DOI: http://dx.doi.org/10.26483/ijarcs.v9i2.5318

ISSN No. 0976-5697

Volume 9, No. 2, March-April 2018

International Journal of Advanced Research in Computer Science

REVIEW ARTICLE

Available Online at www.ijarcs.info

AN ANALYSIS AND COMPARISON OF DATA DEDUPLICATION APPROACHES TO EFFICIENT STORAGE

Hema S
Research Scholar,
Department of Computer Applications,
Govt. Arts College (Autonomous), Salem-7 India

Dr.Kangaiammal A
Assistant Professor,
Department of Computer Applications, Govt. Arts College
(Autonomous), Salem-7 India

Abstract: When the volume of digital data increases, it requires more storage space and efficient technique to effectively handle these huge data. While handling huge amount of data, duplicacy is unavoidable. Backup of duplicate data increases the storage time and consumes more resources. Data deduplication is one of the essentially used techniques in storage environment that employs various techniques to manage duplicate data efficiently, which removes redundant data; minimizes bandwidth and optimizes storage utilization. This paper reviews various deduplication techniques and evaluation process used to measure their performance. Also compares and correlates security related to data deduplication techniques.

Keywords: Deduplication, Chunking, Convergent Encryption, Proof-of-Ownership.

I. INTRODUCTION

In the current digital world enormous amount of digital data being produced from different devices, backup of this huge amount of data has become one of the most difficult and important tasks in mass storage systems. Thus, using an efficient optimization technique can minimize storage management issues. Deduplication is one such a storage optimization technique that minimizes the transmission and storage of duplicate data.

Data de-duplication often called "intelligent compression" or "single-instance storage" is a technique which eliminates duplicate data by storing only a single copy of each file or block. Data Deduplication has become a popular storage optimization technique necessary for secondary in addition to widely adapted in primary storage and cloud like large storage areas in order to reduce storage space. Deduplication can either be file-level [24] or block-level [25]. Block level approach breaks the file into fixed-size or variable-size blocks [23]. To process each block of data the hash algorithm, which is used to generate unique hash value namely MD5/SHA1/SHA256/SHA512. The generated hash value is then, compared with an index of existing hash values. If the hash value of data block is already in an index, the data block need not be stored and it is considered as duplicate. Otherwise, the data block is considered a new one and it has to be stored. Figure1 shows an overview of Data Deduplication.

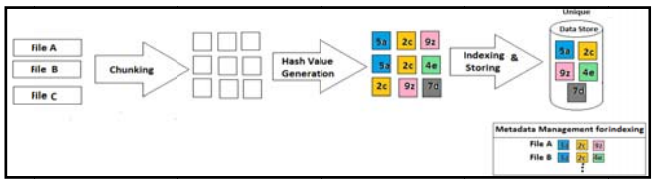

Figure 1: Overview of Data Deduplication

II. PROCEDURES USED IN DEDUPLICATION

Following are some of the mechanisms used in de-duplication to improve the performance of deduplication process.

A) Hash-Based Algorithms

Hash based algorithms can be used to uniquely identify chunks of data. Most commonly used algorithms are Secure Hash Algorithm (SHA-1) [26] and Message-Digest Algorithm (MD5) [27]. Both SHA-1(160-bit) and MD5 (128 bit) designed for cryptographic purpose which divides data into chunks and generate unique hash key for each chunk. SHA-1 has very less chances of incident of data collision. While compared with SHA-1, the MD5 algorithm is weaker and less secure. But MD5 is faster than SHA-1.

B) The Basic Sliding Window algorithm(BSW)

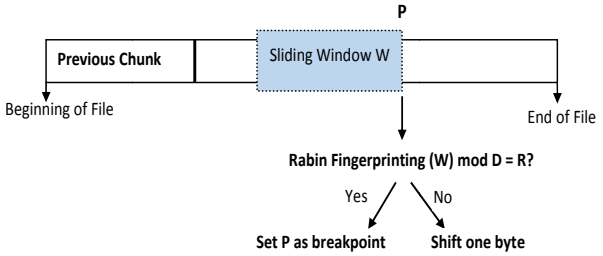

Figure 2: Basic Sliding Window Algorithm

One of the most important chunking algorithm named as Basic Sliding Window (BSW) [28] gives best de-duplication ratio than previous techniques like k-gram and 0 mod p algorithms. The BSW algorithm mainly uses three main factors to segment the data stream. i) fixed size window (W), ii) integer divisor (D), and iii) integer remainder (R), where R < D. A problem arises in BSW

algorithm is that it may create too small chunk or too large chunk because of it has very poor control over the chunk size. Figure 2 shows Basic Sliding Window Algorithm.

