Computational Analysis of Encrypted Database to Provide Confidentiality

Swetha Tera, S. Ramachandram, and P. Shankar Murthy

***Abstract*—Cloud computing, is network based model used to illustrate a variety of computing concepts that involves a large number of computers connected through a synchronized communicational network such as the Internet. Cloud users demands security to their data which are stored in data repositories of cloud service provider. Thus the concept of Network Security can be applied over the cloud network, where several encryption algorithms are applied to provide integrity on the data. Such algorithms include Symmetric encryptions, Asymmetric encryptions, Hashing algorithms and Digital signatures. Even though these algorithms provide security, however these are not applied on query based data retrieval from databases where certain queries are used to invoke the data. Till to date many researchers proposed several models to provide security to the data in database and are not upto the mark. Since the operations are done in database which is located remotely, away from user, providing encryption on queries and data together will make an efficient approach. Such mechanisms like Homomorphic encryption, Order-preserving encryption are examined and a novel approach is defined to meet all security issues over a cloud termed as "CryptDB".**

***Index Terms*—Cloud computing, encryption mechanisms, homomorphic encryption, oder preserving encryption, CryptDB.**

I. Introduction

The rapid development in data processing and data storage technologies; the success of the Internet, computing resources have become cheaper, more powerful and more ubiquitously available than ever before. This technological trend has evolved in new network based computing called as Cloud computing [1]. Cloud computing, recently emerged as a new archetype for hosting and delivering wide range of services over the Internet.

There are several definitions which provide the perfect meaning to cloud computing. CISCO defined cloud computing as, IT resources and services that are abstracted from the underlying infrastructure and provided on-demand. According to NIST (National Institute of Standards & Technology) cloud computing is defined as, it is a model for enabling on-demand access to shared pool of configurable computing resources with minimal management effort. It is sold on demand, typically by the minute or the hour. Summarizing all the definitions it is defined as, Cloud computing is computational model, not a technology where wide range of services such as Iaas, Paas, Haas, Saas are provided through Virtualization concept. In this model of computing, all the network servers, networks, application interfaces (API) and other elements related to data centers are made available to cloud users via the Internet.

Till date there is no exact meaning of cloud computing, several researchers proposed several different definitions of cloud computing. Reuven Cohen defined cloud computing as an Internet Centric Software, which is the shift of the traditional single tenant approach to software development to scalable, multi network etc. Jeff Kaplan defined it as, cloud computing is a broad array of web-based services aimed at allowing cloud users to obtain a wide variety of functional capabilities on a 'pay-as-you-go' basis that previously required tremendous hardware and software investments and professional skills to acquire. Douglas Gourlay stated that Cloud computing [1] is a Virtual network, where network components are used virtually.

Cloud computing is often compared with other computing methods like Grid computing: it is a distributed computing prototype which coordinates on networked resources to achieve a common computational objective. Utility Computing: this computing represents the model of providing resources on-demand and charging end-users based on usage of service rather than a flat rate. Autonomic computing: this computing is developed by IBM, where a system acts as a sole responsible for internal and external processes without any interference of human simply it is a network with automated machines. Virtualization: is a technology that abstracts away the details of physical hardware and provides hardware and software resources virtually. Cloud computing is the combination of all the techniques which is available to the end user based on different payment schemes.

In a cloud computing environment, the conventional role of the service provider is divided into two categories: the infrastructure providers, who manage cloud platforms and lease resources such as CPU, storage system etc., (according to a usage-based pricing model), and service providers, who rent resources from one or many infrastructure providers to serve the cloud users.

The infrastructure providers distribute their infrastructure elements based on pricing system, mainly static price system: where the fixed price is maintained whatever may be the demand of it in the market, dynamic pricing system: it is a payment scheme where the service provider charges the end user based on what the infrastructure used at that particular moment.

Cloud computing provides wide variety of services such as IaaS, infrastructure as a service, where all the hardware elements are made virtual so that they are made available to the cloud users this concept is termed as virtualization. PaaS, platform as a service, this service provides a platform like user interfaces and programming interfaces which helps users

Manuscript received February 25, 2014; revised April 26, 2014.

Swetha Tera, S. Ramachandram and P. Shankar Murthy are with the Computer Science & Engineering, Osmania University, India (e-mail:{swethareddy.07, schandram, murthy619}@gmail.com).

to interact with the remote databases or servers through these interfaces and users can also develop their own applications and run them on their servers through these platforms. DaaS, data as a service, this type of service provides only data which are requested by the user and also stores the user data remotely and is made avail upon user request.

