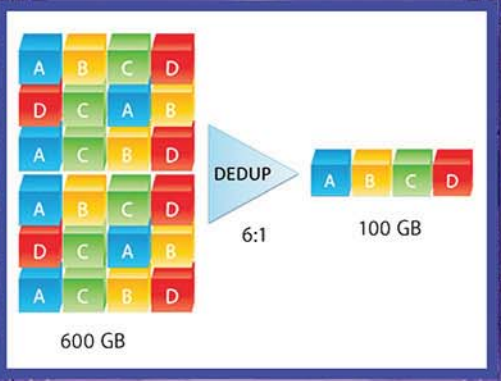
DATA DEDUPLICATION APPROACHES

CONCEPTS, STRATEGIES, AND CHALLENGES



Edited by
Tin Thein Thwel and G. R. Sinha

DATA DEDUPLICATION APPROACHES

CONCEPTS, STRATEGIES, AND CHALLENGES

Edited by Tin Thein Thwel and G. R. Sinha

In the age of data science, the rapidly increasing amount of data is a major concern in numerous applications of computing operations and data storage. Duplicated data or redundant data is a main challenge in the field of data science research. *Data Deduplication Approaches: Concepts, Strategies, and Challenges* shows readers the various methods that can be used to eliminate multiple copies of the same files as well as duplicated segments or chunks of data within the associated files. Due to ever-increasing data duplication, its deduplication has become an especially useful field of research for storage environments, in particular persistent data storage. *Data Deduplication Approaches* provides readers with an overview of the concepts and background of data deduplication approaches, then proceeds to demonstrate in technical detail the strategies and challenges of real-time implementations of handling big data, data science, data backup, and recovery. The book also includes future research directions, case studies, and real-world applications of data deduplication, focusing on reduced storage, backup, recovery, and reliability.

Key Features

- Includes data deduplication methods for a wide variety of applications
- Includes concepts and implementation strategies that will help the reader to use the suggested methods
- Provides a robust set of methods that will help readers to appropriately and judiciously use the suitable methods for their applications
- Focuses on reduced storage, backup, recovery, and reliability, which are the most important aspects of implementing data deduplication approaches
- Includes case studies

About the Editors

Tin Thein Thwel, PhD is a Professor at Myanmar Institute of Information Technology (MIIT), Mandalay, Myanmar. She received her PhD in Information Technology from the University of Computer Studies, Yangon (UCSY), Myanmar. She is a reviewer and technical committee member of the International Conference on Computer and Applications (ICCA) on data deduplication, cyber security, data mining, and information retrieval. She has 16 years of teaching experience at the university level and her research interests include data deduplication, cyber security, data mining and data science, information retrieval, and distributed computing.

G. R. Sinha, PhD is an Adjunct Professor at International Institute of Information Technology (IIIT), Bangalore, India, and presently deputed as Professor at Myanmar Institute of Information Technology (MIIT), Mandalay, Myanmar. He has published 259 research papers in various international and national journals and conferences. He has edited four books with Elsevier, Springer, and IOP; and currently editing seven more books with reputed publishers. He is a Visiting Professor (Honorary) of Sri Lanka Technological Campus Colombo. He is a Senate member of MIIT and also ACM Distinguished Speaker in the field of Digital Signal Processing. His research areas include cognitive science, brain computing, image processing, and data science applications.

ELSEVIER

ACADEMIC PRESS
An imprint of Elsevier
elsevier.com/books-and-journals

ISBN 978-0-12-823395-5

Contents

13

Performance modeling for secure migration processes of legacy systems to the cloud computing

Ankit Kumar[1], Pankaj Dadheech[1], Vijander Singh[2], Linesh Raja[3]

[1]DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, SWAMI KESHVANAND INSTITUTE OF TECHNOLOGY, MANAGEMENT AND GRAMOTHAN, RAMNAGARIA, JAGATPURA, JAIPUR, INDIA [2]DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING, MANIPAL UNIVERSITY JAIPUR, JAIPUR, INDIA [3]DEPARTMENT OF COMPUTER APPLICATIONS, MANIPAL UNIVERSITY JAIPUR, JAIPUR, INDIA

13.1 Data migration in cloud computing

The object and data can be moved from one cloud to another (Coelho, Paulo, Vilaça, Pereira, & Oliveira, 2017). However, this is a challenging process for the transfer of data and covers various key issues of protection such as data integrity, protection, portability, data privacy, data accuracy, etc. (Domaschka, Griesinger, Baur, & Rossini, 2015). To achieve an automatic data migration, a programmatic data migration approach is required to get rid of the tedious tasks of human operation. (Fig. 13−1).