A fixed-size window (W) moves one byte at a time from the beginning of the file to end of the file.

(1) At every location P, it applies Rabin Fingerprinting algorithm to calculate a hash value (h) for the current window content.
(2) If h mod D = R, then that location is the break point for chunk boundary. Sliding window W starts at the break point position P and repeat the task.
(3) If h mod D ≠ R, the sliding window W keeps shifting one byte and continue the process.

C) Two Threshold Two Divisor Algorithm (TTTD)

TTTD[28] algorithm uses everything was similar to the BSW algorithm but it utilize four parameters, the maximum threshold, the minimum threshold, the main divisor, and the secondary or backup divisor. By using minimum threshold and maximum threshold parameters, this approach give a solution of the problem happened in BSW algorithm In TTTD, the main divisor works similar to BSW algorithm and additionally it use secondary divisor to find breakpoint for chunks in case the main divisor fails to find any breakpoint. TTTD performs much better than all the existing algorithms, and enhance the performance of applications that use content based chunking.

D) Content or Application Aware-Based Chunking

Content Aware chunking method understands the format of the files, can give good deduplication ratio than the fixed-size and variable-size chunking methods. Hence, content aware chunking method compares the incoming file with its index table, which contains existing file information to identify similarities and relationship. Moreover, this method is known about the file format, so that it selects a similar format file as a reference file from an index table in order to make a comparison. Finally, it computes difference (delta) between the incoming file and its reference file, then stores this delta value rather than to store the whole incoming file.

.

E) Convergent Encryption

A technique recommended to perform deduplication on encrypted data is known as Convergent Encryption [20]. This technique uses an encryption key which is derived from the data content itself to perform encryption on the data and hence, it will generate same identical cipher text for two identical copies of the files. Moreover, this scheme ensures the privacy of user and at the same time it will perform deduplication, unfortunately it is vulnerable to dictionary attacks.

F) Proof-of-ownership

Specifically client-side deduplication allows an adversary who knows a little information about the file can convince the server, as a result the server permits adversary to access the entire file. To overcome such kind of problem, an approach proposed is known as Proof-of-Ownership (PoWs) [5,21], which allows a client to make confirmation to the server that he is the owner of the entire file not just part of the file.

III. RELATED WORK

Xia W *et al* [1] surveyed the background and various methodologies used in data deduplication, which saves more storage space in storage systems. This approach eliminates redundant data at the file or chunk level and identifies duplicate data using fingerprint value. In this comprehensive study, they explained about the differences between data deduplication and traditional data compression. They have classified the data deduplication into six general categories, and created taxonomy for each category, which gives the development of the technology over the years in addition to pros and cons of the state-of-the-art approaches to all stages of data deduplication. It provides an in-depth study of the new and emerging areas for deduplication, such as delta compression, restore, garbage collection, security, reliability, etc. Further they have discussed about publicly available open-source projects, datasets, and traces for the data deduplication research community. Finally, they have summarized open problems and research challenges in front of data deduplication research.

Z. Yan et ***al.*** **[2]** proposed a deduplication scheme to store encrypted form of data in the cloud . This paper motive is to protect the privacy of data holders. However, encrypted data establish new trouble for cloud data deduplication, which becomes critical for big data storage and processing in cloud. It also suffers from security weakness. Hence, common deduplication schemes may not be used for encrypted data. To overcome this problem they proposed a method based on data ownership challenge and Proxy Re-Encryption (PRE) to deal with deduplicate encrypted data which is stored in cloud. They applied Elliptic Curve Cryptography (ECC) to authenticate data ownership with the support of an authorized party. This method can support data sharing with deduplication and achieved excellent performance even when the data holders are offline. Author evaluated the performance of the scheme and the test results showed that encrypted data can be safely accessed only by the authorized data owner who owns the symmetric keys required for data decryption.

Z. Yan *et.al* [3] proposed a proxy re-encryption (PRE) scheme to store encrypted data in cloud. The cloud users upload their personal data to a Cloud Service Provider (CSP) and permit it to maintain these data. To protect the

privacy of users data outsourced, it is stored in an encrypted form. In this paper, they proposed a scheme based on proxy re-encryption which is used to reduce computation complexity and cost. . This scheme can flexibly support data updation and deduplication. It does not interrupt with the privacy of data owners. The efficiency of PRE and the effectiveness of encryption / decryption is tested by experimentation. As the scheme cannot be tested straightforwardly in the cloud, MYSQL database is utilized to store data files and related information. Output of the scheme upon extensive performance analysis and experimentation demonstrates it as an efficient security model.

