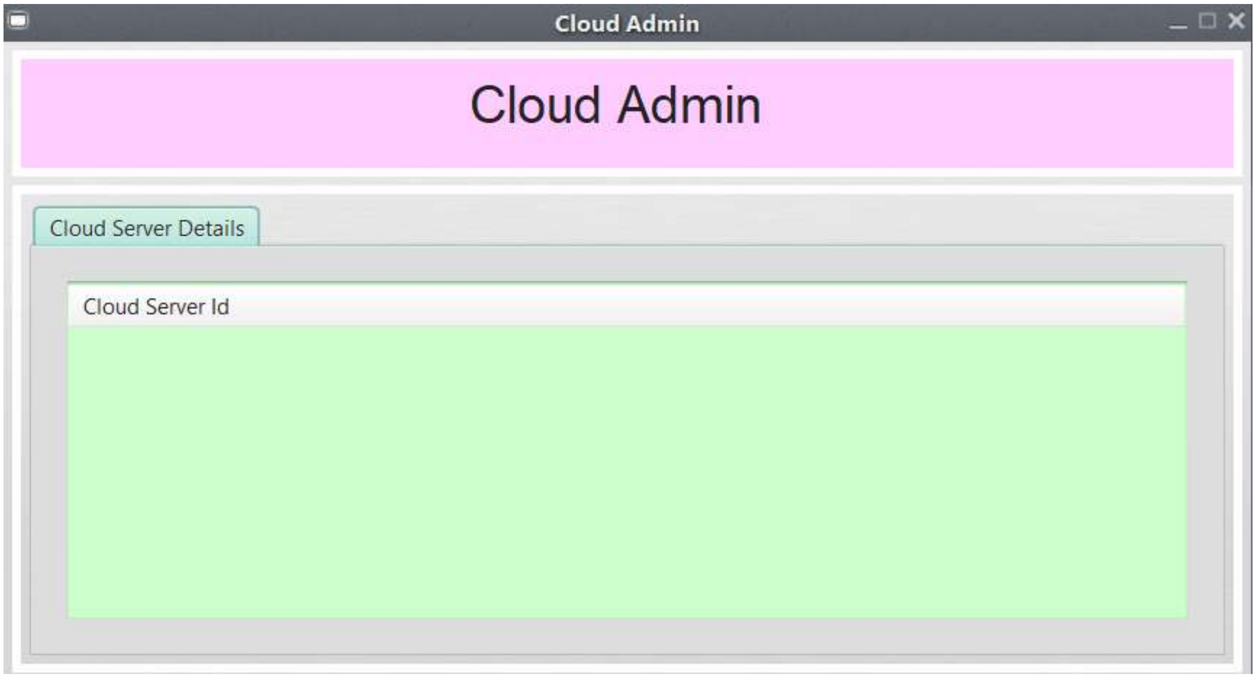


5. Results and Discussion

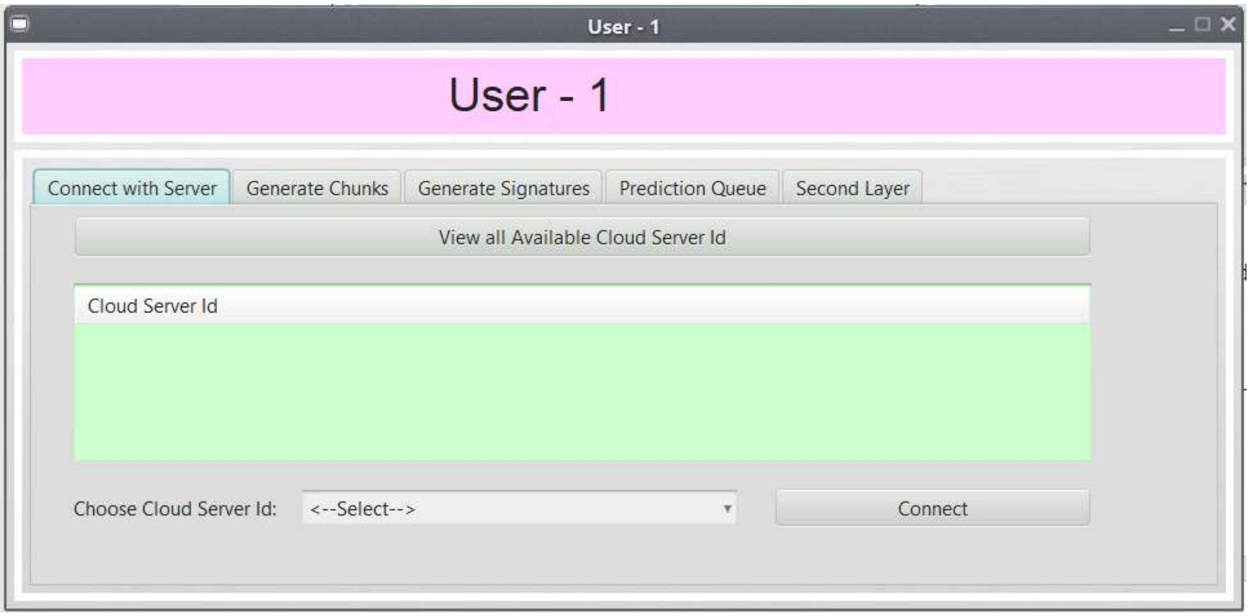
5.1 Implementation Results

The application we tried to build to implement the idea has 3 Components: -

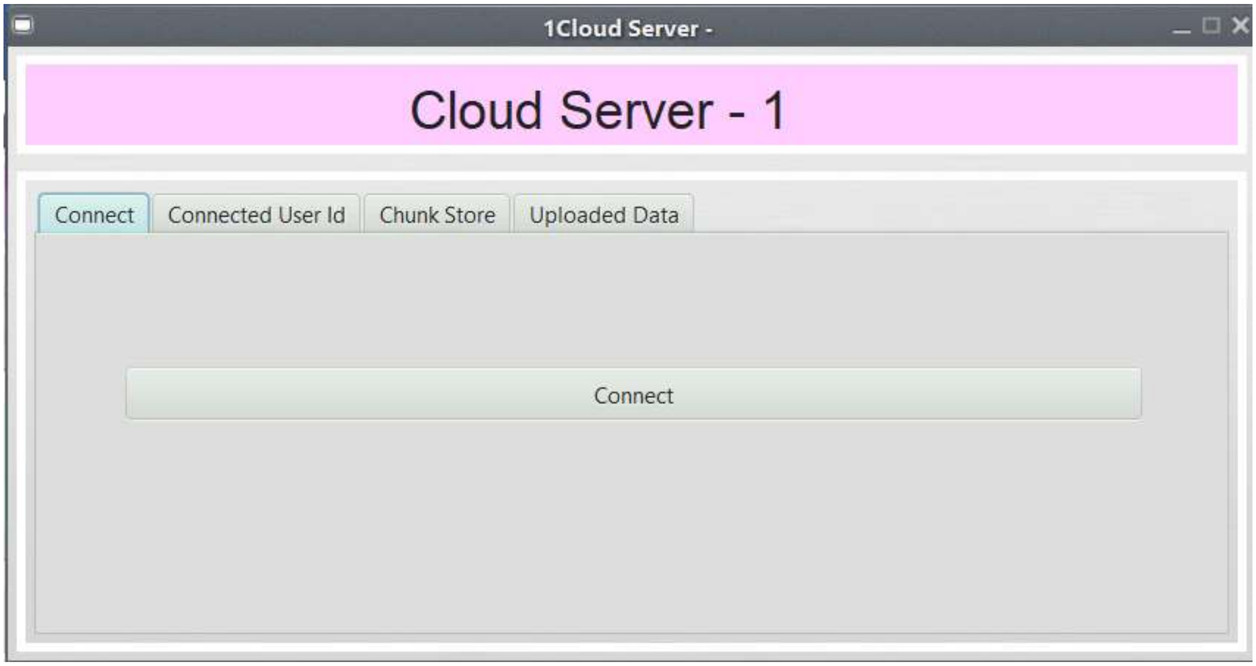
Admin -



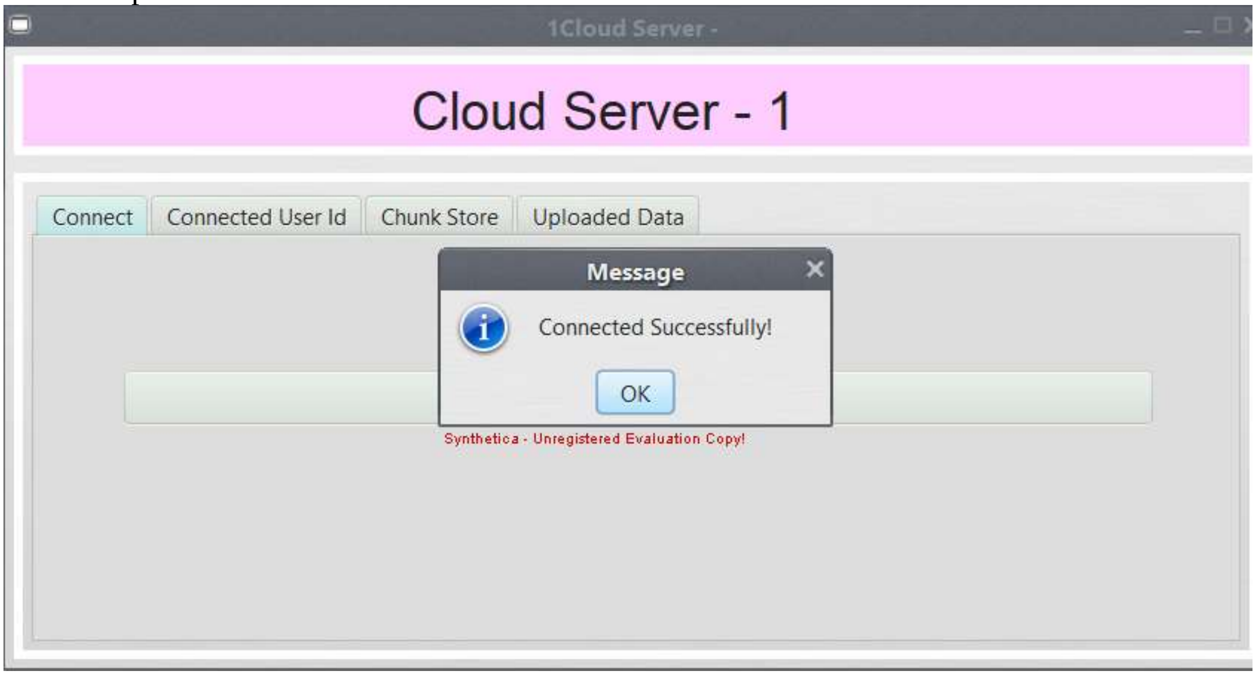
User-



Server –

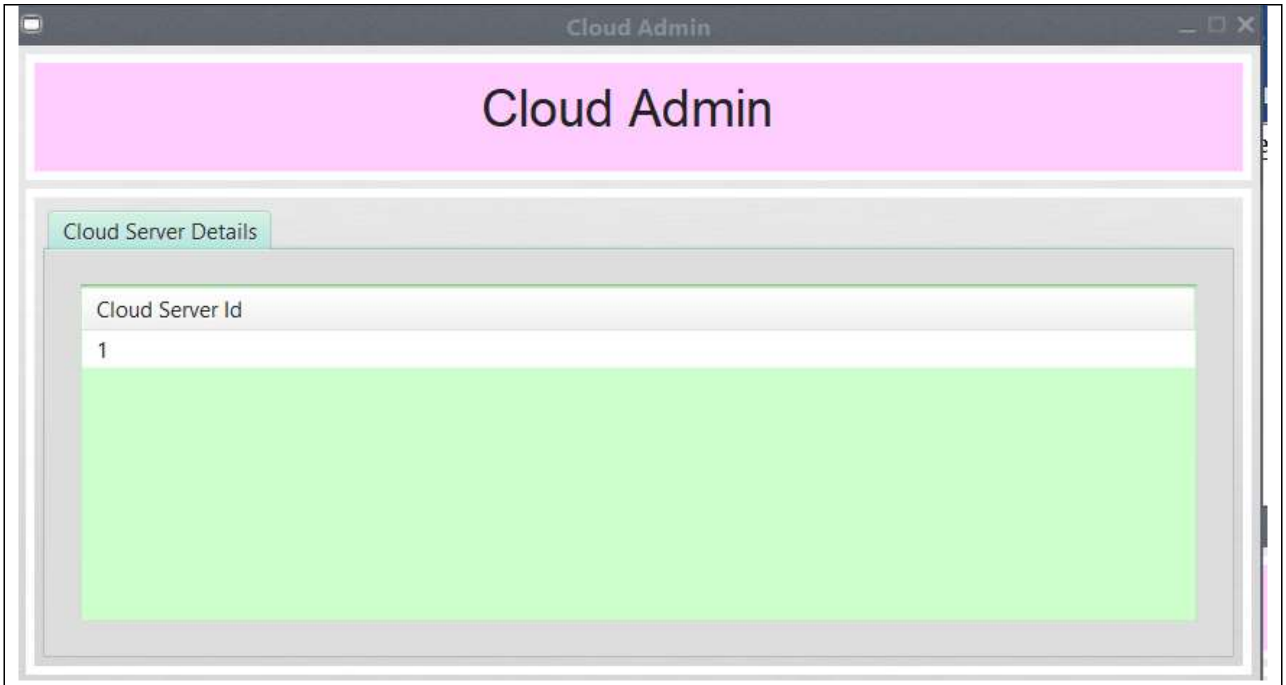


First we open the cloud server and connect it to the user as shown below



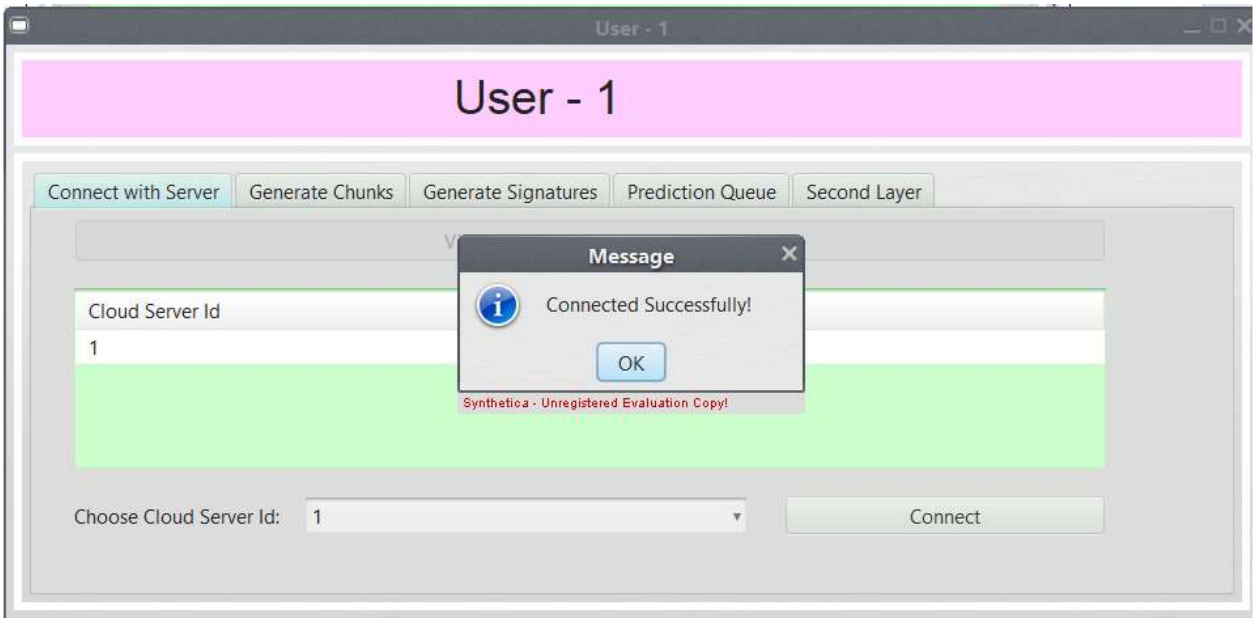
Connected Successfully!

