A Parallel Distributed Way to Secure vulnerable Information, and its Privacy

Hadisur Rahman[1], Md. Mainuddin[2], Md. Meraj Ali[3], Md. Sumon Rony[4]
Department of Computer Science and Engineering, Varendra University[1, 3, 4]
Florida State University[2]
hudacse6@gmail.com, mainuddincse@gmail.com, meraj09034@gmail.com, sumoncse2395@gmail.com

Abstract — **Many researchers thought about privacy protection in data processing as a cardinal topic. That is, the way to make sure the users' privacy whereas their information is strip-mined. Associated with this subject is data processing for the cover of security and privacy. If we have a tendency to not solve the privacy issue, data processing can become an uncomplimentary term to the social public. Some of the folks took into account the matter of data integrity assessment to be very necessary. We solely have a predisposition to quote their observations, "Data mining algorithms are often applied to information that is purposely changed from their original version, so as to mislead the recipients of the info or to counter privacy, and security threats. Such modifications will distort, to an unknown extent, the information contained within the original information. As a result, one among the challenges facing researchers is that the development of measures is not solely to judge the information integrity of a group of information. However, additional measures to judge the information integrity of individual patterns. The data integrity assessment presented many challenges." The primary challenge needs the event of economical algorithms, and information structures to judge the information integrity of a group of information. The second challenge is to evolve an algorithm to secure the impact that the modification of information values has on a discovered pattern's applied mathematical significance, though it would be unfeasible to develop a worldwide measure for all data processing algorithms.**

Keywords— Data Privacy, and Data Security; Cloud Privacy, Algorithm, Collision; Hash Operation.

I. INTRODUCTION

Security threats [1, 2, 9] are among the best threats on social forum, IT sector, and large information sector [9] makes it a lot more advanced and tough to implement an infallible security framework. The network provides an access path for each within and out of doors attacks that, makes it a key access purpose for any form of security threats. Securing information isn't a simple task because it is advanced, it needs careful configuration, and it's subject to human errors. The information ought to be open, however, with the correct security framework, and testing security threats during this reasonably posh information infrastructure with immense, and unstructured information is an on-going method. Hence, we want a security testing model that is capable of spotting any unauthenticated intrusion directly. This paper represents a way to check distributed environments against attacks on information integrity.

In this age, massive of huge information each little or big, previous or new information is taken into account to be of much importance. Numerous analytical systems use this information and manufacture results which may be used for various functions. So, there are no single rather than calculable sources that contribute to the information assortment. The information assortment is going on manually and online through numerous new and precious resources. Information is often created, derived and touched around. Existing distributed computing networks and numerous cloud infrastructures have deployed their own security mechanisms. However, with a fast increase in information amount and its selection, security threats have additionally enlarged. This can be the main reason why enterprise and huge organizations use their own personal information storage rather than victimization public clouds [3]. Privacy preservation may be or must be in Social Networking, E-commerce [4], and repair Orientation, and Cloud.

II. METHODOLOGY AND ALGORITHM

A. *How Much Information we have a tendency to Used?*

- Way back Machine has 2 PB +20 TB/Month (2006)
- Google processes 20 PB a day (2008)
- “all words ever spoke by human being” ~ 5EB
- NOAA has ~ climate data (2007)
- CERN, s LHC will generate 15 PB a year (2008)

B. *Working on the Algorithm’s*

- RSA
- Elgamal
- Hash, and
- Parallel Pollard’s Rho Algorithm

C. *Using Software package*

- Apache Hadoop with its well-known frameworks
    a. Map Reduce
    b. Apache Spark
- Weka

D. *Pc Configuration wherever the info is tested.*

a) 12GB RAM
b) 120GB SSD
c) Core I3(3.6GHZ) processor
d) Window 10 pro (64 bit)

E. *Data Storage in a well-known Distributed System*

In a distributed system, [5] the info to be held on is split into fragments, and these fragments are distributed across many nodes reckoning on the need for information thereon node. This method is named information Fragmentation. Some information or fragments of information might also be replicated judiciously so as to extend dependableness, accessibility, and hence, the performance of a distributed system.

F. *Prevailing System*

The distributed system may be an assortment of undisciplined nodes, as shown below [Fig .1] every of that store information fragments (B1, B2, etc.). The nodes are connected via a network that may well be a native space network or a large area network reckoning that the nodes are settled within the same local space or completely different areas severally.

