IJCSE International Journal of Computer Sciences and Engineering Open Access
Research Paper Vol.-6, Issue-8, Aug 2018 E-ISSN: 2347-2693

Encrypted Query Data Processing in Internet Of Things (IoTs) : CryptDB and Trusted DB

G. Ambika[1*], P. Srivaramangai[2]

[1,2]Department of Computer Science, Marudupandiyar College (Affiliated to Bharathidasan University), Thanjavur - 613 403,Tamilnadu,India.

[*]*Corresponding Author: ambikag1212@gmail.com, Tel.: +91-12345-54321*

Available online at: www.ijcseonline.org

Accepted: 16/Aug/2018, Published: 31/Aug/2018

Abstract - Internet of Things (IoT) consists of two merging technologies IoT and Cloud Computing is known as IoT. In the cloud, the large amount of data might be gathered by various IoT applications. Privacy and Security concerns on behalf of IoT are proven targets of great significance. Encrypted Query Processing is securely to protect data confidentiality as preserving confidentiality and performing a standard set of SQL queries in an accurate manner for IoT. In this paper, compare the CryptDB and TrustedDB encrypted query processing systems with the purpose of IoT data stores securely in the Cloud database. The performance of an encrypted databases CryptDB and TrustedDB are compared by using SQL queries from TPC-H benchmark. As a result of that TrustedDB is performed as more efficient and scalable to large datasets.

Keywords- Internet of Things, TrustedDB, CryptDB, Encrypted Query Processing, Homomorphic Encryption

I. INTRODUCTION

In recent days, IoT develops into more relevant to the practical world due to the development of data analytics, embedded and ubiquitous communication, mobile devices and cloud computing. In IoT concept, it has invented in the year of 1999 by a member of the Radio Frequency Identification (RFID) under the development community. IoT is an internet based concept that consists of three significant things [1]: they are (1) People to people, (2) People to machine /things, (3) Things /machine to things /machine, Interacting through internet. With the reference of IoT communication becomes extended via Internet to all the things that surround us. IoT is much more efficient compare than 2G/3G/4G,GSM,GPRS,RFID, microprocessor, microcontroller, WI-FI, , wireless sensor networks, machine to machine communication, GPS, etc. IoT abbreviates as Internet of things and it is also a network of physical objects. The internet is not only considered as a network of computers, but it may also developed into a network of all type of devices and sizes , medical instruments, cameras, smart phones, peoples, constructions, vehicles, toys, home appliances and industrial systems, all connected and communicated and sharing information that depends upon on stipulated protocols so as to achieve tracing, safe and control, smart reorganizations and positioning, even through process and administration control, personal real time online monitoring and online up-gradation[1,2]. The major goal of IoT is enhanced to allow things to be communicated and connected by anywhere, anyone and anytime preferably utilizing any network/path and any service.

Currently, IoT security is the most important concern to be addressed [3]. When security measures are not properly provided for doing transmissions and data operations, after that the data performs at high risk, lots of organizations are holding return back from adopting the technology because of security concerns and issues. To ensure security, data owners should prevent from unauthorized access even as the data is to be processed or stored in an effective manner. Storing data on an un-trusted database needs protection for security measures against curious personnel working for the outside attackers or service provider utilizing software vulnerabilities at the database server. Additionally, data owners must authority of control over data access for their own personnel. The data is encrypted before storing in an effective way to guarantee confidentiality. An un-trusted database performs with encrypted query processing [5, 8, 1, 2, 6] where queries are executed on encrypted data. In cryptographic techniques, it allows computations that carry out on encrypted data using several encrypted query processing approaches are introduced [11, 17]. For the meantime, the capability to run on encrypted data in the cloud requires to be assured. Different encryption query processing approaches are promoted and the capabilities of executing SQL queries over encrypted data contain hardware and software systems. Existing practical systems are proposed MONOMI, SDB, CryptDB, SEEED, MuteDB, Arx, ENKI and TrustedDB etc, which integrates efficient

encrypted query processing into the database system. Software based encrypted query processing system are performed by using SDB, CryptDB and MONOMI. Therefore, a hardware approach to provide data security has taken by TrustedDB [7] and Cipherbase [9]. An outsourced database prototype is called TrustedDB and it enables clients to execute SQL queries with privacy under regulatory compliance without having to trust the service provider. In TrustedDB accomplishes by leveraging server-hosted tamper-proof trusted hardware in critical query processing stages.