Among all these services DaaS is considered to be the most demanded service from cloud users as they exchange, store, modify their information through internet. Since this service is processed through internet there is chance of intrusion in the data and hack it. The very confidential data such as credit card numbers, online banking, passwords etc are readily available to the hacker if there is no security provided over the network [2]. In order to overcome such attacks certain security prototypes have been developed which resists such ad hoc attacks, Denial-of-Service attacks, jamming attacks and other fabrication attacks. Security algorithms are based on encryption standards, where the original text is encrypted to cipher text in the sender's end and is transmitted through any network, upon receiving cipher text by the receiver, decrypts it and reads the original message.

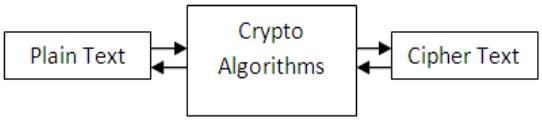

Fig. 1. Encryption and decryption scheme.

Such security algorithms [3], shown in Fig. 1, can be classified into symmetric algorithms and asymmetric algorithms. A single key is shared among sender and receiver to encrypt and decrypt the data known as common key/shared key/symmetric key. Algorithms like AES (Advanced Encryption Standard), Big-fish, Two-fish are based on single key encryptions [4], [5]. Other algorithms includes RSA, DSA are based on two key management schemes where the public key is used to encrypt the data and secret key/private key is used to decrypt the data, these algorithms are called as asymmetric algorithms since they use different keys for encryption and decryption.

All these security paradigms involve generation of keys to encrypt and decrypt the original message. Cipher text is sent through the network instead original text there is less chance of leaking the original message and other attacks are moderately prevented. Since this encryption revolves around keys for encryption and decryption, in order to decrypt the encrypted message by the receiver he must know the private/secret key which the sender has to send besides the cipher message. Simply one cannot send the key along with cipher text as attackers can easily decode it. Here, Key Exchange comes in play to exchange these secret keys in unreadable format which the attacker cannot understand (only the sender and receiver can understand).

Key exchange algorithms are only meant for exchanging the keys among sender and receiver. Such algorithms are Diffie-Hellman, German Army Enigma, and Key Wrap etc.

II. Literature Review

Now-a-days the data is stored in a database where the DBA controls the data and access controls of that data are under the DBA, who cannot be easily trustworthy.

Theft of sensitive private data from the database is a significant problem. Database management systems (DBMSs) [6] are an especially appealing target for attackers, because they often contain large amounts of confidential information. When individual users or enterprises store their sensitive data in a DBMS, they must trust that the server hardware and software are under safe state, that the data center itself is physically protected, and assume that the system and database administrators (DBAs) are trustworthy. Otherwise, an adversary who gains access to any of these avenues of attack can compromise the entire database.

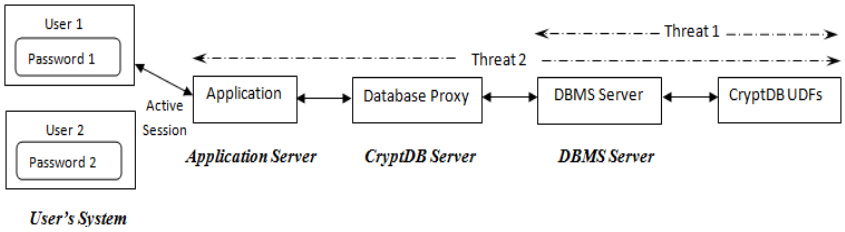

Fig. 2. CryptDB architecture.

For this reason we can make use of modern cryptosystem in database i.e., CryptDB. It works by intercepting all user issued SQL queries [6] in a cryptdb-database proxy, which rewrites queries to execute on encrypted data, as CryptDB assumes that all queries go through its proxy. The proxy encrypts and decrypts data and query, by generating parse tree preserving the semantics of the query. The DBMS server never receives keys to decrypt the cipher text to get plaintext, so there is no chance of accessing confidential data by DBMS server. This ensures that a curious DBA cannot gain access to private information (which is termed as Threat 1). The servers such as Application server, CryptDB-proxy and DBMS server are not guarded against hacking. They may get compromise (termed as Threat 2), if this happens the hacker gains control over all the servers and that data in database of the currently logged in user and still CryptDB assures that the data of the other users who are not logged in are under safe state.