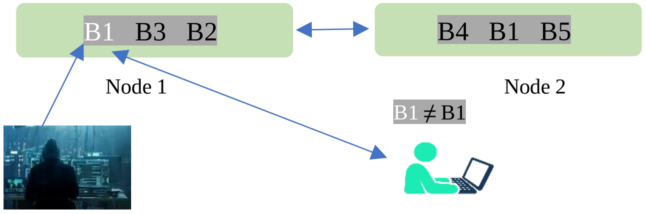

Fig: 1. Distributed system with replicated data fragments

G. *Data Replication has the subsequent benefits*

- Unmitigated duplicate data is avoided since the unsuccessful duplicate data is replaced by another duplicate data. In different words, even though a website fails, however, it has its information replicated on another site, so that information would still be accessible. This improves each accessibility and dependableness.
- Placing a replica of information among adjacent proximity to the method victimization can cut back the interval of that, data resulting in increased performance.

H. *Maintaining consistency: Two-part Commit Protocol*

The drawback of getting multiple copies of equivalent information is maintaining consistency in the replicated information. This implies that if one copy of the information is modified, the corresponding modification should be created to any or all copies of the info. So as to attain this tight consistency, we have a tendency to use the 2 Phases Commit Protocol (2PC)

In the following example, the organizer needs to update a duplicate of some information within the info whereas P1 and P2 are the participant nodes that additionally contain replicas of equivalent information.

- *Phase 1: Vote Part* – The organizer sends a vote request (PREPARE T), when adding the request to its log, to any or all the participants with the replicated copy of the info to be updated. The participants either comply with commit (READY T) or abort (NOT) the dealing (which is that the modification of data).
- Phase 2: Decision Part – The organizer sends a vote request (PREPARE T), when adding the request to its log, to any or to all the participants with the replicated copy of the info to be updated shown [Fig. 2. 2(a)]. The participants either comply with commit (READY T) or abort (NOT) the dealing (which is that the modification of data) shown [Fig. 2. 2(b)].

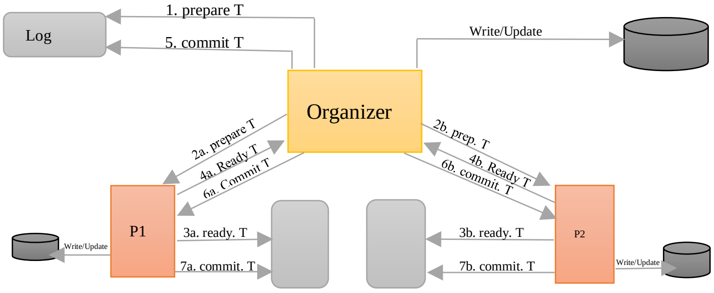

Fig. 2. 2(a) 2PC with a committed transaction

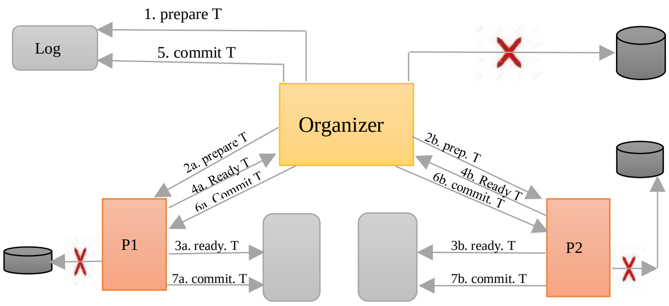

Fig. 2. 2(b) 2PC with an aborted transaction

I. *Problem: Attack on Integrity Information*

Suppose an assaulter [11] makes a felonious modification to a replica of some information in Node one [Fig.1]. For the user, someone requests access to information from B1, the distributed system may be a coherent system and he's unaware that a copy of the information is formed out there with him. The user may well be given access to information on B1 of either Node one or Node Two. By dint of the attacker, the content of B1 in Node one is completely different from the content of B1 in Node two. Since the user has no approach of supportive the integrity of information, he won't notice that the information has been modified by an unauthorized person and he could continue to utilize the malicious data B1.

J. *Solution: Hash Operation*

To overcome the matter, we will add a worldwide Hash [6] Store to the distributed system that consists of hash operating values of all the data/fragments hold on, as shown in [Fig. 3.] A hash operates may be a unidirectional operation that takes as input a message of any arbitrary length and returns a hard and fast length output that is named the hash or message digest of the input message. This hash worth is appended with the message and recomputed by the receiver of the message so as to see for any transmission errors or to discover attacks on the integrity of information. Even a tiny low modification of a message causes the hash of that message to vary and this property of hash functions makes them appropriate for making certain message integrity. And thus, if the recomputed hash worth doesn't match the appended hash worth, the receiver will conclude that the content of the message has been modified. Adding an International Hash Store to a Distributed System can create the hash values of all the information and data fragments, out there to the genuine users. During this approach whenever a user accesses any information, he/she will ensure the integrity of that information by hard the hash of the info and compare it with the hash worth retrieved from the hash Store. The computed hash worth won't match the retrieved hash worth if the info has been changed illicitly by an assaulter.