II.COMPARATIVE ANALYSIS

A. CryptDB

In CryptDB approach is to execute queries over encrypted data using a collection of efficient SQL-aware encryption schemes. Therefore, curious database administrators and hackers never determine to access the decrypted data. The major goal of CryptDB exploits with this three concepts to accomplish. Initially , it uses well defined SQL queries that rely on equality checks, aggregates, and joins which in turn can be used to adapt encryption schemes to work on encrypted data. Secondly, it utilizes onions of encryption to store multiple cipher texts contained by each other, each layer of encryption utilizes an encryption scheme designed particularly to contain a different kind of query. Finally, it chains encryption keys to user passwords, with the intention that even if the database is compromised, the attacker must know the user's password to gain access to their data [15]. At this point the original CryptDB schemes are performed.

- Random (RND): RND offers the maximum privacy, and two equal values will be planned to different ciphertext. It is not able to allow efficient computation to be implemented on the ciphertext.
- Homomorphic Encryption (HOM): It is an encryption scheme that enables the server to perform computations on encrypted data to be securely protected.
- Word search (SEARCH): It enables the server to sense or detect repeating words in a particular node and also the implementation of a cryptographic protocol for performing keyword searches lying on encrypted text.
- Deterministic (DET): DET implements to the equality checks to be performed and that only leaks the encrypted values equivalent to the same data value, and no more.
- Order-preserving Encryption (OPE): It allows performing comparisons and establishing order relations between data values and it depends upon their encrypted versions.

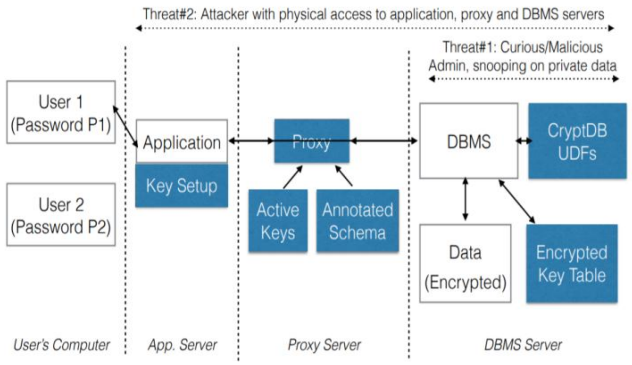


Figure 1: CryptDB architecture

The architecture of CryptDB in encrypted query processing system illustrates in Figure 1. It receives queries from the application server, protectively secures them and transmits them into the database server. After that, it receives the encrypted data from the database, then decrypts it and transmits to application server to be transmitted to the requester.

In CryptDB review, it rewrites queries to perform on encrypted data by intercepting the entire SQL queries within a database proxy, which decrypts and encrypts the entire data from the proxy database. Moreover, it modifies a few query operators at the same time as protecting the semantics of the query. In DBMS server process, it cannot receive the plaintext from decryption keys, as a result, it cannot notices sensitive data, ensuring that a curious DBA cannot gain access to private information (threat 1). DBMS server negotiates (threat 2) to safeguard against application, proxy and that developers define their SQL schema to different principals, all those keys allows to decrypt different parts of the database. Figure 4 reprsents a small modification to their applications to afford encryption keys to the database proxy. The proxy decides which parts of the database must be encrypted under which key is used. As a result of CryptDB assurances the data confidentiality belonging to users that are not log in during a compromise is detected and fixed by the administrator (eg.,user 2 in Figure 1). Although CryptDB safeguards data confidentiality, it cannot ensure the integrity, completeness or freshnessof results return back to the application. An opponent that compromises the database proxy, application, or DBMS server, or a malicious DBA, can remove all or any of the data saved within the database. Likewise, attacks on user machines, like cross-site scripting, are outside of the scope of CryptDB.