The Admin window lists the server currently running



Cloud Server Id
1

Then we connect user to the server as shown below



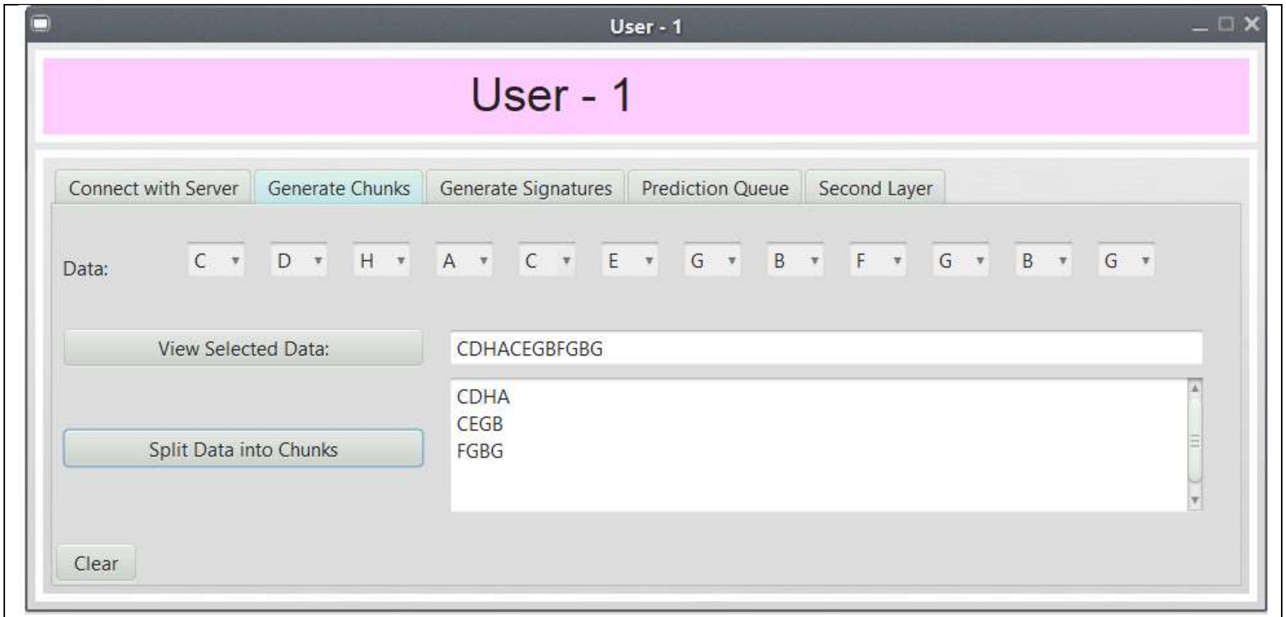
Connected Successfully!

Synthetica - Unregistered Evaluation Copy!

Cloud Server Id
1

Choose Cloud Server Id: 1

Then we split data into chunks

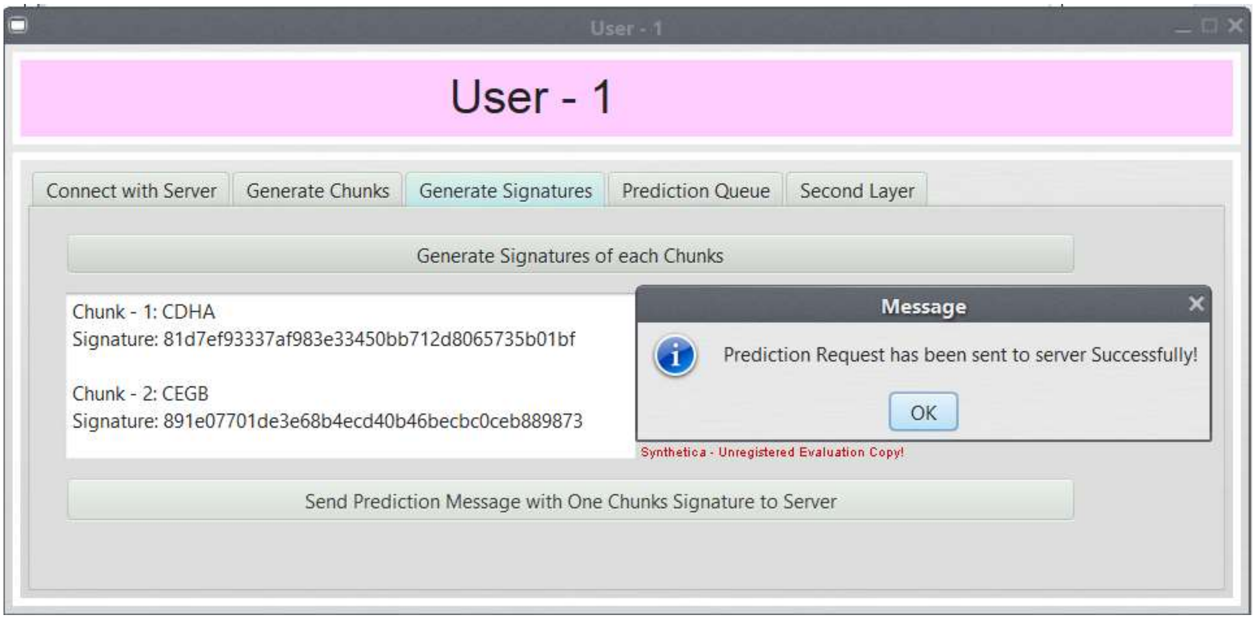


Data: C D H A C E G B F G B G

View Selected Data: CDHACEGBFGBG

CDHA
CEGB
FGBG

Then we generate signature of each chunk and send the preduiction request with one chunk to the server as shown