Puzio et.al in [4] designed a system, based on Convergent Encryption Technique. The design guarantees both block level deduplication and confidentiality of data. An additional encryption layer is introduced in this system to protect the confidentiality of the data and to avoid well known attacks against convergent encryption. This system recommended an efficient key management task performed by metadata manager (MM).MM contains a small database and a linked list in order to maintain file ownerships, file composition and avoid duplicate data storage. MM utilized file table – to maintain file Meta data, pointer table – to handle storage and signature table – to keep signature Meta data. Finally, it is concluded that the solution avoids curious cloud providers from inferring the user's stored data.

Halevi, Shai, *et al*. [5] introduced the new concept known as proofs-of ownership (PoWs). Deduplication technique stores only a single copy of reoccurring data. Hash value is computed for each and every data needed to be stored in cloud. If a data is identified as duplicate using hash value, it is not stored, instead a pointer is generated for that duplicate data thus saves storage and cost.This paper mainly focused on client-side deduplication and it is applied at the file level. In client side approach, hash values of data are computed by client and forwarded for duplicate check. In this process, attacks to hack the hash value are identified. By knowing hash value, hacker may convince storage service to access the entire file of other users. To secure data and to overcome such attacks, a concept known as proofs-of ownership (PoWs) is introduced. PoW is a protocol worked between two parties a prover and verifier. First the verifier generates a (shorter) verification code "v" , a prover needs to calculate verification code "v" and send it to verifier in order to prove his authenticity that he is the owner of the file.

In [6] Bellare *et.al* designed a system name it as DupLESS, which supports deduplicated storage and also resists brute-force attacks. oblivious PRF protocol is used by key-server to provide key to the client for encryption. Secrete key is safe with key server while public key for encryption is shared among the client is established using OPRF protocol. The proof is evident that the DupLESS model enhanced the performance and reduced the storage space of encrypted data.

Ng, Wee Keong *et.al* in [7] introduced a new concept known as private data deduplication protocol for cloud storages. By using this protocol, a client provides proof of ownership to the server by disclosing just the summary string of the data without informing any additional information. They proved that the suggested private data deduplication protocol is provably secure in the framework which is simulation based.

Venish. A et.al in [8] discussed different chunking algorithms and compared its performance. Data has broken into small pieces called "chunks" and each chunk is recognized by hash value. These hash values are used to identify presence of duplicate data. According to this paper, file level chunking technique is efficient for smaller size files, but it is not suitable for larger size files. This paper addressed the boundary shifting problem occurring in fixed size chunking method and this problem can be overcome by using variable size chunking method. A variety of variable chunking methods are also discussed in this paper. Finally, they pointed out that content aware chunking method gives good results for multimedia files like videos, audio, images and decrease the space utilization.

In [9] Wen Xia *et al.* presented a new scheme named as SiLo, a similarity-locality based deduplication system in order to attain high deduplication throughput and reduce RAM overhead than existing approaches. By using similarity algorithm, SiLo combine small correlated files and splits larger files to reduce RAM utilization while the locality algorithm used to remove most of the duplicate data to achieve higher throughput. Under various workload conditions, an experimental result proves that SiLo scheme is very effective.

In [10] Kiran Srinivasan *et.al* proposed an inline deduplication system name it as iDedup. The design, implementation and evaluation of iDedup illustrated in this paper. In inline deduplication algorithm they derived two key insights: i) spatial locality – to perform deduplication only sequences of disk blocks, thus it reduce fragmentation; and ii) temporal locality – allows maintaining dedup-metadata in an in-memory cache in order to avoid additional IOs. Ultimately, this system reduces resource overheads (CPU and Memory) and thus iDedup appropriate for latency sensitive workloads.

Xia, Wen, et.al [11] presented a new scheme called Deduplication-Aware resemblance detection and Elimination Scheme (DARE). DARE is a blend of data deduplication and delta compression process which helps in accomplishing high data reduction efficiency at low overhead. In DARE, first stage is Duplicate-Adjacency based Resemblance Detection (DupAdj) used to identify similar data chunks followed by "improved super-feature approach" which further enhanced the performance of resemblance detection. The experimental results on backup datasets showed that system achieves an additional data reduction with low overhead. Data restoration performance

of deduplication is improved by Delta compression scheme widening the logical space for the restoration of cache.