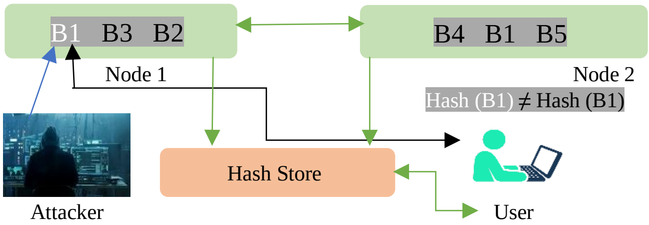

Fig. 3. Distributed Network with Hash Store

K. *Changed the two-part Commit Protocol*

The reckoning of hash worth for every data/data fragment is done at the top of the Two-part Commit Protocol, as shown in [Fig. 4] each time a piece of knowledge is updated or intercalary to the Distributed System. These hash values will then be held on within the international Hash Store from wherever it's accessible to any or all genuine users.

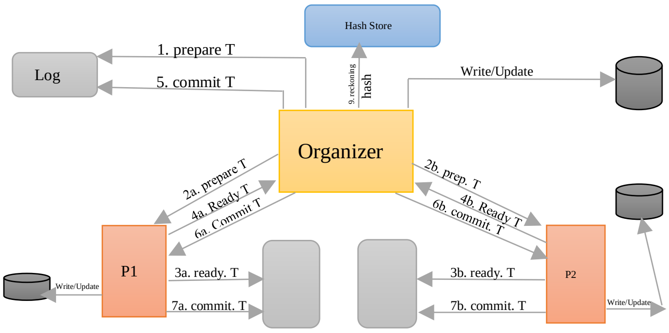

Fig. 4. Modified 2PC Protocol

L. *Testing the prevailing system for Data Change*

If the assaulter modifies the information with its hash collision [8] then the hash operation of the changed information can calculate to an equivalent worth because of the hash of the initial data. Since the hash worth of each the legitimate, and therefore the malicious information is likely to be the same, the user won't be ready to distinguish between them, rendering the target of the Hash Store to fade.

Hash Collisions:
Hash Functions [7] ought to be collision resistant i.e. it ought to be computationally unfeasible to seek out 2 completely different messages that hash to an equivalent worth. If 2 or a lot of messages have an equivalent hash worth then such a scenario has been named a collision fourteen,15. Let H: M->R be a hash operation, wherever M is that the message area and R is that the ensuing hash worth area. Since M is quite R, in line with the pigeonhole principle, there'll be quite one message in M which can hash to an equivalent worth in R. These collisions are noted employing a parallel collision search technique16,17 as follows:

Let M: be the message that we want to seek out a hash collision, m'? M be the message the victim can volitionally sign and gm: R ->M be injective operate that maps a hash end in R to a perturbation of message m in M. The perturbation of the message is specified it does not modify the linguistics which means of message m. Since the metric weight unit is an injective operating for one r, there is just one m, m specified gm(r) = m.

Partition of the set R into same-sized subsets R1, and R2 i.e. |R1| = |R2| and outline an operation f: R -> R specified that

F (r) = H (gm of (r)　　if (r of R1)
H (gm' of (r))　　　　if(r of R2)

- Generate a trail of points xi = f(xi-1) for i = 1,2, … till a distinguished purpose x_d, with a simple testable property, is found
- Store the distinguished purpose found in enlisting and continue on the top of the method till the same distinguished point seems certainly twice within the list. Finding the same distinguished points in the list might imply that two trails intersected at a point i.e. $f(x_i) = f(x_j)$ such that $i \neq j$.
- Check if xi and x_j belong to completely different subsets R1 and R2. If they do, then we have got successfully a collision otherwise repeat the top steps until you discover a collision.

III. ENCRYPTION AND DECRYPTION PROCESS

Encryption [Fig. 6.] is the method of translating plain text information into one thing that seems to be random and meaningless ciphertext. The decipherment is a method of changing ciphertext reverse to plaintext. To write quite a tiny low quantity of information, radially symmetrical secret writing is employed. A radially symmetrical key's used throughout each the secret writing and decipherment processes. To decipher a selected piece of ciphertext, the key that was wont to write the info should be used.