Although CryptDB provides data confidentiality, it does not ensure the data integrity, completeness of results returned to the application (cryptdb user terminal). An antagonist that compromises the application, proxy, and DBMS server, or a malicious DBA, can modify or delete any or all of the data stored in the database.

CryptDB's architecture (as shown in Fig. 2) consisting of two parts: CryptDB-proxy server (CryptDB server) and DBMS server. CryptDB uses user-defined functions (UDFs) to perform cryptographic operations in the DBMS. Database proxy and cryptdb udf's are the components added by CryptDB and the application server is the default component

used by DBMS server. CryptDB addresses two kinds of threats, shown as dotted lines. In threat 1, a curious database administrator with complete access to the DBMS server snoops on private data, in this case CryptDB prevents the DBA from accessing secret information. In threat 2, an adversary gains complete control over both the software and hardware of the application, proxy and DBMS servers, in this case CryptDB ensures the adversary cannot obtain data belonging to users that are not logged in (e.g., user 2), but the data of the logged in user may get hacked.

III. Working with CryptDB

CryptDB is a library file which is linked dynamically while installing Mysql database, it adds new components besides mysql server such as parser, mysql proxy, key table, encrypted data etc.

CryptDB is a new encrypted system [7] that provides realistic and incontestable confidentiality in the face of these attacks for applications backed by SQL databases. It works by executing SQL queries over encrypted data using a collection of efficient SQL-aware encryption schemes. CryptDB chains encryption keys to the user passwords, so that the data item can be decrypted only by using the password of one of the users to access that data. As a result, a Data Base Administrator (DBA) never gets access to decrypted data, and even if all servers are compromised, an adversary cannot decrypt the data of any user who is not logged in [8]. Here, CryptDB address two threats, as specified above.

A. Downloading the CryptDB

CryptDB is developed in MIT University; it is stored in public repository called git-hub. This version of cryptdb is a freeware; any interested personnel can make use of it. The command to download this library is "*git clone -b public git://g.csail.mit.edu/cryptdb*"

B. Packages Needed to Support CryptDB

Linux packages like Bazaar, Bison, Gtk-doc, Autoconf, Automake, Libtool, Flex, Gcc, G++, MYSQL, MYSQL-Proxy.

C. Commands to Install and Run CryptDB

As cryptdb is supported only in Linux operating system, we may use Debian or Ubuntu flavors of Linux. Command to install Cryptdb is *"./scripts/install.rb file-path-root"*

Upon successful installation, use the following command to run cryptdb. *"obj/main/cdb_test ./shadow database-name"*

D. Onion Layers

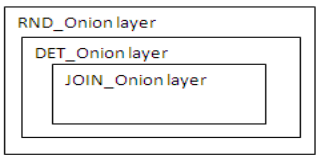

Fig. 3. Onion Layers for JOIN operation.

Cryptdb has a special function called Onion layer. It is defined as layer-by-layer encryption/decryption schemes which are called upon when a specific query is issued, such as JOIN operation calls ONION-JOIN onion layer, as shown in Fig. 3, each is identified by an *onion_id*. There are certain onion layers which are predefined and are specified as stored procedures and are called when appropriate query is triggered.

IV. Results and Discussions

CryptDB's design supports most relational queries, encryption schemes [9] and aggregates on regular data types, such as numbers and varchar types. Besides regular operations additional operations can be added to CryptDB by extending udf's, which is done by modifying existing onion layers of encryption, or by adding new onion layers for specific data types (e.g., spatial, aggregate functions and multi-dimensional range queries). There are certain SQL computations CryptDB cannot support on encrypted data. For example, it does not support both computation and comparison on the same column, such as *WHERE salary >age*3+20*. CryptDB can process a part of this query, but it would also require some help from the proxy. The general query processing done in cryptdb is shown below.

A. Processing a Query in CryptDB

1) User issues a query, which is intercepted by Database proxy and re-writes the table and column name using a 'Key'.
2) Proxy checks if the DBMS server is to be given keys to adjust onion layers, the proxy issues update command instead of issuing keys to call appropriate UDFs.
3) The Database proxy forwards the query to the DBMS server which executes using standard SQL.
4) DBMS server returns the query result to Database proxy which decrypts and returns plain text to the user.