In [12] M. Lillibridge, et al presented a technique known as Sparse indexing that applied sampling and locality concepts for large-scale backup storage. Chunk-based deduplication scheme needs a full chunk index to find duplicate, unfortunately, it is difficult to keep full index in RAM and thus it creates chunk-lookup disk bottleneck problem. In this approach, the above said issue has been solved by using locality principle. Segment creation, the key factor of stream deduplication, is used in this technique. sampling and sparse indexing is also used in addition to identify similar segments. Previously stored small numbers of similar segments have been chosen and deduplicate each segment against this chosen few segments whereby it avoids keeping full chunk index in RAM. This approach needs only few searches for each segment so that chunk-lookup disk bottleneck problem was avoided and simultaneously it improved the deduplication process tremendously.

D. Bhagwat et al [13] introduced a new technique called Extreme Binning for Scalable and parallel deduplication, which is Suitable for non-traditional backup workloads. Instead of locality principle, Extreme Binning utilized file similarity principle. This approach divides the chunk index into two tiers. First tier is placed in RAM and the second tier is stored in disk. Extreme Binning system avoids chunk lookup access for each chunk in a file and it performs only one disk access for entire file. Thus, it reduced the disk bottle neck problem and gives reasonable throughput.

Fanglu Guo et al in [14] mainly focused on system design instead of enhancing the duplicate detection algorithms. The proposed single-node deduplication system use progressive sampled indexing in memory which offers high scalability and efficient memory usage. Resource management and reclamation problem can be solved by using mark-and-sweep mechanism, which reduced disk accesses and accomplished near-optimal scalability. Furthermore, they have proposed an asynchronous interface to the server back-end, in order to forward data into server at high-enough rates. Finally, their prototype implementation oversees dedupe challenges as well as used to attain high backup, restore throughput, and gives efficient deduplication.

Kaiser J et al in [15] described a new deduplication system for parallel processing of numerous backup streams. This approach used sorted order index for fingerprints instead of locality, so that every streams have right to access the same index region simultaneously. The implementation results proved that this approach guarantees a disk-friendly access pattern by consuming much lesser memory than Data Domain deduplication file system and Sparse Indexing.

In [16] Xia, Wen et al put forward Ddelta, a deduplication inspired fast delta compression method to speed up the delta encoding and decoding processes. This paper employed Gear-based chunking algorithm to quickly carry out chunking process and Spooky based fingerprinting in order to speed up duplicate identification. Moreover, greedy byte-wise scanning approach is utilized to discover more redundancy. Experimental outcome showed that Ddelta accomplishes amazing encoding and decoding when compared to traditional delta-compression approaches like Xdelta and Zdelta.

Wang J et al [17] suggested I -sieve, a high performance inline deduplication system, which aims to build a small scale storage system. This system exploited a lightweight indexing table and a two-level cache structure using SSD in order to enhance deduplication performance. They executed I-sieve prototype and simulated common I/O features using IOmeter tool. They evaluated it with three different models, the traditional iSCSI (RAW for short) model, cache module (DC), and bloom filter modules (DCB).When compared with cache module(DC), bloom filter module (DCB) gives better performance, which speedup block retrieval. Finally, the evaluation results illustrated that I-sieve has brilliant deduplication ratio and foreground performance.

In [18] Liu J et al. developed the first single-server scheme which supports client-side encryption in order to protect user's privacy. This cross-user deduplication scheme employed a PAKE (password authenticated key exchange) protocol that allows two clients confidentially compare their secrets and exchange their encryption key. If two parties upload the same file, the duplicate was identified by server and it stored only one copy. Furthermore, this strategy prevents online brute-force attacks and it does not need any additional independent servers. Finally, implementation result showed the effectiveness and the efficiency of their strategy.

Xu J et al. in [19] Presented a secure client-side deduplication (CSD) system aims to protect the privacy of users sensitive data. This system secures data not only from outside adversaries but also from curious cloud storage server. They enhanced the convergent encryption method and permitted one-time leakage of a target file before their system begins to execute. Very low min-entropy sensitive files can apply this approach for deduplication due to its one-time leakage.

Table 1 shows the comparison of different secure related data deduplication techniques.