13.1.1 Need for migrating data into the cloud

For integrating data under the various projects of an enterprise, data migration plays a crucial role in the field of cloud computing technology. Since business demands are proliferating, more and more applications are required to support these demands, and for this purpose, the cost involved for running and managing the databases for applications is increasing due to the increasing demands. Therefore in order to gain the benefits of cloud computing and to meet the growing demands by resolving the cost issues for integrating data for any organization, there is much need for bringing the concept of data migration into the cloud.

In the simplified form, the need for migrating data into the cloud arises if the merging of computer systems is performed or if an old computer system is to be replaced by a new

Data Deduplication Approaches. DOI: https://doi.org/10.1016/B978-0-12-823395-5.00003-3

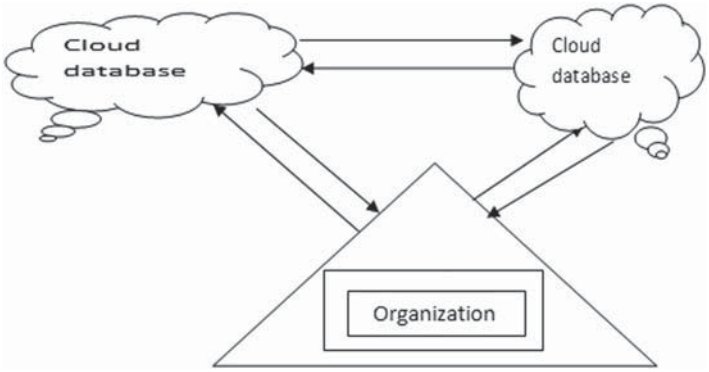


FIGURE 13–1 Data migration in cloud.

computer system or is upgraded to a new system. All the reasons supporting the need for data migration in the cloud are summarized below:

- If a company wants to transfer its data to another company, it is thought that better support can be obtained for their requirements by another cloud service provider (CSP).
- When the systems or accounts are to be merged by an organization after the acquisition.

13.1.2 Scenarios of data migration over cloud

The different scenarios of data migration over cloud are (Hong, Yao, Liu, & Qi, 2018):

1. On-premises to cloud
2. Cloud to on-premises
3. Intercloud migration
4. Data migration as a service (Fig. 13−2).

13.1.3 Migrating data from local data store to new cloud data store

Now in order to move data from a local cloud store to the new cloud data store, some simple steps are required to be followed. This can be explained with the help of a suitable diagram as follows: (Fig. 13−3)

The steps involved for data migration into new cloud data store are:

1. Initially, the scenario for cloud migration is identified.
2. In the second step, the focus is on the description of the cloud data hosting solution.
3. Selection of cloud data storage is made after describing the cloud data hosting solution.
4. Now, the data layer patterns are identified so as to solve the conflicts.
5. Before migration, an adaptation of the data access layer and upper application layers are created if needed.
6. At last, the migration of data into the selected cloud data store is performed.

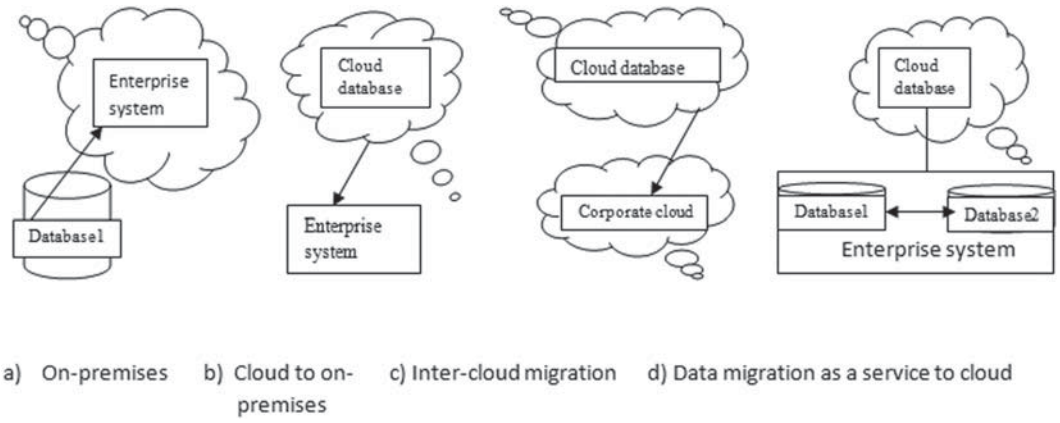


FIGURE 13–2 Scenarios of migration over cloud.