B. TrustedDB (Hardware based design)

Hardware based design to perform a computational ability on encrypted data, the keys are required to be present at the server to decrypt the data, compute, and after that encrypt over again. This model is the vulnerability of compromising cryptographic keys that considers the negative aspect. As a

result, many approaches and techniques have been discussed to overcome such vulnerabilities.

From these approaches exploit secure protection, tamper-proof hardware components attached to the server to perform computation over encrypted data and to store cryptographic keys [16]. In this manner avoiding the need to utilize expensive cryptographic operations, tamper resistant designs offer a secure execution environment for applications. On the other hand, they are considerably constrained for both memory capacity and computational ability that performs to implement fully featured database solutions using secure coprocessors (SCPUs) extremely challenging.

By exploiting general unsecured server resources perform to the maximum extent possible with the support of TrustedDB that conquers these limitations. For instance, TrustedDB allows the SCPU to perform transparently access the external storage at the same time as protecting data confidentiality by means of on-the-fly encryption. Likewise, to recognize the sensitive operations are pre-processed using client queries to run within the SCPU. In case of that non-sensitive operations are off-loaded to run in the un-secured host server that can be reduced the cost of transactions and also greatly improves performance of these operations in an effective manner.

TrustedDB makes use of secure, tamper resistant hardware like the IBM 4764/5 [17, 18]. The service provider's side to effectuate a complete SQL database processing engine is deployed by the cryptographic coprocessors. The TrustedDB design offers to perform strong data confidentiality assurances. Trusted hardware devices are largely utilized for protecting and secure coprocessors within Automated Teller Machines (ATMs), security protection and secure authentication purpose, for example, smart cards.

The most important proposals of processing queries reserved inside tamper-proof wrapped with trusted hardware, such as secure coprocessor or a Field Programmable Gate Array (FPGA)-based secure programmable hardware [19]. At particular stage, CSP's components are physically hosted in an effective manner. A limited set of queries over cipher-texts are allowed and accessed using the encryption keys. TrustedDB is an SQL database processing engine that makes use of IBM 4764/5 cryptographic coprocessors [21] to run custom queries securely process [20]. In TrustedDB, the cryptographic constructs are depending upon trapdoor function and at present viable trapdoors based on modular exponentiation in large fields and viable homomorphisms involve a trapdoor function for computing the ciphertexts.

TrustedDB extends SQL syntax by using keywords to check whether one attribute is sensitive or not. Every part of decryption is performed using TrustedDB within the secure confinements of the secure coprocessors. TrustedDB is an approach with the intention of merges the encryption process and the secure servers, and then it runs a lightweight SQLite database on the SCP and a feature-rich MySQL database on the commodity server. The strong data confidentiality assurances are provided by the support of TrustedDB design. In TrustedDB, a set of core components comprises a query parser, a processing agent, and a request handler, a query dispatch module, a paging module, communication conduit, two database engines, and a cryptography library in Figure 2. Many cryptographic schemes presented by IBM 4764/5 [20] Coprocessors such as RSA, the Advanced Encryption Standard (AES), cryptographic hash functions, the Triple Data Encryption Standard (3DES) and pseudo-random number generation. But, cryptographic coprocessors are considerably constrained in both memory capacity and computation ability. Therefore, a trade-off must be considered between low-priced query processing on un-trusted main processors and expensive computation inside secure coprocessors (at the CSP's). A secure coprocessor or the user must be processed and decrypted the sensitive data. Non-sensitive data are only stored un-encrypted at the CSP's, if a query issue is raised, it is encrypted at the user's side, after that rewritten as a set of sub-queries and in the secure coprocessor database engine or execute at the CSP's, with similar to data sensitivity. As a final point result is assembled, encrypted by the secure coprocessor and transmit return back to the user.