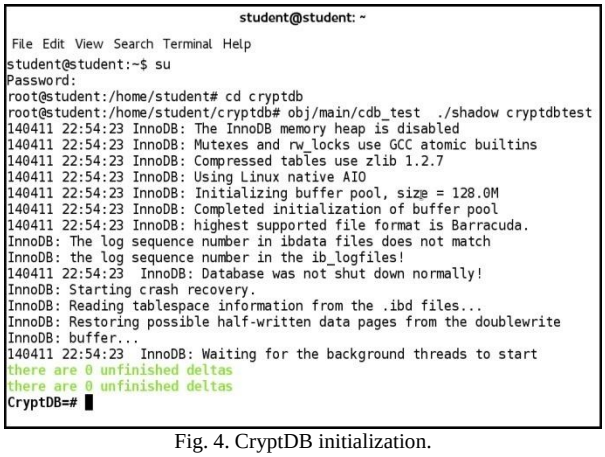

Fig. 4. CryptDB initialization.

In order to launch the cryptdb, the users need to log-in into the system as the root user by using his root password (i.e., by logging in as *sudo* user). The user needs to enter into the directory where the downloaded cryptdb is located, by *cd cryptdb* command. The above Fig. 4 shows the downloaded software is in student directory and the command "*obj/main/cdb_test*" initializes the cryptdb and the second half command "*./shadow cryptdbtest*" creates a new database in the MYSQL. Therefore the cryptdb is now ready to setup.

After successful installation of CryptDB we have created 'emp' table and inserted the columns 'eno' of number type and 'ename' of varchar type. When a query such as *SELECT*

* *FROM emp;* is issued, .It returns the encrypted query and equivalent result of that query, as shown in Fig. 5. Only CryptDB users can able to visualize the encrypted results and decrypted results. Whereas, the normal Database Administrator (DBA) can visualize only the encrypted data when he attempts to access the emp table from MYSQL, cryptdb prevents him accessing the decrypted results.

Fig. 5. Creating and selecting a 'emp' table from cryptdb.

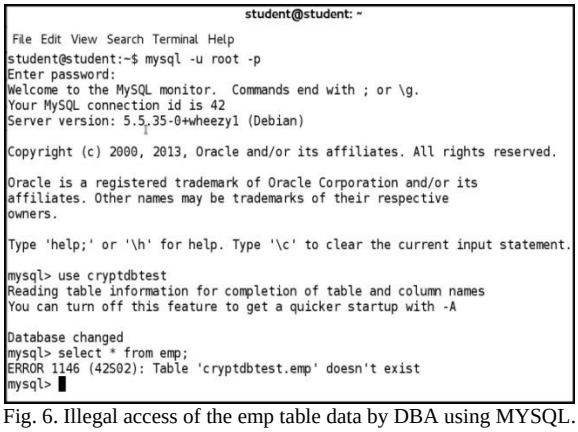

Fig. 6. Illegal access of the emp table data by DBA using MYSQL.

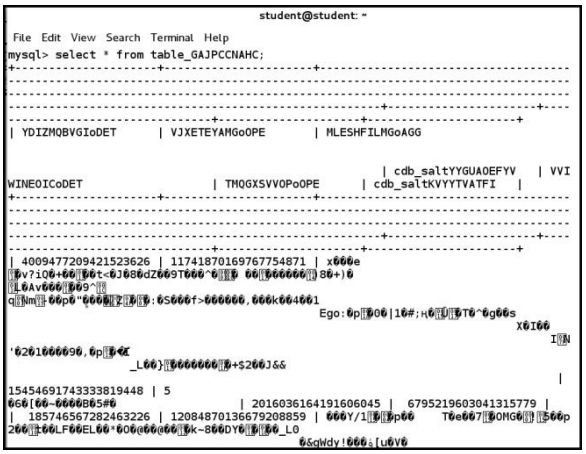

Fig. 7. Results displayed to the illegal user.

In order to guess the encrypted name of the 'emp' table by DBA, he can call any of the DET [10] encryption method, suppose let the 'emp' table is encrypted using AES encryption algorithm, one of the DET (deterministic) method.

*Select * from AES*(*emp*);

Which requires a symmetric key , which can generated through a Key Generator() [11], one cannot assure that the key generated in cryptdb is same as the key generated by the DBA manually, but there is a chance of getting same key, by which the DBA can know the encrypted name of 'emp' table [12]. Suppose the DBA or attacker successfully knows the encrypted name of the 'emp' table, and if he issues a query from MYSQL, he encounters only cipher text, as shown in Fig. 6, and plain text is hidden from him. Thus the private data is secured even though the DBA knows the encrypted names as shown in Fig. 7.