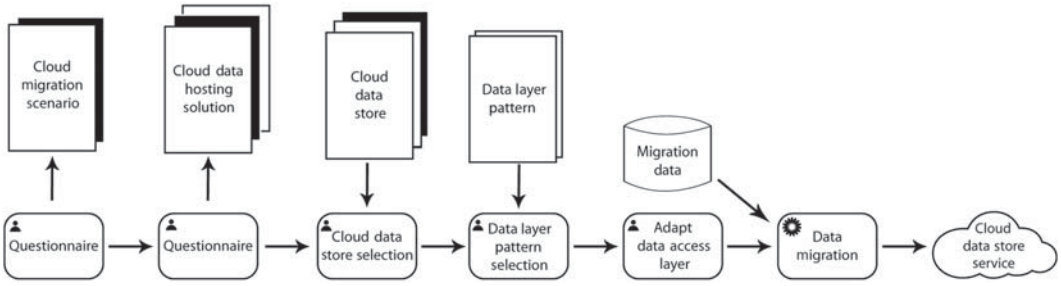


FIGURE 13–3 Data migration from local cloud store to new cloud store.

13.1.4 Complicated and error-prone migration to cloud

Many factors are responsible for making the task of migrating data to the cloud complicated and error-prone. These could also make this process quite expensive in nature. These reasons are:

- There are many configurations of the cloud which prove to be invalid in nature when they occur due to the changes in the cloud computing environment as they are dependent on the computing environment. In this regard, let us consider a suitable example of a database as a service migration to the cloud, in which the cloud gets the database server when it is migrated from a local data center. The change in the location of the database server from the local data center to the cloud by the migration process causes the change in its IP address. Thus a large number of operations for updating the configurations and IP address in all the components are performed during migration. But this could be problematic as the whole system will get out of operation if even a single update operation is skipped.

- Another reason that makes the migration process to cloud more complicated and error-prone is the presence of a large number of different components during the implementation of an enterprise system. For example, to balance the load into the system, it may require a large number of web and application servers. This may further lead to more complicated dependencies among the multiple servers. Then actions are taken to sort out these dependencies, which are a time-consuming task as compared to the migration process.
- The migration process is responsible for breaking the massive controlling settings such as the access control, that are hidden in nature, and this may further lead to the security threat in the system.
- Also, there are many careless errors made by human operators in the migration process, which cannot be easily identified.

13.1.5 Security challenges for data migration

While migrating databases into the cloud, a number of security threats arise, which require some strong actions to be taken into consideration so as to maintain the security of data during the migration process. Moreover, it is the responsibility of the organization itself to check and assure the security of the database rather than the service providers. Since the data that has to be migrated to the cloud may contain some secret information which the user does not want to share with anyone, it should be secured at its best by making proper measurements including data confidentiality, authenticity, and data integrity for securing the database.

- Data confidentiality
- Authenticity
- Data integrity

Also, the database security can be viewed and maintained in the cloud when the data is at rest, in motion, or in use.

- When data is in motion or being transmitted from and to the database, then some secure methods are carried out to protect the data so as to prevent it from being damaged.
- A method of encrypting the data for providing confidentiality in the cloud database is used.
- Properly monitored and controlled data access at the platform of the cloud database provider is required in order to achieve data integrity at a high level.
- Moreover, data in the cloud can now be secured using some standard communication protocols and procedures for security such as HTTPS, public key certifications, etc.

13.2 Literature review

Ibrahim, Aburukba, and El-Fakih (2018) proposed that cloud computing has become a future generation infrastructure for computing. Normally cloud computing is defined as a bunch of

computing resources accessed by the internet. Basically users store their data in the cloud with a firewall and other security to keep the data safe from a third party or intruders. In the cloud computing resources, there are many service providers who provide the services to the user, and it's their duty to keep the data safe for the user or protect the data from unwanted access, or the user can take the help from a third-party auditor to keep their data safe from the unwanted access and provide the security to their data in the cloud over the internet. This third-party application does not store their data from their end, and it is only a security provider between the service provider and the user. At the time of data migration by the hashing algorithm, this application provides data integrity verification. Also Secure Sockets Layer (SSL) protocol provides security for the data that is migrated in several ways, such as from any organization to one cloud to another cloud. It is challenging to transfer the data from one cloud to another cloud, with various security issues arising like data integrity, privacy, and data accuracy. In related work, a framework was proposed called provable data possession. This protocol is used to verify the data storage of the particular file. Third party authentication is used as a framework with SSL protocol. It is integrated with the random mask technology to achieve a privacy-preserving public auditing system.