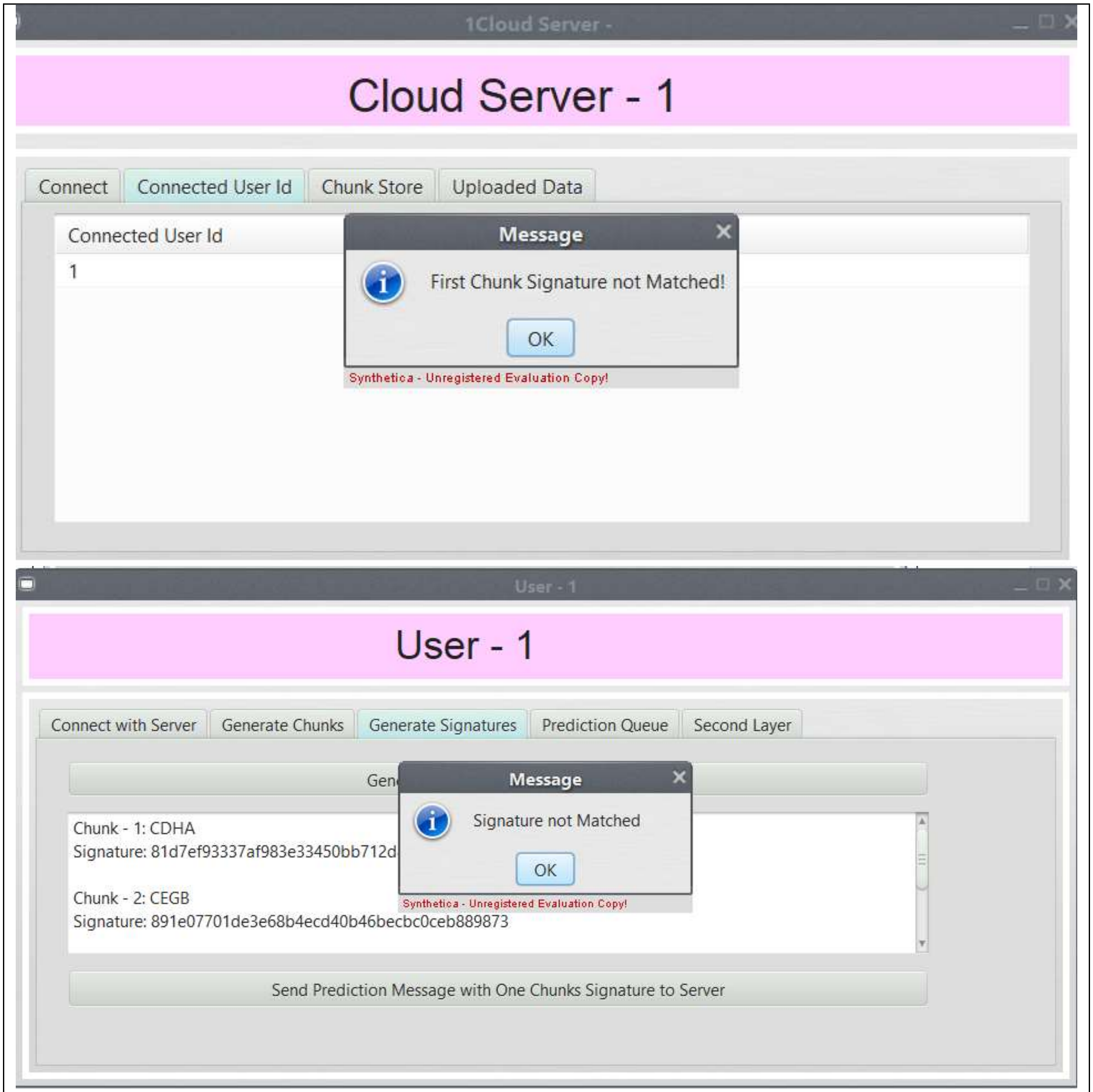

User - 1
User - 1
Connect with Server
Generate Chunks
Generate Signatures
Prediction Queue
Second Layer
Generate Signatures of each Chunks
Send Prediction Message with One Chunks Signature to Server
Message
OK
Synthetica - Unregistered Evaluation Copy!

Chunk - 1: CDHA
Signature: 81d7ef93337af983e33450bb712d8065735b01bf

Chunk - 2: CEGB
Signature: 891e07701de3e68b4ecd40b46becbc0ceb889873

Prediction Request has been sent to server Successfully!

The server doesn’t find any match data with the chunk sent by the user and sends the prediction result back to the user.



Connected User Id
1

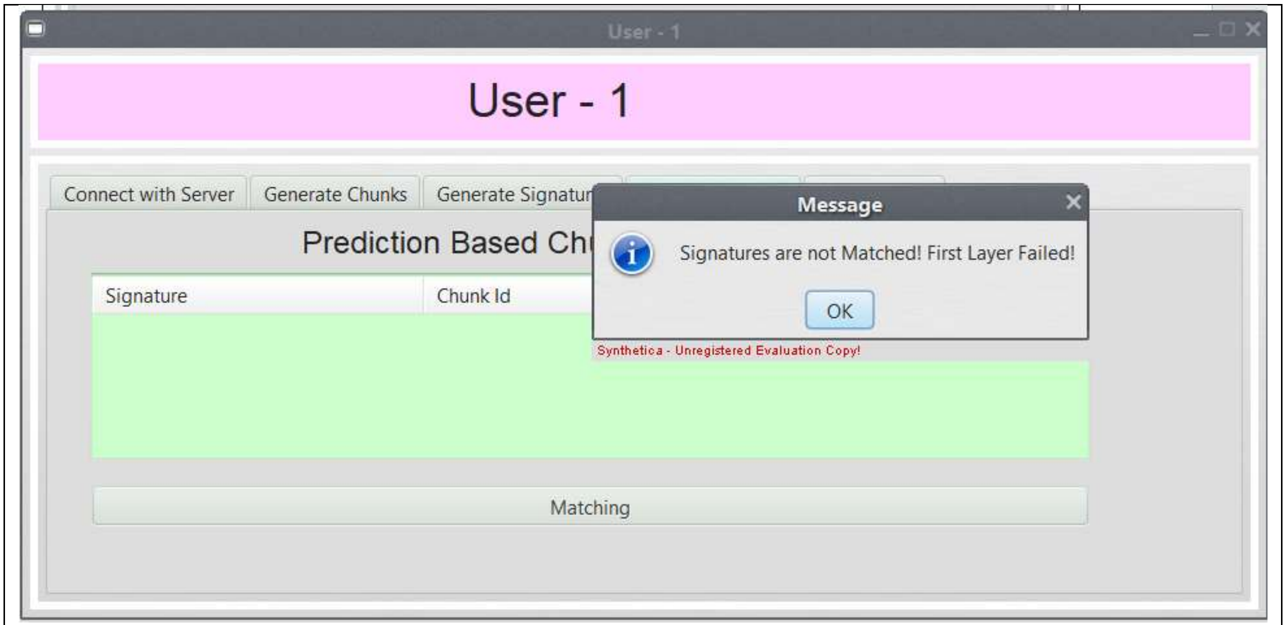
First Chunk Signature not Matched!

Chunk - 1: CDHA
Signature: 81d7ef93337af983e33450bb712d

Chunk - 2: CEGB
Signature: 891e07701de3e68b4ecd40b46becbc0ceb889873

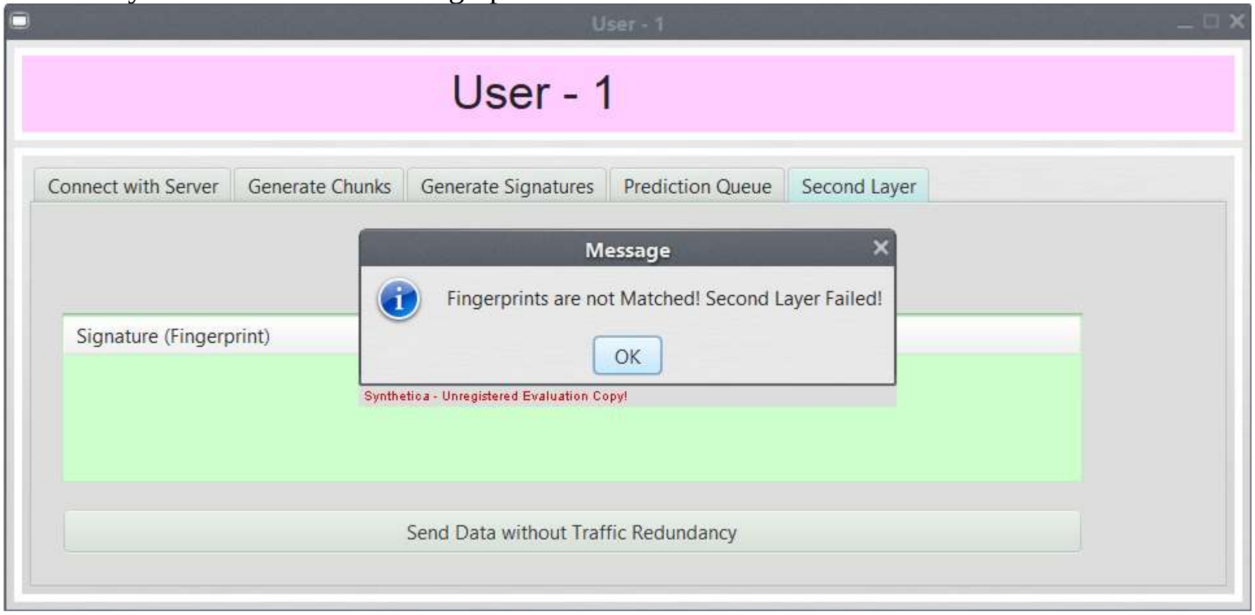
Signature not Matched

Then we try matching with the prediction queue which is empty right now since it is first data sent and hence first layer fails.