Table 1 Comparison of secure related data deduplication techniques

Scheme	Hash Function/ AES keys	Method of Deduplication	Exclusive Feature
Secure deduplication of encrypted data without additional independent servers	SHA-256	File and Block level approach	• Based on PAKE (Password authenticated Key Exchange) Protocol.
Deduplication on Encrypted Big Data in Cloud	SHA-1/AES keys (128, 196,256)bits	Block level approach	• Use Proxy Re-Encryption (PRE) scheme to manage encrypted data along with deduplication. • Use Eillptic Curve Cryptography (ECC) to verify data ownership.
Encrypted Data Management with Deduplication in Cloud Computing	SHA-1/ AES keys (128, 196,256)bits	Block level approach	• Based on attribute-based encryption (ABE) to deduplicate encrypted data.
Secure Deduplication with Encrypted Data for Cloud Storage	SHA-256/ AES key size 256 bits	Block level approach	• It combines features of both deduplication and convergent encryption. • An additional server was defined which prevent attacks against Convergent Encryption. • Metadata manager (MM) responsible for deduplication and key management operations
Proofs of Ownership(PoW) Technique used in Remote Storage Systems	SHA-256	Block level approach	• Design the concept of proof-of-ownership • Solutions based on Merkle trees.
Client-side Deduplication of Encrypted Data in Cloud Storage	SHA-256	Block level approach	• Utilize enhanced convergent encryption method • Permits one-time leakage of a target file
DupLESS: Server-Aided Encryption for Deduplicated Storage	SHA-256	Block level approach	• Resist Brute force attack. • By using OPRF protocol the client can get encryption key from key server(KS) and also it gives guarantee that OPRF does not permit KS to access client inputs as well as it does not permit clients to learn about the key.

V. CONCLUSION

This paper has surveyed various deduplication schemes and has compared deduplication techniques and presents their relationship. Among various methods, some approaches handle encrypted data and mostly works are on the basis of convergent encryption method. It is observed from this survey, the file level approach is relatively easy to understand but not efficient for large size files while variable-size approach has high redundancy detection ability than fixed size approach. Further, this survey explores how the content aware chunking method gives better deduplication ratio than the fixed-size and variable-size chunking methods. Finally, this paper discusses some efficient indexing techniques such as sparse indexing, similarity and locality based approaches etc. Such indexing techniques support duplicate identification and also save memory space.

VI. REFERENCES

[1] Xia W, Jiang H, Feng D, Douglis F, Shilane P, Hua Y, Fu M, Zhang Y, Zhou Y. A comprehensive study of the past, present, and future of data deduplication. Proceedings of the IEEE. 2016 Sep;104(9):1681-710.

[2] Z. Yan, W. Ding, X. Yu, H. Zhu and R. H. Deng, "Deduplication on Encrypted Big Data in Cloud," in *IEEE Transactions on Big Data*, vol. 2, no. 2, pp. 138-150, June 1 2016.

[3] Z. Yan, M. Wang, Y. Li and A. V. Vasilakos, "Encrypted Data Management with Deduplication in Cloud Computing," in *IEEE Cloud Computing*, vol. 3, no. 2, pp. 28-35,Mar.-Apr.2016.

[4] Puzio, Pasquale, Refik Molva, Melek Önen, and Sergio Loureiro."ClouDedup:Secure Deduplication with Encrypted Data for Cloud Storage." .In Cloud Computing Technology and Science(CloudCom),2013 IEEE 5th International Conference on (Volume:1) p.363 – 370.

[5] Shai Halevi , Danny Harnik , Benny Pinkas , Alexandra Shulman-Peleg, Proofs of ownership in remote storage systems, Proceedings of the 18th ACM conference on Computer and communications security, October 17-21, 2011, Chicago, Illinois, USA.

[6] Bellare, Mihir, Sriram Keelveedhi, and Thomas Ristenpart. "Dupless: Server-aided encryption for deduplicated storage." Proceedings of the 22nd USENIX conference on security. USENIX Association, 2013.

[7] Ng, Wee Keong, Yonggang Wen, and Huafei Zhu. "Private data deduplication protocols in cloud storage." Proceedings of the 27th Annual ACM Symposium on Applied Computing. ACM, 2012.

[8] Venish, A., and K. Siva Sankar. "Study of Chunking Algorithm in Data Deduplication." Proceedings of the International Conference on Soft Computing Systems. Springer, New Delhi, 2016.