Li, Zhai, Du, and Han (2018) proposed implementing the Rivest, Shamir, and Adleman (RSA) algorithm to obtain data security in cloud computing. Cloud computing is an internet-based technology, where a good amount of resources are shared as a service. It is a payment-based model where the user pays money for cloud services (Li et al., 2018). They mostly work on the security of users' data. Third party auditor and auditing mechanisms are proposed. In the proposed framework, the author implements the RSA algorithm to secure the user data in the cloud. Security of the user data is the issue or major concern. Secure cloud storage is a promising technology, but for the successful use of this technology, we have to provide assurance about the security gaps where the security is lacking. To provide assurance about security gaps, we have to make the strategy at the client-side of the cloud.

Pathan, Voudouris, and Stenstrom (2018) proposed a cloud computing application and a testing methodology. In this paper, the proposed cloud computing provides new security to the user. This new technology creates a platform for the user and activates the internal operation of the cloud to the user. To provide a better quality of service, it is representing security testing and assures better quality and accuracy for whatever user design. According to the authors the cloud computing technology works on the user demand service and provides access to shared resources and common infrastructures to the user. In cloud computing virtualization is a major term, and this virtual switching enables the user to spawn network topology as a virtual machine and connect them with a software. Because the performance of the data or software depends on the load of the server, it is challenging to test virtual circuit switching capability to notice high bandwidth requirements. The authors present different testing techniques like a stress test, load and performance, functional testing, compatibility test, latency test, SOASTA test, and other tests over the cloud. Testing of the cloud can be completed with many cloud services infrastructure and testing tools.

Whaiduzzaman, Naveed, and Gani (2018) proposed a study on testing as a service on the cloud. Testing of software is a process which is used for evaluating an attribute or capability

of the program and makes sure that it meets the requirement of the user. Nowadays, testing has become a significant activity in terms of security, performance, and usability. Cloud computing provides anything as a service, where a user can use the services of testing without maintenance and upgradation of the system, which is more cost-effective for the user. Testing as a service provides an on-demand service which can be static or dynamic. The main motive for this service is to reduce the IT budget of the businesses or organizations to focus their business by software testing to the third party authentication model. Cloud testing is important in the current scenario because not everyone can afford their own testing hardware and software, so they use the cloud testing techniques to reduce the cost to the company and for better performance with the minimum resources.

Zhang et al. (2016b) proposed testing techniques of software in cloud computing. Cloud computing is the next phase of internet development, which depends on resource sharing over the internet to get coherence. It is connected to new computing standards that rely on different fields, including software testing. There are multiple types of software testing techniques which are used for testing of software. They convert the methods of computing resources and also manage and deliver services with their solution. This creates new phases and requirements in the field of software testing. Software testing in the cloud minimizes the requirements of software and hardware resources and provides a scalable and efficient testing process. Software testing is used to confirm the completion of software functions according to the user's or client's requirements.

13.3 Proposed work

The object of this work is to build a new encryption algorithm which enhances the cloud data protection while migrating that would be better than using existing encryption algorithms such as PBE (prediction-based encryption) and IBE (identity based encryption).

There are various different types of encryption techniques of cryptography, as discussed in the chapter, which provide security to data during their transmission from one system to another over a network, but they possess some drawbacks. Therefore in order to improve the security for data migration over the internet (in a cloud computing environment), an enhanced encryption technique has been proposed in this chapter. One of the significant characteristics of our enhanced encryption method is that it makes use of both *the public key encryption* and *the private key encryption* in combination rather than using a single encryption technique (either public or private encryption), so as to find a better solution to the security challenges.

13.3.1 Enhanced encryption technique

The enhanced encryption technique used in our proposed work is basically a combination of the public-key (asymmetric key) encryption technique and private key (symmetric key) encryption technique. This encryption technique is said to be an enhanced encryption technique as it involves the enhancement of one encryption method by adding another encryption method so as to improve and thereby increase the security strength while migrating data in the cloud computing environment (Kaur & Aggarwal, 2018; Li, Yan, & Zhang, 2019).