If the DBA or attacker is clever enough to crack the encrypted names of the table, column names and tries to modify/delete the data from MYSQL console by issuing certain commands like update, drop or delete, the cryptdb prevents such instances by throwing errors. Such situation is shown in Fig. 8 [13].

Fig. 8. Errors thrown by Cryptdb while illegal user modifies the data from MYSQL.

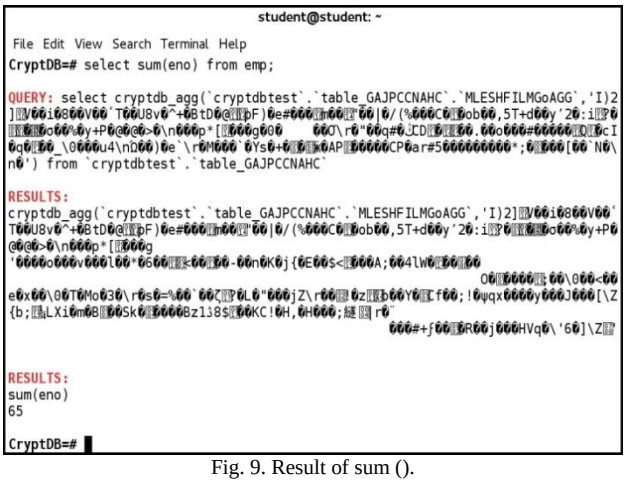

Fig. 9. Result of sum ().

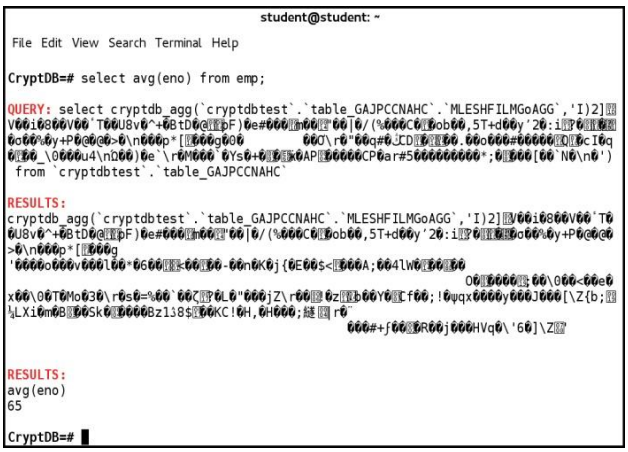

Fig. 10. Result of avg ().

The observations over aggregate functions such as sum()

and avg() when triggered in SQL commands in cryptdb resulted as in Fig. 8 and in Fig. 9 respectively. As the sum () should return the summation of the column 'eno' it returned value as 65, whereas the avg() function returned the same value of 65 instead of 21.6. By this analysis it is depicted that the avg() is not fully implemented.

As in the Fig. 4, the contents of the 'emp' table are 'eno' and 'ename' fields. The 'eno' field consists of employee number i.e., 35, 15, 15. The sum(eno) results in summation of these field values as 65. When avg(eno) is called it meant to return the avg of eno field values as shown in Fig. 10.

The User defined function for the avg() in cryptdb can be defined by using SQL statements within in C++ language[14], [15]. The blue print for avg() is shown below.

User defined Onion Layer for AVG()

AVG ()
{
Onion1: Decrypt using DET to get plain text.
Select "*SUM*(*column name*)" *from* "*table _name*";
Store the result in 'S' variable;
Select "*COUNT*(*column name*)" *from* "*table_name*";
Store the result in 'C' variable;
Return (*S/C*)*;*
Onion 1: Encrypt using DET to get cipher text.
}

The other observation observed is on the multiplication operator. Certain operators like *'*, /, %, ^, like'* are not implemented in cryptdb. The following figure shows the '*' operator bug.

V. Conclusions

Analysis done over cryptdb concludes that the cryptdb provides confidentiality to the user's private data through encryption schemes. It is based on relational databases and supports SQL queries, but not upto full extent. There are certain limitations which are observed in some of the queries where there it needs some improvements such as in avg () and moreover the onion layers need to be improved and can user defined onion layers can be implemented. Finally this paper concludes encryption in cloud tasks are very innovative, future extends some of the sql functions are not working in the CryptDB, it needs to improve CryptDB by working with all sql functions with results.