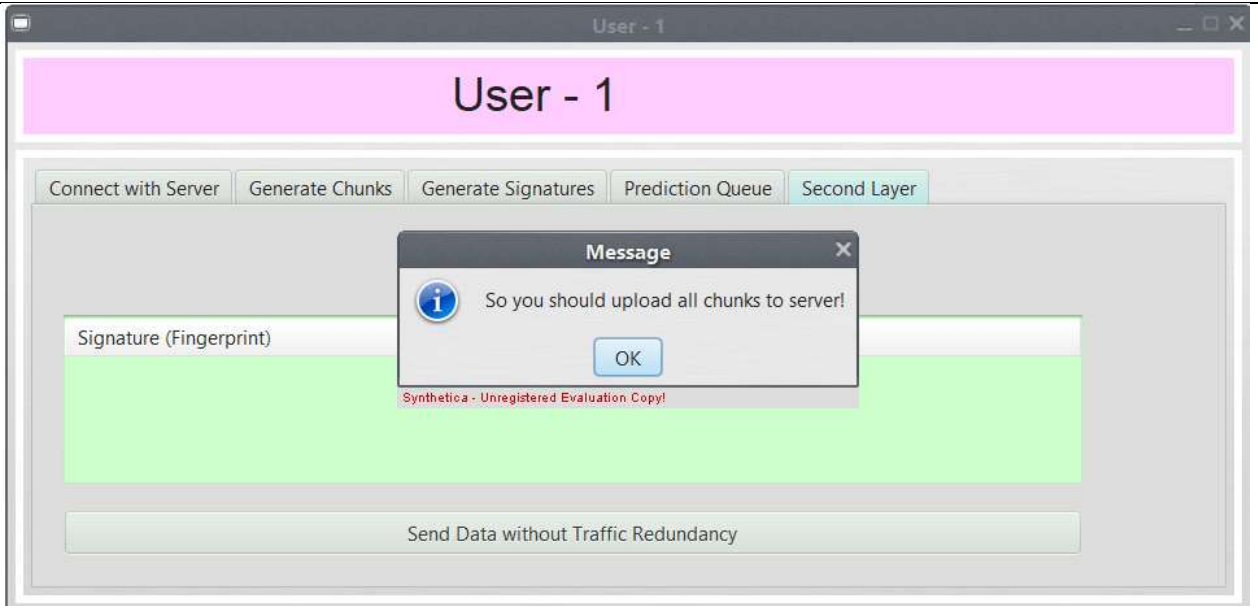
Signatures are not Matched! First Layer Failed!

Second layer also fails since no fingerprint is stored



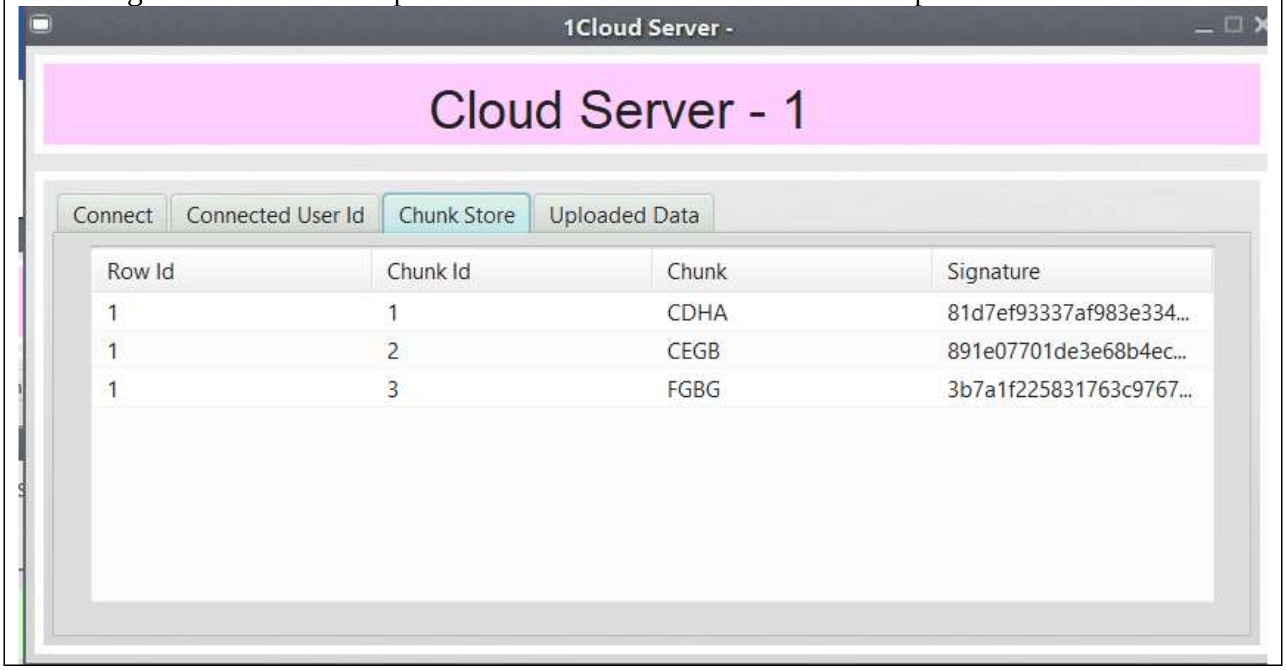
Fingerprints are not Matched! Second Layer Failed!

We get the message to upload the entire data

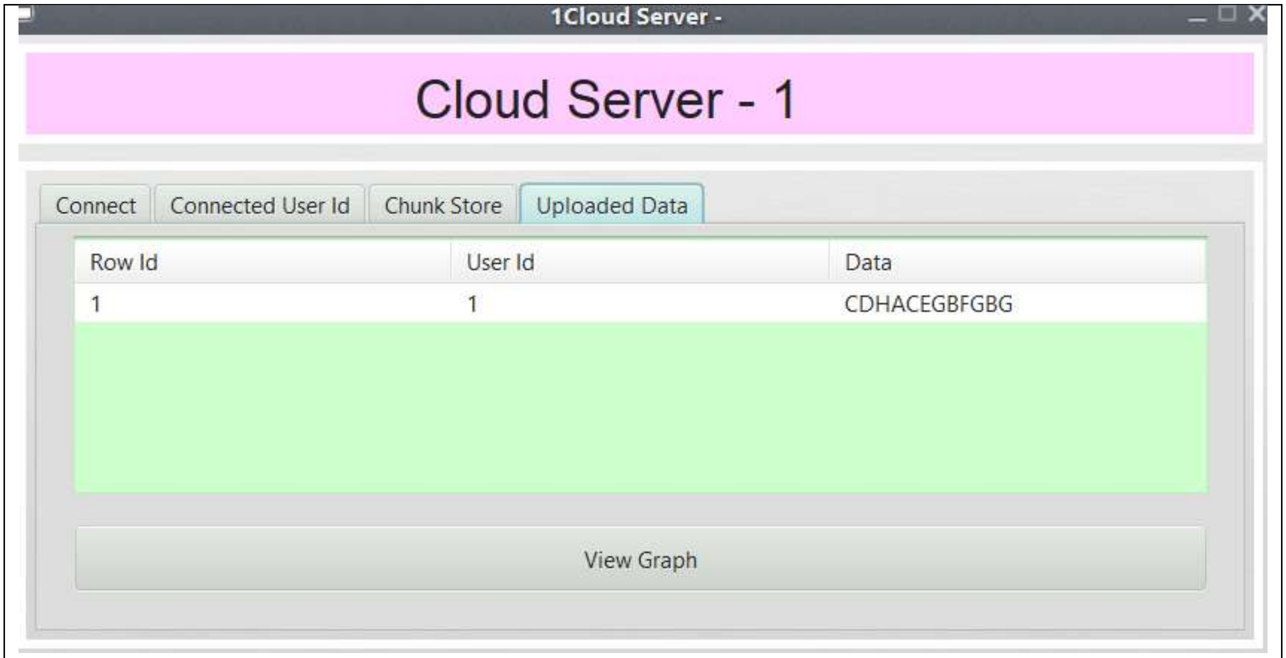


So you should upload all chunks to server!

And we get the chunk store updated on server side and entire data is uploaded

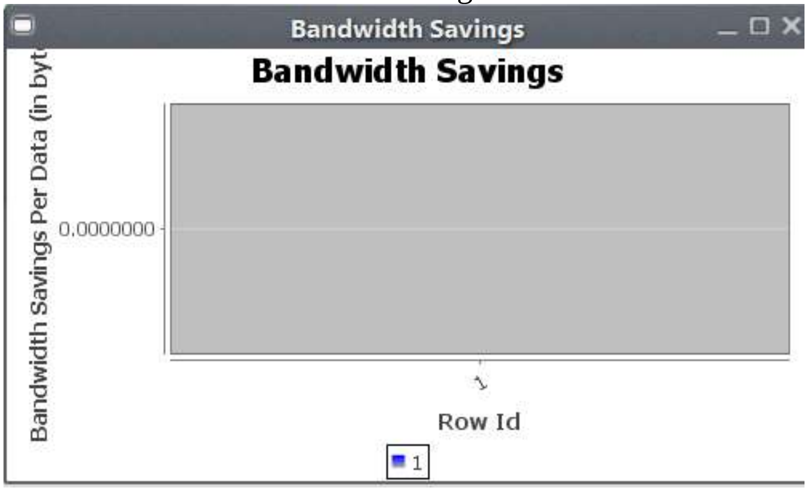


Row Id	Chunk Id	Chunk	Signature
1	1	CDHA	81d7ef93337af983e334...
1	2	CEGB	891e07701de3e68b4ec...
1	3	FGBG	3b7a1f225831763c9767...