The goal of each secret writing rule is to create it as tough as the potential to decipher the generated ciphertext while not victimizing the key. If a very smart secret writing rule is employed, there's no technique considerably higher than methodically making an attempt at each potential key. For such a rule, the longer the key, the harder it is to decipher a bit of ciphertext while not possessing the key. It's tough to work out the standard of a secret writing rule. Algorithms that look promising typically end up being terribly simple to interrupt, given the right attack. Once choosing a secret writing rule, it's an honest plan to settle on one that has been in use for many years and has with success resisted all attacks.

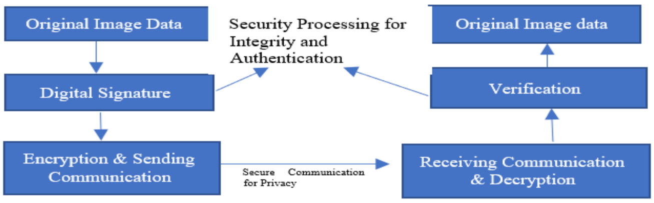
Original Image Data
Digital Signature
Encryption & Sending Communication
Security Processing for Integrity and Authentication
Secure Communication for Privacy
Original Image data
Verification
Receiving Communication & Decryption

Fig. 5. Encryption and Decryption process

IV. TESTING RESULTS

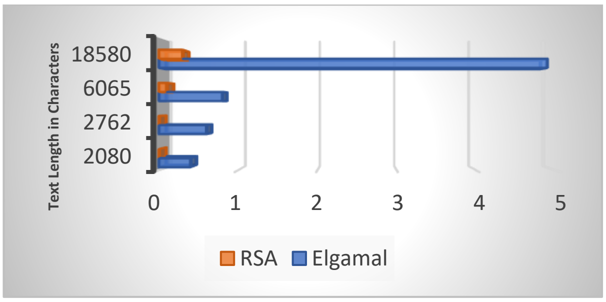

Fig. 6. Execution Times for Encryption and Singing

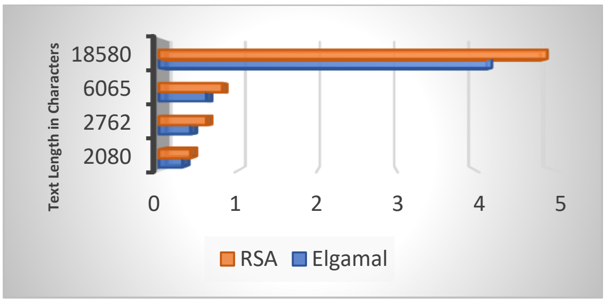

Fig. 7. Execution Times for Decryption

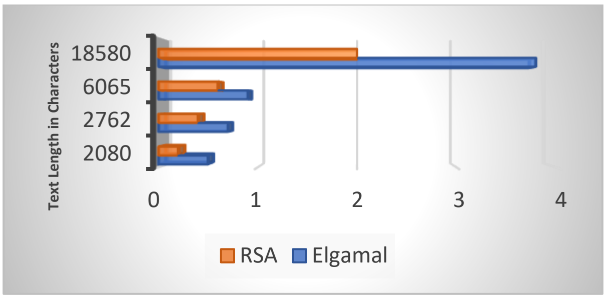

Fig. 8. Execution Times for Signature verification

Table I. Execution Times for Execution Times for Encryption and Singing

Text length in characters	*Elgamal*	*RSA*
18580	29083.63 MS (28s)	3818.85 MS(3s)
6065	8380.38 MS (8s)	224.86 MS(0s)
2762	3731.93 MS (3s)	67.60 MS (0s)
2080	2892.60 MS (2s)	49.91 MS (0s)

Table II. Execution Times for Decryption

Text length in characters	*Elgamal*	*RSA*
18580	111.945 MS (0s)	162.42 MS(0s)
6065	23.88 MS (0s)	40.57 MS(0s)
2762	12.11 MS (0s)	19.253 MS (0s)
2080	8.60 MS (0s)	14.342 MS (0s)

Table III. Execution Times for Signature verification

Text length in characters	*Elgamal*	*RSA*
18580	3803.319 MS (3s)	2013.042 MS(2s)
6065	705.58 MS (0s)	133.162 MS(0s)
2762	302.890 MS (0s)	39.577 MS (0s)
2080	224.3202 MS (0s)	27.050 MS (0s)