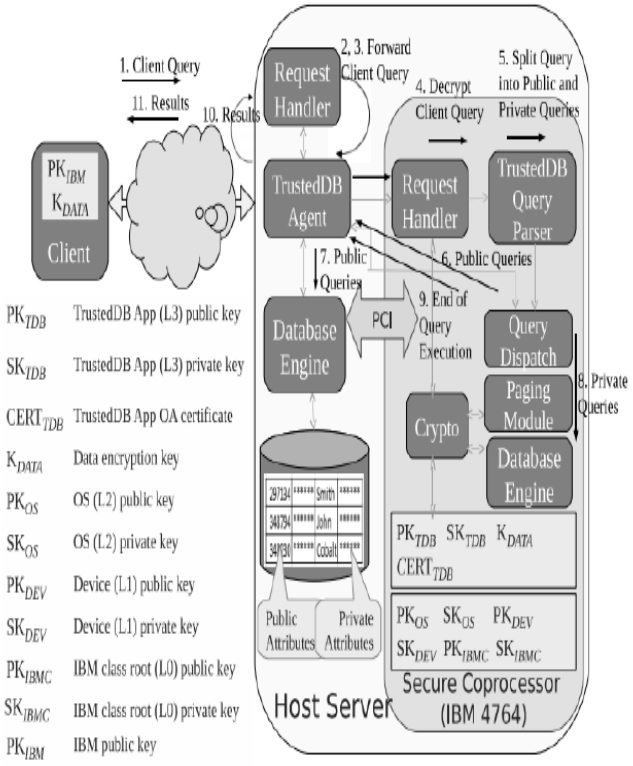


Figure 2: TrustedDB architecture [21]

III. IMPLEMENTATION PROCESS

A. Processing a Query in CryptDB

- User concerns a query that has intercepted by Database proxy and then re-writes the column and table name using a "Key".
- Database proxy checks whether when the DBMS server is to be specified keys to adjust onion layers, the proxy concerns update command instead of issuing keys to call appropriate UDFs.
- Database proxy forwards the DBMS server from the query which is implemented using standard SQL.
- DBMS server returns the database proxy form the query result that decrypts and returns plain text to the user.

B. Query Parsing and Execution in TrustedDB

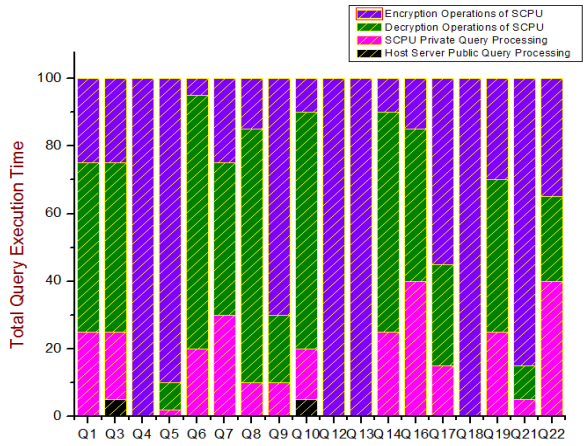
- A database schema defines a client side operation and partially populates it. Sensitive attributes are marked using the SENSITIVE keyword which the client layer transparently processes by encrypting the corresponding attributes, For example: CREATE TABLE student (ID integer primary key, Name char (72) SENSITIVE, Address char (120) SENSITIVE, Mobile integer (15));
- A query request is transmitted by a client to the host server through using a standard SQL interface. With the support of SCPU, in client site encryption process, the query request transparently encrypts using the public key. But, the host server does not able to decrypt the query. Therefore, the encrypted query forwards from the host server to the Request Handler within the SCPU.
- The Request Handler decrypts the request query; afterward it can be forwarded to the Query Parser. A set of plans are generated by the query parser and then each plan constructs as a result of rewriting the original client query into a set of sub-queries and their target data set classification, in the plan of each sub-query is recognized as either private or public.
- The Query Optimizer then estimates the execution costs of each of the plans and chooses the superior plan (one with least cost) for execution forwarding it to the dispatcher.
- The Query Dispatcher forwards the private queries to the SCPU database engine and the public queries to the host server at the same time as handling dependencies. The net result is that the maximum possible work is run on the host server's cheap cycles. As a final point, the query result is encrypted, digitally signed by the SCPU Query Dispatcher, and transmit to the client site.