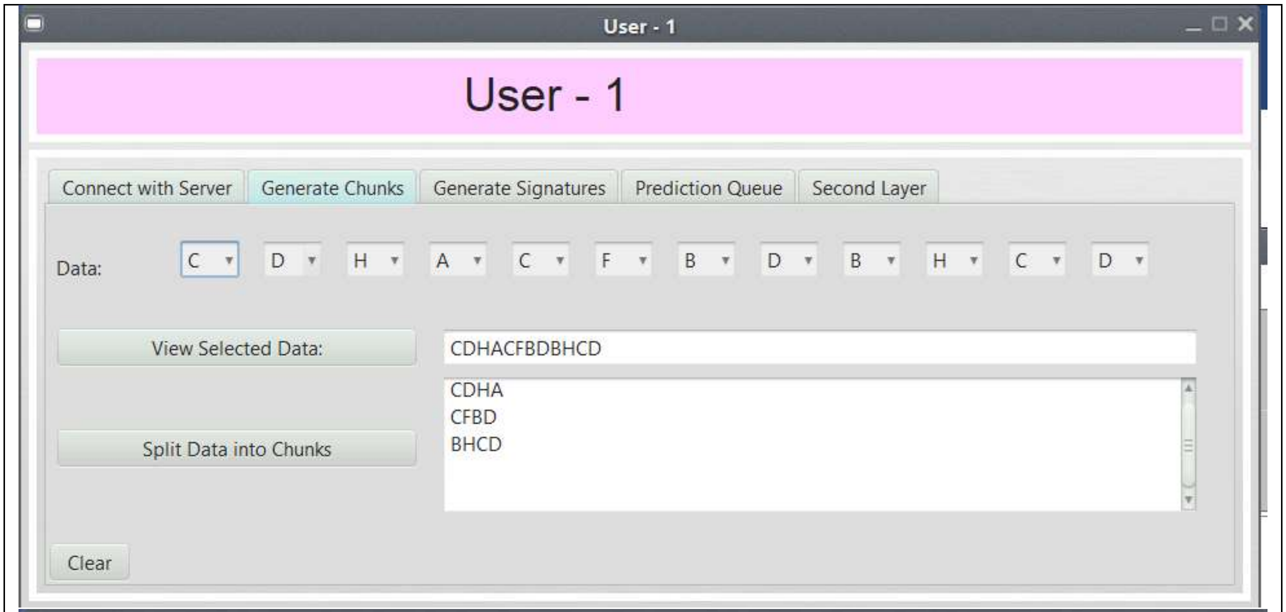


Row Id	User Id	Data
1	1	CDHACEGBFGBG

We see there is no bandwidth saving in this case



But if we upload partially duplicate data as shown below

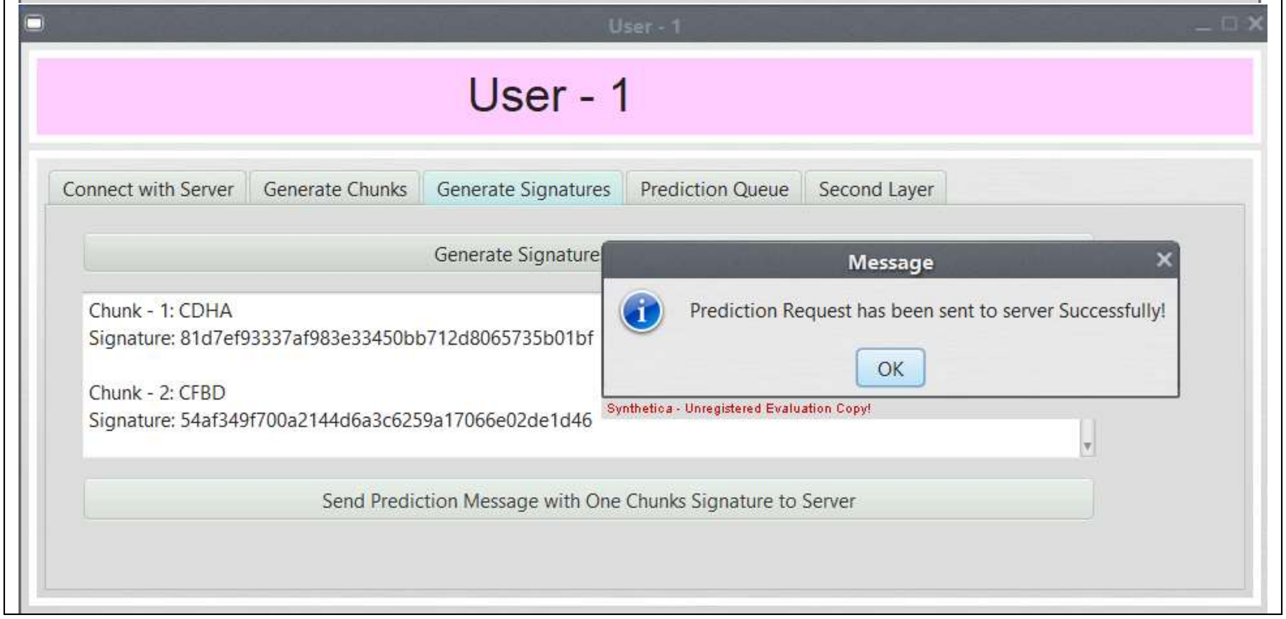


User - 1

Data: C D H A C F B D B H C D

View Selected Data: CDHACFBDBHCD

```
CDHA
CFBD
BHCD
```



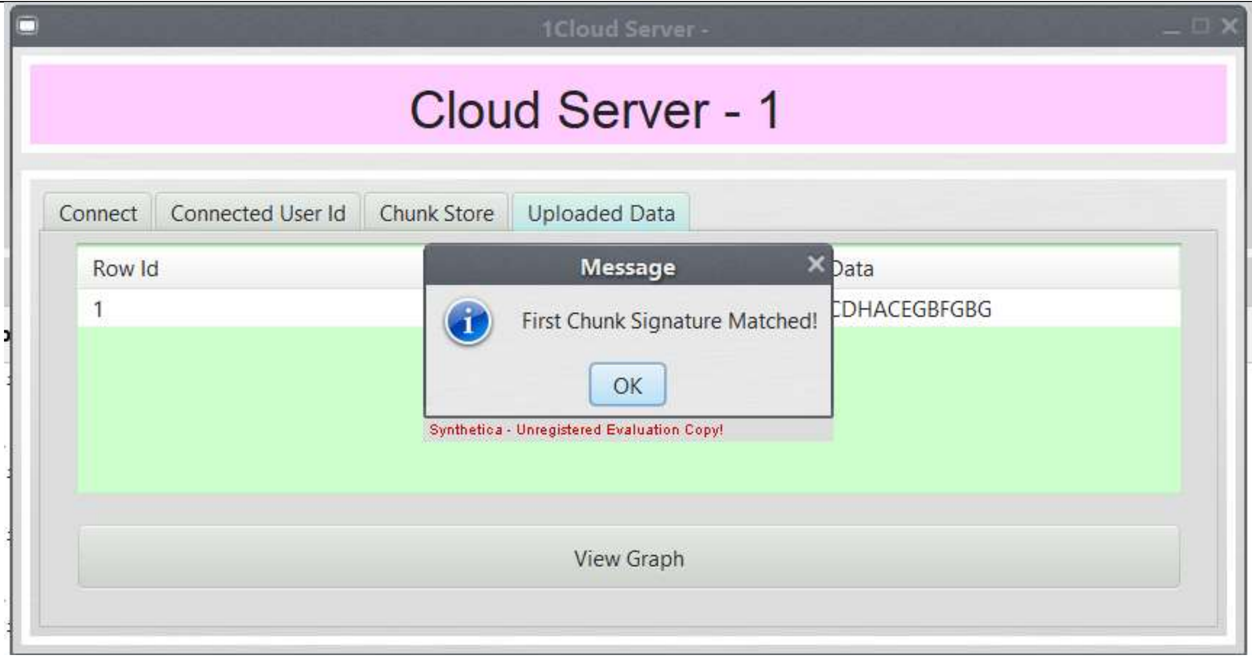
User - 1

```
Chunk - 1: CDHA
Signature: 81d7ef93337af983e33450bb712d8065735b01bf

Chunk - 2: CFBD
Signature: 54af349f700a2144d6a3c6259a17066e02de1d46
```

Message

Prediction Request has been sent to server Successfully!



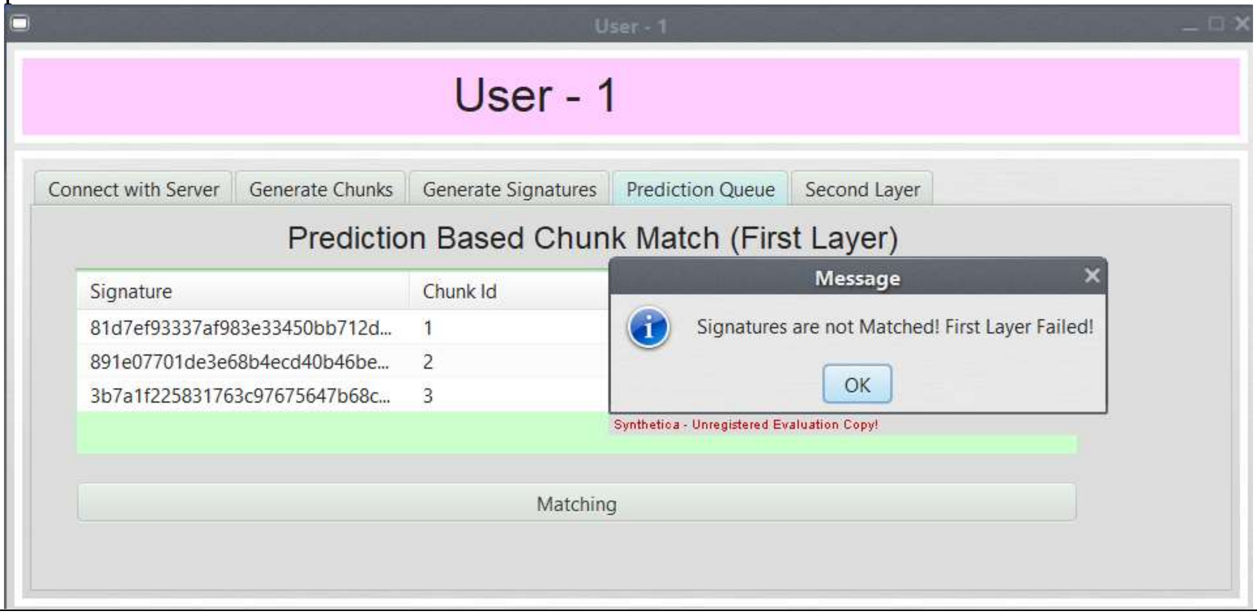
Row Id	Data
1	CDHACEGBFGBG

Message

First Chunk Signature Matched!