Table IV. Nodes for cluster scheduled

To run Map/Reduce Tasks	Private Key	Public key
Node 31	4.5E+029	65537
Node 8	6.1E+017	17
Node 23	271041713	65537
Node 32	5.70E+017	17
Node 17	2.5E+593	65537
Node 26	1.54E+020	65537

Table V. Generation of public and private keys for varied key sizes

	Private key size in bits …………………………			
Public key based on bit size	8	16	32	64
17	53	31729	670549409	5.27e+018
257	93	26393	70765473	6.42e+018
65537	113	7973	1148897573	1.05e+018

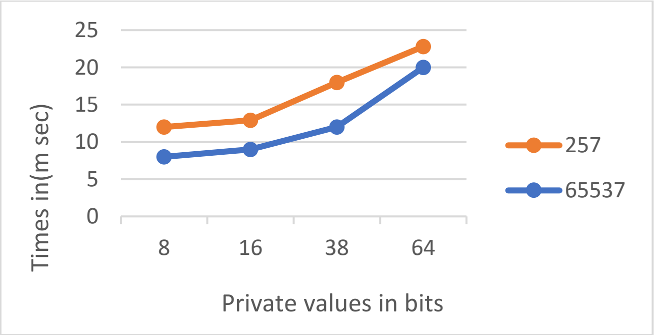

Fig. 9. Show the time taken for key generation using asymmetric RSA using different public keys

V. CONCLUSION

The security [12] of a Hash operation [7] depends on its collision resistant property that is set by the dimensions of its variety. Typically, this size is of the order of 2128,2256, etc. This immense set area makes the plain methodology, of choosing distinct inputs xi for i = one,2,3 ... and checking for a collision among the f (xi) values, terribly time intense. Thus, there's a requirement to marshal massive processing power from distributed processors and run the collision search task in parallel so as to seek out collisions in possible time. We have a tendency to aim to implement the testing rules mentioned on top of victimization of the Apache Hadoop's Map-Reduce framework. MapReduce's multiprocessing capability caters to the necessity of the parallel system needed by the testing rule. It mechanically partitions the info, schedules the execution of the rule across completely different machines and handles communication between machines [10].

VI. FUTURE WORK

Implement an algorithm based on this above experiment that works best with accurate time, and actual accuracy with fewer errors.

VII. REFERENCES

[1] http://www.cs.uvm.edu/~icdm/10Problems/index.shtml

[2] Ragib Hasan, Suvda Myagmar, Adam J. Lee, William Yurcik. Toward a Threat Model for Storage Systems. National Center for Supercomputing Applications (NCSA) The University of Illinois at Urbana-Champaign (UIUC).

[3] T. H. Ngac, I Echizen, K. Komes and H. Yoshiura, "New approach to quantification of privacy on Social Network sites," in IEEE, 2010, pp 556-564.

[4] Kevin Hamlen, Murat Kantarcioglu, Latifur Khan and Bhavani Thuraisingham. Security Issues for Cloud Computing. Technical Report UTDCS-02-10. February 2010.

[5] Theodoros K. Dikaliotis, Alexandros G. Dimakis, Tracey Ho. Security in Distributed Storage Systems by Communicating a Logarithmic Number of Bits. 4 May 2010.

[6] Anthony W. Scabby, Anil L. Pereria. Using Hashing to Maintain Data Integrity in Cloud Computing Systems. Department of Accounting, Computer Science and Entrepreneurship Southwestern Oklahoma State University Weatherford, OK73096.

[7] Selman Haxhijaha 1, Gazmend Bajrami 1, Fisnik Prekazi 1. Data Integrity Check using Hash Functions in Cloud environment.

[8] Jean –Jacques Quisquater and Jeab-Paul Delescalille. How easy is a collision search? New results and applications to DES. Springer - Verlag. 1998.

[9] Kumar et al Managing Cyber threats: Issues, Approaches and Challenges Springer Publishers, 2005.

[10] Ahmad Baraani-Dastjerdi; Josef Pieprzyk; Baraani-Dastjerdi Josef Pieprzyk; Reihaned Safavi-Naini, Security in Databases: A Survey Study, 1996.

[11] Mr. Saurabh Kulkarni, Dr. Siddhaling Urolagin, Review of Attacks on Databases and Database Security Techniques, Facility International Journal of Engineering Technology and Database Security Techniques Research, Volume 2, Issue 11, November-2012.

[12] Thuraisingham, B. Big data security, and privacy. In Proceedings of the 5th ACM Conference on Data and Application Security and Privacy, San Antonio, TX, USA, 2–4 March 2015; pp. 279–280.