IV. PERFORMANCE ANALYSIS

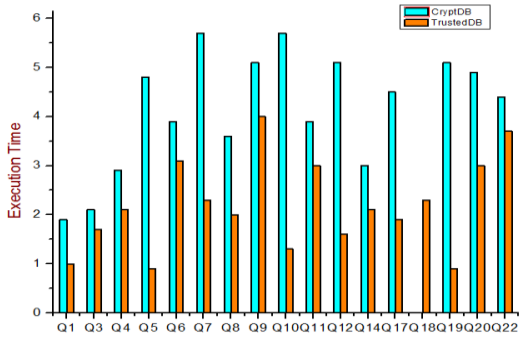
To estimate the performance of CryptDB, a machine with eight 3.4 GHz AMD Opteron 8431 6-core processors and 64 GB of RAM capacity to run the CryptDB proxy and the clients are utilized, a machine by means of two 2.4 GHz Intel Xeon E5620 4-core processors and 4 GB of RAM to run the MySQL 14.12 server and the clients are utilized. Designed for TrustedDB the SCPU of option is the IBM 4764-001 PCI-X with the 3.30.05 release toolkit featuring 32 MB of RAM and a PowerPC 405GPr at 233 MHz. The SCPU sits on the PCI-X bus of an Intel Xeon 3.4 GHz, 4 GB RAM Linux box (kernel 2.6.18). The server DBMS is to be a standard MySQL 14.12 Distribution 5.0.45 engine. The SCPU DBMS is a heavily changed SQLite custom port to the PowerPC. The TrustedDB stack includes Query Parser, Communication Conduit, Paging Module, TrustedDB Agent Query Dispatcher, Crypto Engine, etc. is written in C. TPC-H Query Load, to evaluate the runtime of generalized queries from the TPC-H set [22] of varying degrees of privacy. The TPC-H scale factor is 10 i.e, the database size is 1GB. TPC-H schema is enhanced with SENSITIVE attributes.

The breakdown of execution times of the private and public sub-queries depicts in Figure 3(a). The Private queries include the execution times needed for encryption and decryption operations inside the SCPU. The public queries execute on the host server and also include the processing times to interface the TrustedDB stack with the server database engine and output the final results. In figure 3(b) shown as the execution time of all queries in TPC-H supported by both CryptDB and TrustedDB. Note that Q13 and Q16 are neither supported by TrustedDB or CryptDB underlying PPKC scheme is not able to support for matching keyword involving two or more patterns. CryptDB does not support views and for this reason cannot run in Q15. Moreover, Q21 owing to correlating subqueries times out with both CryptDB and TrustedDB. For the remaining queries, the results that illustrate in comparison to run an unencrypted trace of TPC-H workload on Postgres, there is an overall enhancement within execution time by: 2.34x for performing CryptDB, 1.73x for performing TrustedDB.

The execution time of Q10 with differing scale factors that depicts in figure 4. TrustedDB as exposed in Figure 4, the execution time linearly enhances with increasing table size. But, for CryptDB, the query execution time enhances exponentially with increasing table size. The reason being that calling Manually implemented CryptDB for equality comparisons is much slower than the internal equality mechanisms for which a database is optimized and TrustedDB utilized.



(a) Query time Profiles



(b) Execution Time

Figure 3: Query time profiles, Execution time of TPC-H workload running CryptDB and TrustedDB.