Synthetica - Unregistered Evaluation Copy!

The first chunk matches but the first layer detects the uniqueness in data and redundancy is not predicted.



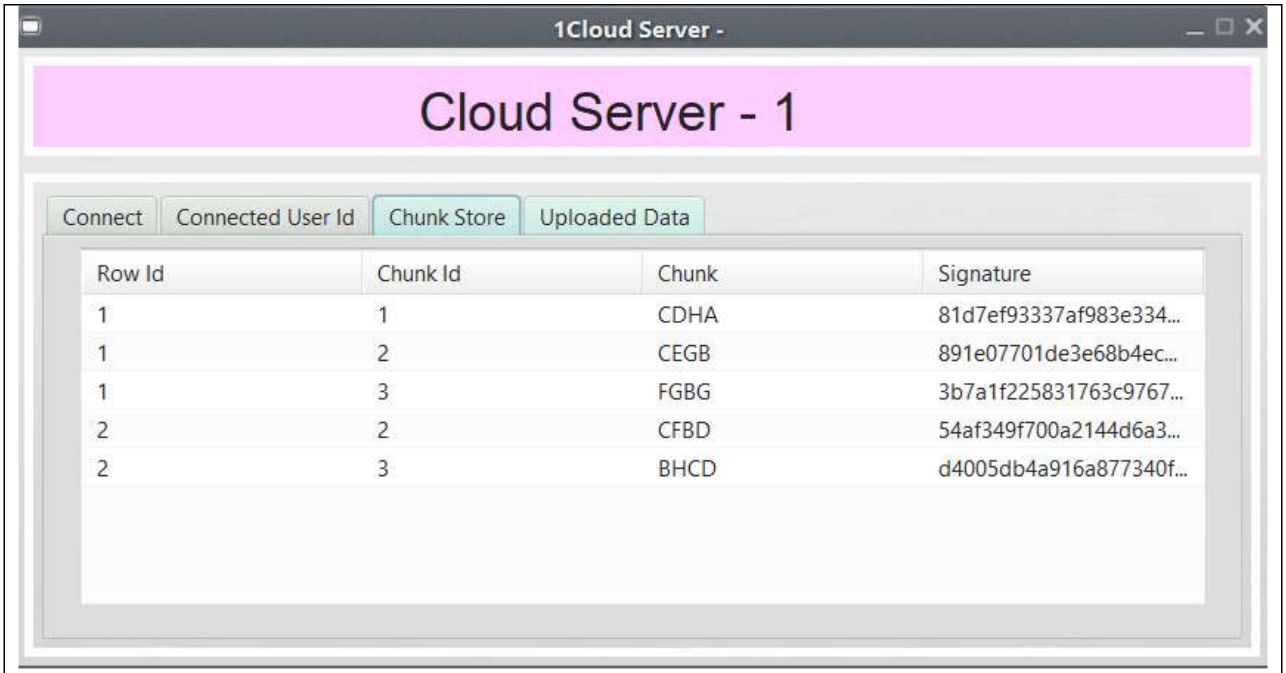
Prediction Based Chunk Match (First Layer)

Signature	Chunk Id
81d7ef93337af983e33450bb712d...	1
891e07701de3e68b4ecd40b46be...	2
3b7a1f225831763c97675647b68c...	3

Message

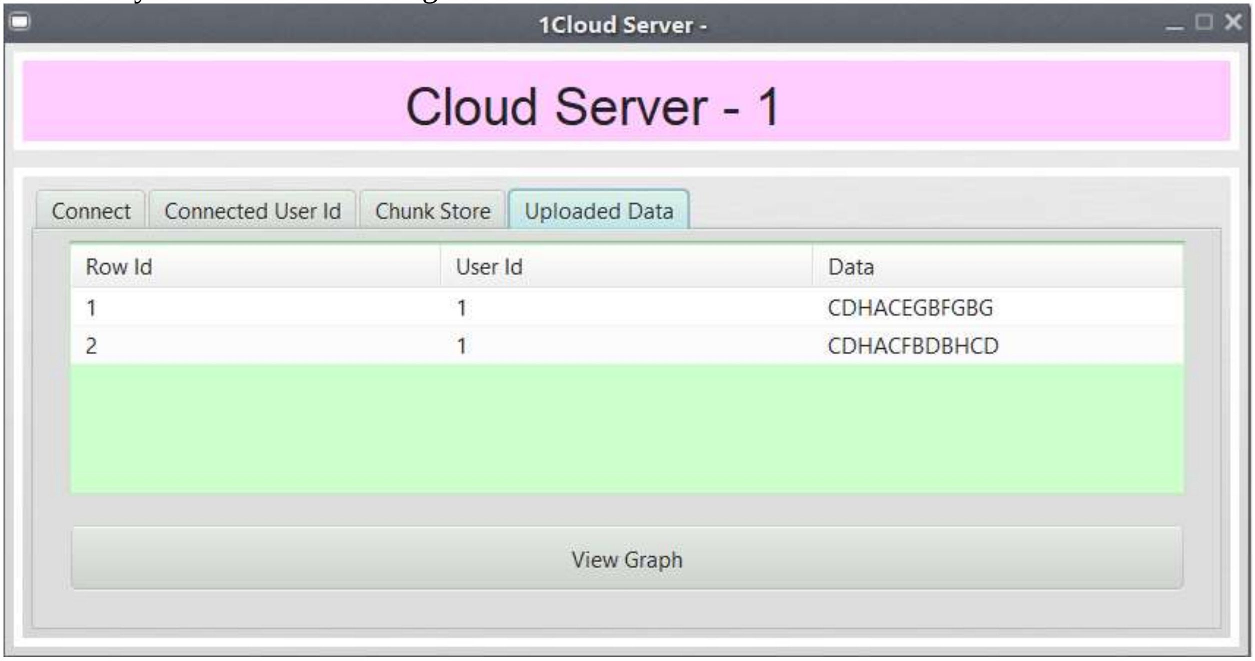
Signatures are not Matched! First Layer Failed!

Synthetica - Unregistered Evaluation Copy!



Row Id	Chunk Id	Chunk	Signature
1	1	CDHA	81d7ef93337af983e334...
1	2	CEGB	891e07701de3e68b4ec...
1	3	FGBG	3b7a1f225831763c9767...
2	2	CFBD	54af349f700a2144d6a3...
2	3	BHCD	d4005db4a916a877340f...

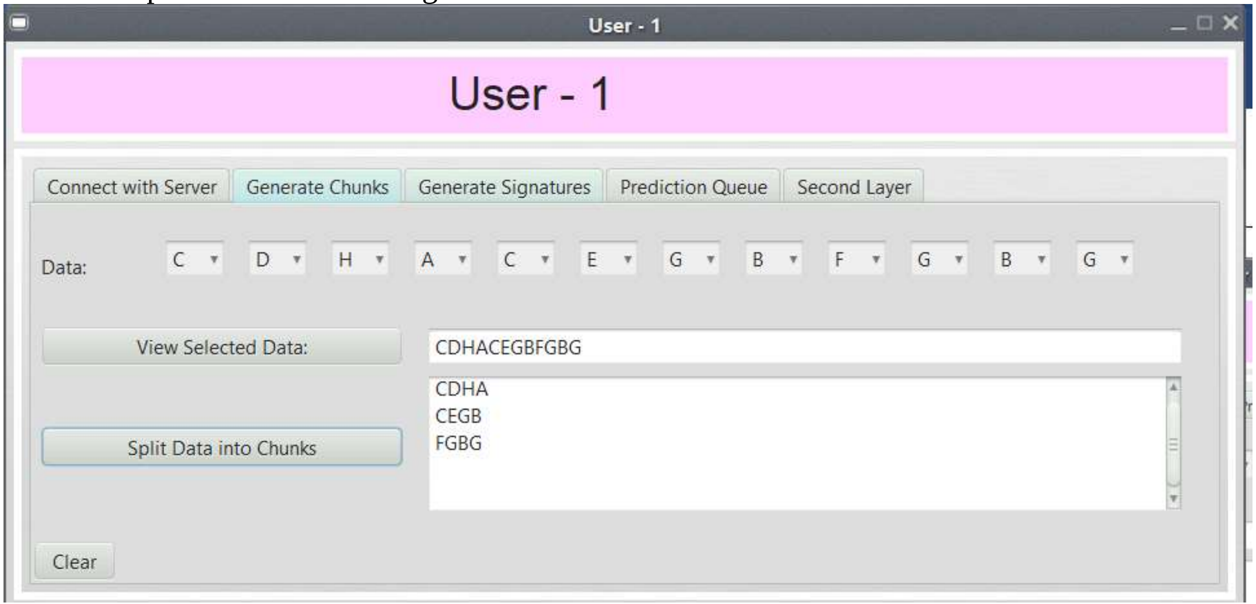
Hence only the chunks different gets transmitted



Row Id	User Id	Data
1	1	CDHACEGBFGBG
2	1	CDHACFBDBHCD

Hence the bandwidth saving is of 4 bytes.