References

[1] R. Buyya, "Introduction to the IEEE transactions on cloud computing," *IEEE Transactions on Cloud Computing*, vol. 1, no. 1, January-June 2013.
[2] Privacy rights clearinghouse. Chronology of data breaches. [Online]. Available: http://www.privacyrights.org/data-breach
[3] P. Arora, R. C. Wadhawan, and Er. S. P. Ahuja, "Cloud computing security in infrastructure as a service," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 2, January 2012.
[4] O. Goldreich, *Foundations of Cryptography: Volume I Basic Tools*, Cambridge University Press, 2001.
[5] R. Agrawal, J. Kiernan, R. Srikant, and Y. Xu, "Order preserving encryption for numeric data," in *Proc. the 2004 ACM SIGMOD International Conference on Management of Data*, Paris, France, June 2004.
[6] G. Amanatidis, A. Boldyreva, and A. O'Neill, "Provably-secure schemes for basic query support in outsourced databases," in *Proc. the 21st Annual IFIP WG 11.3 Working Conference on Database and Applications Security*, RedondoBeach, CA, July 2007.
[7] A. Desai, "New paradigms for constructing symmetric encryption schemes secure against chosen-ciphertext attack," in *Proc. the 20th Annual International Conference on Advances in Cryptology*, August 2000, pp. 394-412.
[8] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Proc. Advances in Cryptology (CRYPTO)*, Santa Barbara, CA, August 2010.
[9] C. Gentry, "Fully homomorphic encryption using ideal lattices," in *Proc. the 41st Annual ACM Symposium on Theory of Computing*, Bethesda, MD, May-June 2009.
[10] M. Bellare, M. Fischlin, A. O'Neill, and T. Ristenpart, "Deterministic encryption: Definitional equivalences and constructions without random oracles," in *Proc. CRYPTO 2008*, 2008, pp. 360-378.
[11] J. Daemen and V. Rijmen, "Rijndael: The advanced encryption standard," *Dr. Dobb's Journal*, pp. 137-139, 2001.
[12] E. Damiani, S. D. C. di Vimercati, S. Jajodia, S. Paraboschi, and P. Samarati, "Balancing confidentiality and efficiency in untrusted relational DBMSs," in *Proc. the 10th ACM Conference on Computer and Communications Security*, Washington, DC, October 2003.
[13] C. Curino, E. P. C. Jones, R. A. Popa *et al.*, *Relational Cloud: A Database-as-a-Service for the Cloud*, 2011, pp. 235-240.
[14] R. A. Popa, C. M. S. Redfield, N. Zeldovich and H. Balakrishnan, "CryptDB: Protecting confidentiality with encrypted query processing," in *Proc. 5th Biennial Conference on Innovative Data Systems Research*, pp. 85-100.
[15] R. A. Popa, N. Zeldovich, and H. Balakrishnan, "CryptDB: A Practical Encrypted Relational DBMS," in *Proc. the Twenty-Third ACM Symposium on Operating Systems Principles*, 2011, pp. 1-13.

Swetha Tera received the bachelors degree in computer science and engineering from Jawaharlal Nehru Technological University, Hyderabad, India in 2008 and the M.Tech degree in computer science and engineering from Jawaharlal Nehru Technological University, Hyderabad, India in 2010. She is currently pursuing the Ph.D. degree in computer science and engineering at Osmania University, Hyderabad, India from 2011.

S. Ramachandram received the bachelers degree (BE) in electronics and communication engineering in 1983 from Osmania University, Hyderabad, India. He received the masters degree (ME) in computer science and engineering in 1985 from Osmania University and the Ph.D. degree in processing of read-only transactions in mobile broadcast environment in June 2005 from Osmania University, Hyderabad. He is working as a professor since June 2005 and the head of the Dept. of CSE from June 25, 2007. He worked as an associate professor since Sept. 1991 to June 2005; as an assistant professor from Oct. 1988 to Sept. 1991; as an adhoc lecturer from Feb. 1988 to Oct. 1988; as a senior faculty member from June 1987 to Jan. 1988 at Computer Literacy Foundation, Hyderabad; as a lecturer from Nov. 1986 to April 1987 at CBIT, Hyderabad; as an associate faculty member from Feb. 1985 to Oct .1986 at Indian Institute of Computer Technology, Cochin. He is the author of many books and publications.

P. Shankar Murthy received the bachelor degree in computer science and engineering from JNTU, Hyderabad, India in 2008 and the M.Tech degree in computer science and engineering from Jawaharlal Nehru Technological University, Hyderabad, India in 2010. He is currently working as an assistant software engineer in Cognizent.