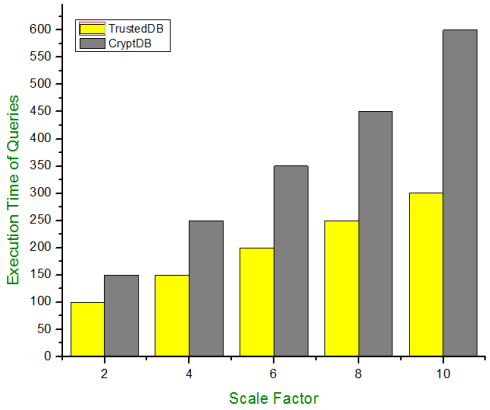


Figure 4: Execution time of Q10 with differing scale factors

In figure 5 demonstrates the time consumption to execute, rewrite, and post process a mix of 22 choose queries perceive in the TPC-H benchmark and compares them to their execution time within a single user mode. In support of the single user mode executes the encrypted queries without access the policy but, the multi user mode utilize the Linux Security Modules (LSM) access policy: they are two users and three user groups. LSM is considered into an internal SAP solution for supporting facility management to plan resources. All users get access to the private data and each user provide access to shared data, so that a multi user mode query performs to execute, rewrite, and post-process. To evaluate the time and compare their total execution time of the single user mode and the results of the 22 TPC-H queries clearly represents in figure 5. An average overhead of 36.98% is in the multi user mode to compare the single user mode (median overhead off 33.797%). It is considered as an absolute performance penalty of 0.6181 ms on average. Figure 5 demonstrates that Query 20 (including range condition) and Query 18 (including sum operator) both the performance consumes a significant larger amount of execution time to compare the entire other binary and unary relational operations in the single user mode and also in the multi user mode.

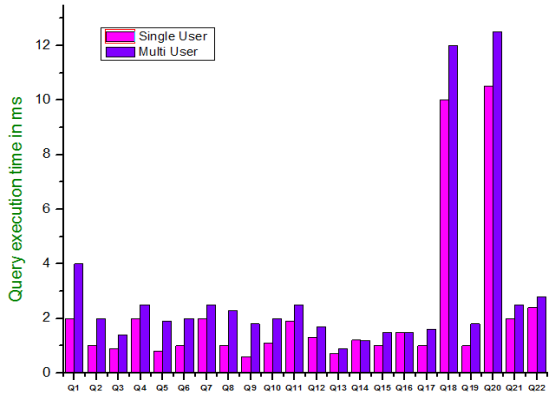


Figure 5: TPC-H: Query Execution Time for and Multi User Mode

Space and CPU overheads

TrustedDB is imposed to evaluate the CPU overhead, it consider in two scenarios. First scenario, the user performs to run the query request on a local database (by using plaintext Postgres, given that the machine is trusted). Second scenario, the user performs to run the query request on a remote trusted machine by using TrustedDB, and utilizes the local machine to run the TrustedDB client library. The ratio of the CPU execution time by the client library in scenario two and it has divided by the total execution time by the plaintext database in scenario one as shown in figure 6. Ideally, that ratio is always less than one, so that it always indicates less CPU time to outsource query processing with the support of TrustedDB. For most queries, this is true, except for queries 9, 10, 11, and 18. In those queries, the TrustedDB client library takes a lot of time for decryption process. We believe this is acceptable because it is easy to parallelize decryption across many cores on the client (whereas it is hard to parallelize Postgres), and because of TrustedDB does not need the client machines to store any data.

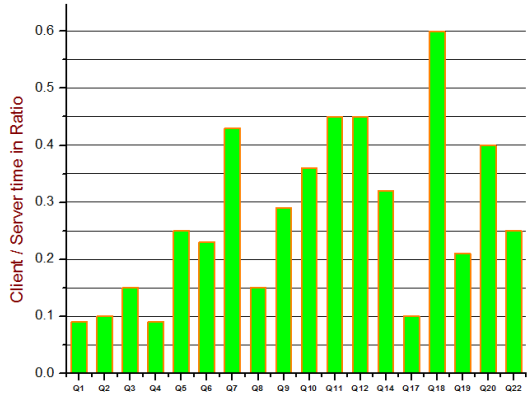


Figure 6: Ratio of the client CPU time required to execute a query using TrustedDB compared to the client time that would be required to execute the query using a local Postgres server.

TPC-H queries compare the performance under a large space budget (S = 2) with their performance under a smaller space budget (S = 1.4) by using two approaches. The first approach is CryptDB designer, and the second approach is a TrustedDB designer, finally that removes the largest column awaiting the space budget is satisfied. As a result of the four queries are affected by the space budget change as shown in Figure 7. Both TrustedDB designer and the CryptDB algorithm make a decision to drop the largest homomorphic column from the queries that significantly reduces the performance of query 1. The TrustedDB decides to drop the Order-preserving symmetric encryption (OPE) of the queries from the TPC-H, which significantly slows down query 6 because it can no longer apply a fairly selective predicate involving query on the server. Collectively, this slows down queries 6, 14, and 18 by a much smaller amount. Instead of CryptDB designer decides to drop two more homomorphic columns from the queries, replacing one with a pre computed deterministic column and it can also remove the OPE of the queries from the TPC-H.