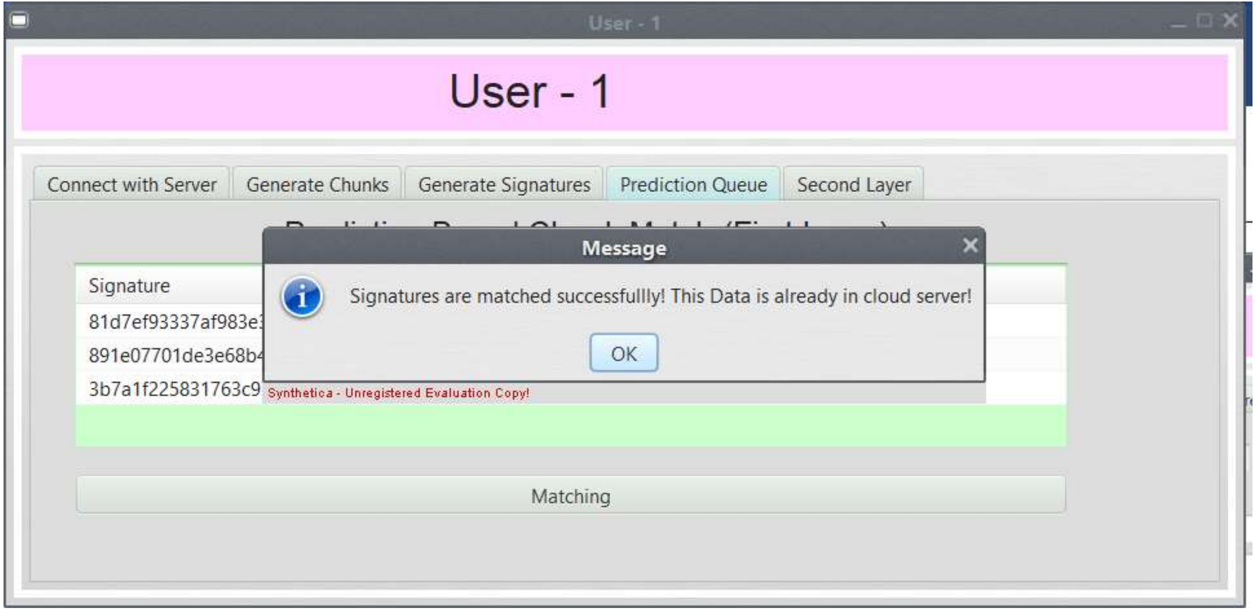
But if we upload the same data again



Data: C D H A C E G B F G B G

View Selected Data: CDHACEGBFGBG

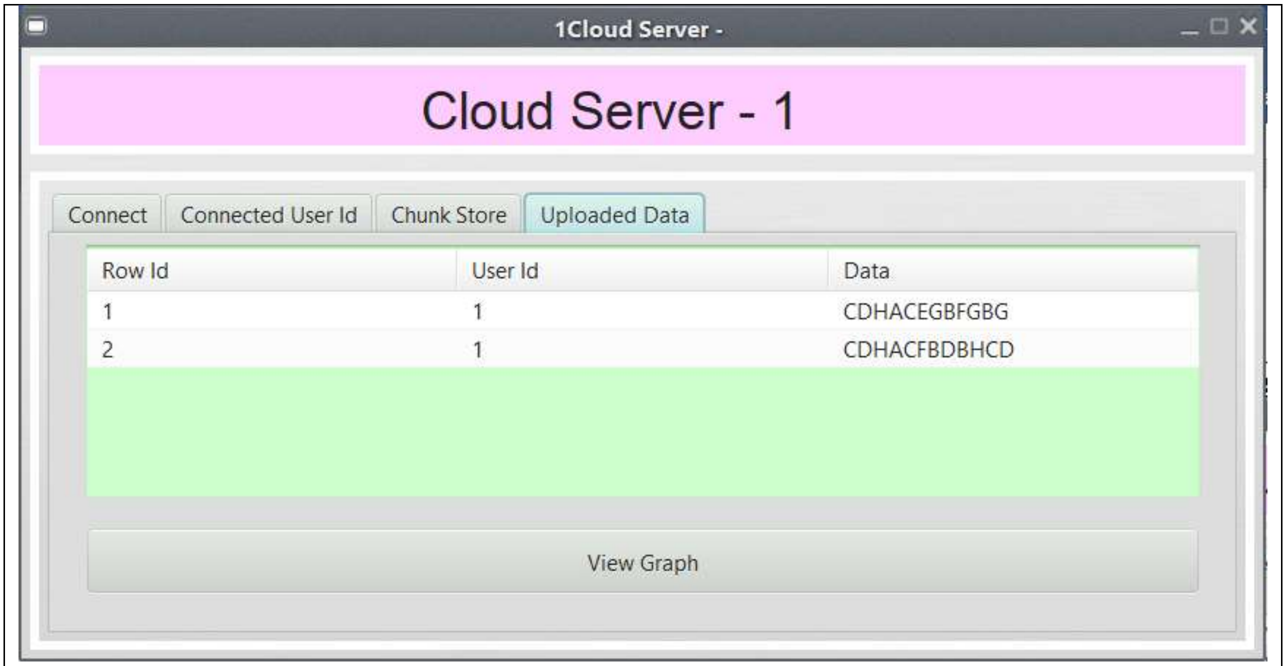
CDHA
CEGB
FGBG



Signature
81d7ef93337af983e
891e07701de3e68b4
3b7a1f225831763c9

Signatures are matched successfullly! This Data is already in cloud server!

The data redundancy is detected in the first layer only and hence no data is transmitted.



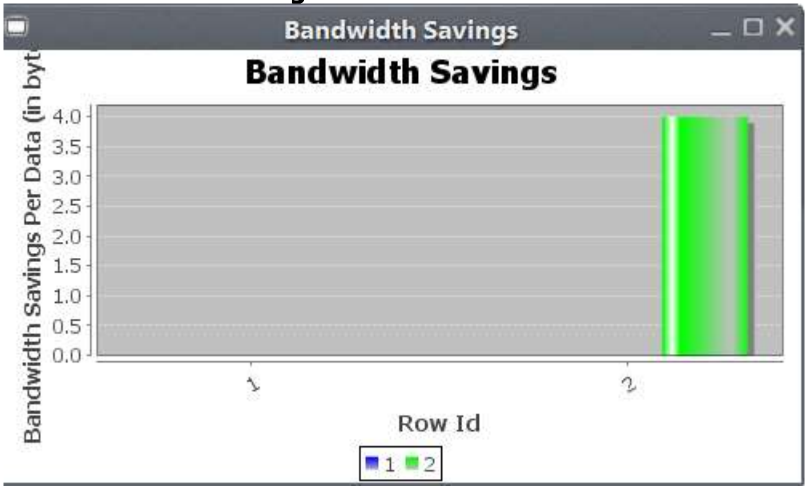
Row Id	User Id	Data
1	1	CDHACEGBFGBG
2	1	CDHACFBDBHCD

So, no bandwidth graph is required as no data is transmitted

5.2 Metrics

We have shown the result for small data size of 1 to 4 bytes here although the same can be scale up to large data of several MBs and GBs. As shown in the next session where we see the graph of the bandwidth saving and chunk size of an experiment on Linux source workload.

5.3 Results in table/Graph/Data (No screenshots, only text form of data in table), Graph should be drawn using Excel tool

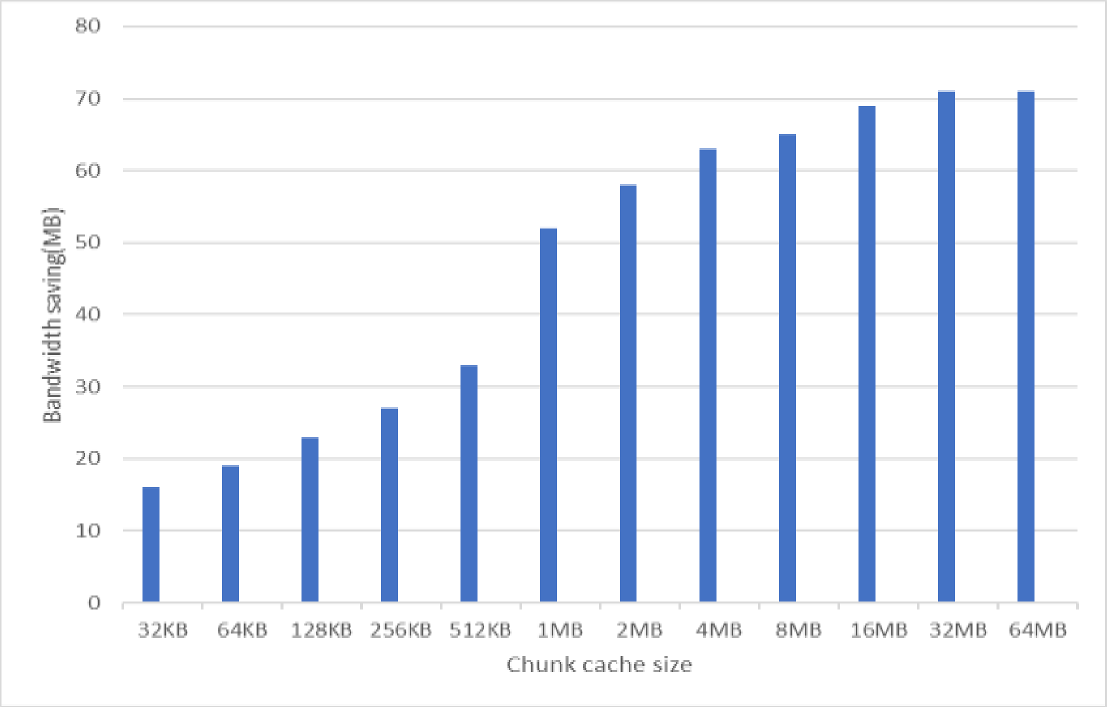


The above implementation graph showing bandwidth saving for case1- unique data upload and case 2- partially redundant data upload.

	Bandwidth Savings	
Data Traffic Name	CoRE(%)	Others(%)
Linux Source	68	54
Email	11	3.6

Univ-HTTP	4.8	0
Univ-SN	8.3	0.3

Bandwidth savings comparison of CoRE and other models



Here we measure the total short-term redundancy detected in the traffic of the Linux source workload with relatively small chunk cache sizes varying from 32KB to 64MB. The result is shown above. We can see that from 512KB to 1MB the detected redundancy has significant increase but after that the increasing size yields diminishing returns. It is because that CoRE uses the chunk cache for short-term TRE, which aims at the bytes repeated within the traffic transmitted in a recent period that a small cache size is sufficient for caching. In our experiment we choose 4Bytes as the default chunk size, which from the figure we can see is a good trade-off between cost and saving when compared to 1 byte graph.

5.4 Mapping the results with problem statement and existing systems

Thus, we saw that with the use of end-to-end Cooperative Traffic redundancy solution where we are using two layers. The first-layer TRE module detects long-term redundancy by a prediction-based Chunk-Match approach like PACK. In the second-layer TRE module, both the server and client maintain a temporary small local chunk cache to store most recently transmitted and received chunks during their communication respectively. This model has led to significant reduction in bandwidth requirement. Through extensive trace-driven experiments, it is found that CoRE can efficiently capture both short-term and long-term redundancy, and can eliminate much more redundancy than other methods like PACK while incurring a low additional operation cost.

cloud providers often depend on data movement and transference. The smaller the data traffic in the cloud the more it reduces costs and release network bandwidth for multiple users and efficient delivery. Hence, Data deduplication not only maximizes storage capacity at the source without transferring the data to the network. Therefore, it frees up the bandwidth and helps sustain network performance, reliability, and development. Thus, this model is beneficial for cloud dependent firms like Netflix, Amazon etc which tend to serve large amount of data to users across the globe and require bandwidth and data storage optimization to reduce cost.