[9] Wen Xia , Hong Jiang , Dan Feng , Yu Hua, SiLo: a similarity-locality based near-exact deduplication scheme with low RAM overhead and high throughput, Proceedings of the 2011 USENIX conference on USENIX annual technical conference, p.26-28, June 15-17, 2011.

[10] Srinivasan, Kiran, Timothy Bisson, Garth R. Goodson, and Kaladhar Voruganti. "iDedup: latency-aware, inline data deduplication for primary storage." In *FAST*, vol. 12, pp. 1-14. 2012.

[11] Xia W, Jiang H, Feng D, Tian L. DARE: A deduplication-aware resemblance detection and elimination scheme for data reduction with low overheads. IEEE Transactions on Computers. 2016 Jun 1;65(6):1692-705.-1705.

[12] M. Lillibridge, K. Eshghi, D. Bhagwat et al., "Sparse indexing: Large scale, inline deduplication using sampling and locality." in Proceedings of the 7th USENIX Conference on File and Storage Technologies (FAST'09), vol. 9. San Jose, CA: USENIX Association, February 2009, pp. 111–123.

[13] D. Bhagwat, K. Eshghi, D. D. Long et al., "Extreme binning: Scalable, parallel deduplication for chunk-based file backup," in Proceedings of IEEE International Symposium on Modeling, Analysis & Simulation of Computer and Telecommunication Systems (MASCOTS'09). London, UK: IEEE Computer Society Press, September 2009, pp. 1–9.

[14] Fanglu Guo and Petros Efstathopoulos. 2011. Building a high-performance deduplication system. In Proceedings of the USENIX Annual Technical Conference (ATC). USENIX, Berkeley, CA, 1–14.

[15] Kaiser J, Süß T, Nagel L, Brinkmann A. Sorted deduplication: How to process thousands of backup streams. InMass Storage Systems and Technologies (MSST), 2016 32nd Symposium on 2016 May 2 (pp. 1-14). IEEE.

[16] Xia, Wen, Hong Jiang, Dan Feng, Lei Tian, Min Fu, and Yukun Zhou. "Ddelta: A deduplication-inspired fast delta compression approach." Performance Evaluation 79 (2014): 258-272.

[17] Wang J, Zhao Z, Xu Z, Zhang H, Li L, Guo Y. I-sieve: An inline high performance deduplication system used in cloud storage. Tsinghua Science and Technology. 2015 Feb;20(1):17-27.

[18] Liu, Jian, N. Asokan, and Benny Pinkas. "Secure deduplication of encrypted data without additional independent servers." In Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security, pp. 874-885. ACM, 2015.

[19] Xu, J., Chang, E.C., Zhou, J.: Weak leakage-resilient client-side deduplication of encrypted data in cloud storage. In: 8th ACM SIGSAC Symposium, pp. 195–206.

[20] J. Douceur, A. Adya, W. Bolosky, D. Simon, and M. Theimer. "Reclaiming space from duplicate files in a serverless distributed file system. In Distributed Computing Systems", 2002. Proceedings. 22nd International Conference on, pages 617{624. IEEE, 2002.

[21] Di Pietro R, Sorniotti A. Boosting efficiency and security in proof of ownership for deduplication. InProceedings of the 7th ACM Symposium on Information, Computer and Communications Security 2012 May 2 (pp. 81-82). ACM.

[22] Mogul J, Douglis F, Feldmann A, Krishnamurthy B (1997) Potential benefits of delta encoding and data compression for HTTP. In: Proceedings of ACM SIGCOMM'97 conference, pp 181– 194, Sept 1997.

[23] Rabin M.O., "Fingerprinting by random polynomials. Center for Research in Computing Technology", Aiken Computation Laboratory, Univ., 1981.

[24] CoxL. P., MurrayC. D., and NobleB. D., "Pastiche: Making backup cheap and easy", ACM SIGOPS Operating Systems Review, vol. 36, no. SI pp. 285–298, 2002.

[25] Wilcox-O'HearnZ. and WarnerB., "Tahoe: the least-authority file system", Proceedings of the 4th ACM International workshop on Storage Security and Survivability.pp.21–26, 2008.

[26] National Institute of Standards and Technology, FIPSPUB 180-1: Secure hash Standards, Technical Report, 1995.

[27] R. Rivest, The md5 message-digest algorithm, http://www.ietf.org/rfc/rfc1321.txt, 1992.

[28] K. Eshghi and H. K. Tang, A framework for analysing and improving content-based chunking algorithms, Tech. Rep. HPL–2005–30(RI),2005.