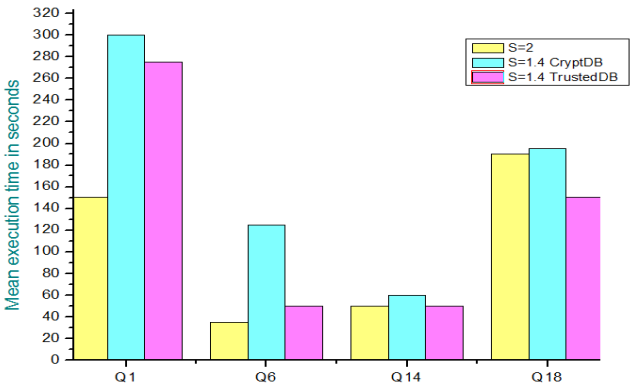


Figure 7: Execution times of queries that were affected by reducing the space budget from S = 2 to S = 1.4.

Throughput

The result on throughput performs while running TPC-H on Postgres, by the comparison to TrustedDB with CryptDB in figure 8. A logged-in user consists of multiple queries for each HTTP request so as to allow a user to create, read, update and/or delete a project(s), experiment(s) or job(s). A loss of throughput rate handle the results by performing 79% for cell level access control, 36% for row level and 64% for column level in both TrustedDB and CryptDB. The logged-in user is only able to access objects after if it is permitted. A reasonable overhead considers the increases in data privacy and confidentiality. The advanced granularity of row level over column level results in better performance because row level consumes less disk space and is able to take benefit of indexing to speed up table scans. Our scheme is most excellent performance for row level access control.

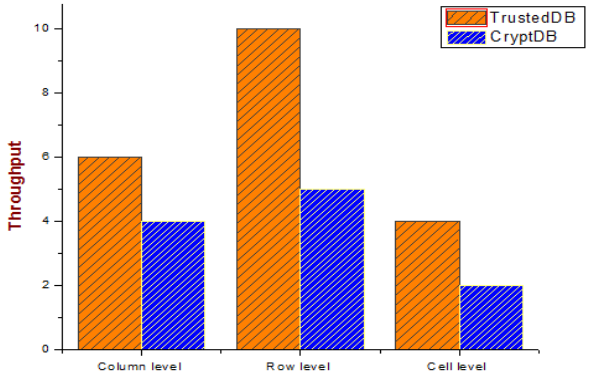


Figure 8: Throughput comparison at row level access control

V. CONCUSION

The sensitive data in the database requires to securely protected from different attacks or accessing from unauthorized hackers. TrustedDB ensures high data productivity in a secure manner with the support of SCPU hardware (Security coprocessor unit (SCPU) hardware) . Comparison between TrustedDB with CryptDB, as a result of TrustedDB is better than CryptDB depending on query execution time processing, throughput at row level and etc. As a final point, the results showed that TrustedDB be successfully in preserving the user experience at the same time as interacting with encrypted data. TrustedDB induces a moderate overhead for an improved level of security protection and data privacy. The hardware based TrustedDB implementations contain the cost overhead but better in performance and supported query types are not limited.

REFERENCES

[1]. Ken Eguro and Ramarathnam Venkatesan. FPGAs for trusted cloud computing. In 22nd International Conference on Field