5.5 Discussions

TRE serves the applications following web service model which are dominated by the cost of egress bandwidth out of the cloud. The evaluation results show that different distributions of traffic redundancy at short-term and long-term time scales can cause different CPU costs for CoRE. Hence, adaptive solutions for CoRE are necessary for improving TRE efficiency and reducing the operation cost. Other types of cloud applications, like MapReduce, most bandwidth cost occurs in the datacenter network. According to the pricing of Microsoft Azure services , the data transfer within the Microsoft cloud is mostly free. Reducing their bandwidth cost inside the cloud does not save much expense for the cloud users, but may improve the networking performance and mitigate the bandwidth competition among cloud applications. We can investigate TRE for such applications in our future work.

6. Conclusion and Future Developments

The limitation on existing end-to-end sender side and receiver side TRE solutions for capturing redundancy in short-term and long-term have been shown which leads us to Cooperative end-to-end TRE CoRE where two-layer integration and several optimizations are used for TRE efficiency. The prediction window size adjustment and selection of layer of TRE is done by proposed several enhancement solutions. Compared to PACK, CoRE can get rid of data redundancy of both short-term and long-term redundancy much more efficiently. Our assessment results suggest that changing traffic redundancy distributions at short-term and long-term time scales might result in varying CPU costs for CoRE, and adaptive solutions for CoRE are required for boosting TRE efficiency and lowering operation costs.

Here we have pushed forward with the idea of TRE for an application following web service model that are being dominated by the cost of bandwidth out of clous as most of the cost occurs in the datacenter network for other cloud applications. In our future work we plan to investigate TRE further as it has not reached its full potential. One such example is that the data transfer within Amazon cloud is mostly free which reduces the bandwidth cost and improves the networking performance.

7. References

[1] Chhabra N, Bala M. A Comparative study of data deduplication strategies. In2018 First International Conference on Secure Cyber Computing and Communication (ICSCCC) 2018 Dec 15 (pp. 68-72). IEEE.

[2] Balasundaram A, Kothandaraman D, Kumar PS, Ashokkumar S. An
Approach to Secure Capacity Optimization in Cloud Computing using

Cryptographic Hash Function and Data De-duplication. In2020 3rd International Conference on Intelligent Sustainable Systems (ICISS) 2020 Dec 3 (pp.1256-1262). IEEE.

[3] Shin H, Koo D, Shin Y, Hur J. Privacy-preserving and updatable block-level data deduplication in cloud storage services. In2018 IEEE 11th International Conference on Cloud Computing (CLOUD) 2018 Jul 2 (pp. 392-400). IEEE.

[4] Pooranian Z, Chen KC, Yu CM, Conti M. RARE: Defeating side channels based on data-deduplication in cloud storage. InIEEE INFOCOM 2018-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) 2018 Apr 15 (pp. 444-449). IEEE.

[5] Leesakul W, Townend P, Xu J. Dynamic data deduplication in cloud storage. In2014 IEEE 8th International Symposium on Service Oriented System Engineering 2014 Apr 7 (pp. 320-325). IEEE.

[6] Tian G, Ma H, Xie Y, Liu Z. Randomized deduplication with ownership management and data sharing in cloud storage. Journal of Information Security and Applications. 2020 Apr 1;51:102432.

[7] Kim WB, Lee IY. Survey on data deduplication in cloud storage environments. Journal of Information Processing Systems. 2021;17(3):658-73.

[8] Harnik D, Pinkas B, Shulman-Peleg A. Side channels in cloud services: Deduplication in cloud storage. IEEE Security & Privacy. 2010 Dec 3;8(6):40-7.

[9] Fu Y, Xiao N, Jiang H, Hu G, Chen W. Application-aware big data deduplication in cloud environment. IEEE transactions on cloud computing. 2017 May 31;7(4):921-34.

[10] Yuan H, Chen X, Li J, Jiang T, Wang J, Deng RH. Secure cloud data deduplication with efficient re-encryption. IEEE Transactions on Services Computing. 2019 Oct 17;15(1):442-56.

[11] Yan Z, Zhang L, Wenxiu DI, Zheng Q. Heterogeneous data storage management with deduplication in cloud computing. IEEE Transactions on Big Data. 2017 May 4;5(3):393-407.

[12] Ng WK, Wen Y, Zhu H. Private data deduplication protocols in cloud storage. InProceedings of the 27th Annual ACM Symposium on Applied Computing 2012 Mar 26 (pp. 441-446).

[13] Reddy BT, Vaishnavi M, Lalitha M, Poojitha P, Kanthi VB. Privacy Preserving Data Deduplication in cloud using Advanced Encryption Standard. In2021 International Conference on Artificial Intelligence and Smart Systems (ICAIS) 2021 Mar 25 (pp. 1205-1210). IEEE.

[14] Wen Z, Luo J, Chen H, Meng J, Li X, Li J. A verifiable data deduplication scheme in cloud computing. In2014 International Conference on Intelligent Networking and Collaborative Systems 2014 Sep 10 (pp. 85-90). IEEE.

[15] Yingdan S, Huiba L. Data deduplication in cloud computing systems. In1st International Workshop on Cloud Computing and Information Security 2013 Nov (pp. 483-486). Atlantis Press.

[16]H. Lou and F. Farnoud, "Data Deduplication With Random Substitutions," in IEEE Transactions on Information Theory, vol. 68, no. 10, pp. 6941-6963, Oct.2022, doi: 10.1109/TIT.2022.3176778.

[17]Mr. M. A. R KUMAR, & Mrs. SRILATHA PULI. (2022). FINDING DATA DEDUPLICATION USING CLOUD. ymer, 21(5), 136–142. YMER || ISSN : 0044-0477.

[18]Lior Aronovich, Ron Asher, Danny Harnik, Michael Hirsch, Shmuel T. Klein, & Yair Toaff. (2016). Similarity based deduplication with small data chunks. Discrete Applied Mathematics, 212, 10–22, ISSN 0166-218X.

[19] Gagandeep Kaur, Mandeep Singh Devgan, "Data Deduplication Methods: A Review", International Journal of Information Technology and Computer Science(IJITCS), Vol.9, No.10, pp.29-36, 2017. DOI: 10.5815/ijitcs.2017.10.03

[20]J. Xiong, Y. Zhang, S. Tang, X. Liu and Z. Yao, "Secure Encrypted Data With Authorized Deduplication in Cloud," in IEEE Access, vol. 7, pp. 75090-75104,2019, doi: 10.1109/ACCESS.2019.2920998.

[21]Md. Jareena Begum, & B. Haritha. (2020). Data Deduplication Strategies in Cloud Computing. International Journal of Innovative Science and Research Technology, 5, 734–738.

[22] M. Adithya, Dr. B. Shanthini,'Security Analysis and Preserving Block-Level Data DE-duplication in Cloud Storage Services by ',Journal of trends in Computer Science and Smart technology (TCSST) (2020) Vol.02/ No. 02
Pages: 120-126

[23]X. Liu, T. Lu, X. He, X. Yang and S. Niu, "Verifiable Attribute-Based Keyword Search Over Encrypted Cloud Data Supporting Data Deduplication," in IEEE Access, vol. 8, pp. 52062-52074, 2020, doi: 10.1109/ACCESS.2020.2980627.

[24]Q. He et al., "Research on Data Routing Strategy of Deduplication in Cloud Environment," in IEEE Access, vol. 10, pp. 9529-9542, 2022, doi:
10.1109/ACCESS.2021.3139757.

[25]W. Shen, Y. Su and R. Hao, "Lightweight Cloud Storage Auditing With Deduplication Supporting Strong Privacy Protection," in IEEE Access, vol. 8, pp.44359-44372, 2020, doi: 10.1109/ACCESS.2020.2977721.

[26]Stanek, J., Sorniotti, A., Androulaki, E., Kencl, L. (2014). A Secure Data Deduplication Scheme for Cloud Storage. In: Christin, N., Safavi-Naini, R. (eds) Financial Cryptography and Data Security. FC 2014. Lecture Notes in Computer Science(), vol 8437. Springer, Berlin, Heidelberg.
https://doi.org/10.1007/978-3-662-45472-5_8

[27]Reddy, B. T., & Rao, M. C. S. (2018). Filter based data deduplication in cloud storage using dynamic perfect hash functions. International Journal of Simulation Systems, Science & Technology.

[28]Priteshkumar Prajapati, Parth Shah,A Review on Secure Data Deduplication: Cloud Storage Security Issue,Journal of King Saud University - Computer and Information Sciences,Volume 34, Issue 7,2022,Pages 3996-4007,ISSN 1319-1578,

[29]Xuewei Ma, Wenyuan Yang, Yuesheng Zhu, Zhiqiang Bai,'A Secure and Efficient Data Deduplication Scheme with Dynamic Ownership Management in Cloud Computing'.
2208.09030v3 Wed, 31 Aug 2022 15:47:52 UTC

[30]P. G., S., R. K., N., Menon, V.G. et al. A secure data deduplication system for integrated cloud-edge networks. J Cloud Comp 9, 61 (2020).
https://doi.org/10.1186/s13677-020-00214-6

Research article must be from reputed journal with TR- Impact Factor only

Documentation without Plagiarism is most important, Turnitin software will be used to evaluate. (Making false sentences, paraphrasing, changing characters in the document, inserting images as paragraph for the purpose of reducing plagiarism will lead to marking ZERO).

Any other related information, you want to add.

VIT Vellore Institute of Technology (Deemed to be University under section 3 of UGC Act, 1956)	**B.Tech (Information Technology)** **FALL 2022 – 2023** **ITE 3007 : Cloud Computing and Virtualization** **10.11.2022 – 14.11.2022**
Title:	
Team Name	

Project Team

S.No	**Register Number**	**Student Name**	**Signature**	**Guided By**
				Dr. Nadesh R.K

Team Member(s) Contribution and Performance Assessment

Components	**Student 1**	**Student 2**	**Student 3**
Implementation & Results -(20)			
Contributed fair share to the team project -(05)			
Documentation without Plagiarism - **(20)**			
Q & A - (05)			

Student Feedback(Student Experience in this Course Project)	**Evaluator Comments**

Name & Signature of the Evaluator **(Dr.Nadesh R.K)**