Programmable Logic and Applications (FPL), Oslo, Norway, pages 6370, 2012.
[2]. Dr. Ovidiu Vermesan SINTEF, Norway, Dr. Peter FriessEU, Belgium, "Internet of Things: Converging Technologies for Smart Environments and Integrated Ecosystems", river publishers' series in communications, 2013.
[3]. Dr. Ovidiu Vermesan SINTEF, Norway, Dr. Peter FriessEU, Belgium, "Internet of Things–From Research and Innovation to Market Deployment", river publishers' series in communications, 2014.
[4]. https://www.ida.gov.sg/~/media/Files/Infocomm%20Landscape/Technology/TechnologyRoadmap/InternetOfThings.pdf
[5]. Popa, R. A. (2014). Building practical systems that compute on encrypted data. PhD thesis, Massachusetts Institute of Technology, Cambridge, MA, USA.
[6]. R. Agrawal, A. Evfimievski, and R. Srikant. Information sharing across private databases. In SIGMOD '03, 2003.
[7]. R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu. Order preserving encryption for numeric data. In SIGMOD '04, 2004.
[8]. M. R. Asghar, G. Russello, B. Crispo, and M. Ion. Supporting complex queries and access policies for multi-user encrypted databases. In Proceedings of the 2013 ACM Workshop on Cloud Computing Security Workshop, CCSW '13, pages 77–88, New York, NY, USA, 2013. ACM.
[9]. M. J. Atallah, K. B. Frikken, and M. Blanton. Dynamic and efficient key management for access hierarchies. In CCS '05, 2005.
[10]. A. Boldyreva, N. Chenette, Y. Lee, and A. O'Neill. Order-preserving symmetric encryption. IACR Cryptology ePrint Archive, 2012.
[11]. V. Ciriani, S. De Capitani Di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati. Keep a few: Outsourcing data while maintaining confidentiality. In Proceedings of the 14th European Conference on Research in Computer Security, ESORICS'09, pages 440–455, Berlin, Heidelberg, 2009. Springer-Verlag.
[12]. E. Damiani, S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati. Key management for multi-user encrypted databases. In StorageSS '05, 2005.
[13]. E. Damiani, S. D. C. Vimercati, S. Jajodia, S. Paraboschi, and P. Samarati. Balancing confidentiality and efficiency in untrusted relational dbmss. In CCS '03, 2003.
[14]. S. De Capitani di Vimercati, S. Foresti, S. Jajodia, S. Paraboschi, and P. Samarati. Over-enrcryption: Management of access control evolution on outsourced data. In VLDB, 2007.
[15]. H. Hacigümü¸s, B. Iyer, C. Li, and S. Mehrotra. Executing sql over encrypted data in the database-service-provider model. In SIGMOD '02, 2002.
[16]. R. A. Popa and N. Zeldovich. Multi-key searchable encryption. IACR Cryptology ePrint Archive, 2013:508, 2013.
[17]. R. A. Poppa, C. M. S. Redfield, N. Zeldovich, and H. Balakrishnan. Cryptdb: Proctecting confidentiality with encrypted query processing. In SOSP '11, 2011.
[18]. D. X. Song, D. Wagner, and A. Perrig. Practical techniques for searches on encrypted data. In SP '00, 2000.
[19]. Y. Yang, F. Bao, X. Ding, and R. H. Deng. Multiuser private queries over encrypted databases. Int. J. Appl. Cryptol., 1(4):309–319, Aug. 2009.
[20]. IBM 4764 PCI-X Cryptographic Coprocessor. Online at http://www-03.ibm.com/security/cryptocards/ pcixcc/overview.shtml, 2007.
[21]. IBM 4765 PCIe Cryptographic Coprocessor. Online at http://www-03.ibm.com/security/cryptocards/ pciecc/overview.shtml, 2010.
[22]. TPC-H Benchmark, http://www.tpc.org/tpch/, 2013.

Authors Profile

G.Ambika received her M.Phil Degree from Tamil University, Thanjavur in the year 2013. She has received her M.C.A Degree from Bharathidasan University, Trichy in the year 2010. She is pursuing her Ph.D (Full-Time) Degree at Marudupandiyar College of Arts & Science, Thanjavur, Tamilnadu, India. She has published 4 papers in International Journals. Her areas of Internet of Things, Cloud Computing and Mobile Computing..

Dr.P.Srivaramangai received her Ph.D Degree from Mother Teresa University, Kodaikanal in the year 2012. She received her M.Phil Degree from Manonmaniam University, Tirunelveli in the year 2003. She received his M.C.A Degree from Bharathidasan University, Trichy in the year 1996. She is working as Associate-Professor, PG and Research Department of Computer Science, Marudupandiyar College of Arts & Science, Thanjavur, Tamilnadu, India. She has above 30 years of experience in academic field. She published 25 papers in National & International journals so far. Her areas of interest include Computer Networks, Internet of Thing, Grid Computing, Cloud Computing and Mobile Computing.