A method of encryption which involves the merging of two or more encryption techniques such as a combination of asymmetric and symmetric encryption so as to take out the benefits from each of them is known as "*Hybrid Encryption*" or "*Enhanced Encryption*."

This kind of encryption provides a high level of security to the encryption system because of the presence of highly secured public and private keys (Lan, Fong, Song, Vasilakos, & Millham, 2017).

To carry out this type of encryption, initially the symmetric or private key encryption is performed with the help of some unique keys which are randomly generated in order to transfer the data. After that, the randomly generated key is encrypted using the asymmetric or public key by implementing the asymmetric encryption technique. Also, this public key of the recipient is then further used to decrypt the session key (random key), and at last, the data is decrypted by using the decrypt session key. In this way, by implementing an enhanced encryption technique in the data migration process, a high level of security can be easily achieved (Bermbach & Tai, 2014; Zhang, Chen, Luo, & Song, 2016a).

13.3.2 Concept of randomization in encryption

The concept of randomization used in our enhanced encryption technique generally defines a procedure in which initially a message or plain text P is encrypted into a number of ciphertexts such as C_1, C_2,. . ., Cn and then any one of the N ciphertexts is randomly selected. Second, the plain text is enciphered by returning all of the ciphertexts to the original plain text, since whoever decrypts the text does not know which one is selected. Since the message space will increase in size by adding a random ciphertext to it, the randomized encryption procedure will attain a high level of security in cryptographic systems and this system when used in cloud computing environments will provide a stronger and more secured data migration process.

Therefore by connecting a set of ciphertexts or codes to each plain text or encoding a plain text by randomly selecting any cipher text from a set of ciphertexts, the randomization in encryption enhances strong security to such codes or ciphertexts against the attack on the given plain text. Jefferson-Bazeries Wheel Cipher and its variations that were used at the time of both World Wars [Kru81] and [Kah67] by the United States is the best example of using the concept of randomization in encryption technique (Kuhlenkamp, Klems, & Röss, 2014).

The procedure of encryption using randomization can be defined by a relation "A" which is a subset of (M*K*C). Here, "M" refers to message space, "K" refers to keyspace, and "C" refers to ciphertext space. Now consider two cases as given below:

- *Case 1*: At most one message x belongs to M (message space) for each key k belongs to K (keyspace), and each ciphertext c belongs to C (ciphertext space) such that (x, k, c) belongs to "A."
- *Case 2*: In it, at least one ciphertext c belongs to C (ciphertext space), for each key k belongs to K (Keyspace), and each message x belongs to M (Message space) such that (x, k, c) belongs to "A."

Hence, the randomized encryption system can be defined as the quadruple (M, K, C, A).

From the above two cases of randomized encryption procedure, it has been concluded that the size of the ciphertext space "C" will be large compared to the size of the message space "M." This would further lead a transmitting channel to expand its bandwidth as the larger-sized ciphertext space requires more bits to be transmitted for its identification rather than identifying the comparatively smaller-sized message space. Since the bandwidth is increased during randomization encryption, this is known as *"Bandwidth Expansion."* This is the only disappointing factor that cannot be avoided while implementing the randomized encryption technique in the cloud environment (Grolinger, Higashino, Tiwari, & Capretz, 2013; Krintz, 2013), and it causes a significant cost in using such type of encryption. Hence, as a useful solution to this problem, a factor for expanded bandwidth has been defined. This factor is calculated as the ratio of a number of ciphertext bits transmitted to the corresponding number of message bits, as shown below:

$$\text{Bandwidth Expansion Factor} = \frac{\text{No. of transmitted cipher-text bits}}{\text{No. of corresponding message bits}}$$

The problem of expanded or increased bandwidth cannot be avoided or discarded but can be handled and controlled by the bandwidth expansion factor. Also, if this factor comes to be variable, then an average bandwidth expansion factor must be calculated instead of the bandwidth expansion factor. On the other hand, in a randomized encryption technique, the main focus is on generating random bits of data. Therefore some source is required for generating such a type of bits (Kumar & Sinha, 2019a; Yonghong, Na, & Guofeng, 2016). There are various ways that can be used to create such a source of random bits, and the examples include neon discharge tube, noisy diodes, radioactive decay. or some other natural source for providing a degree of randomization (Fig. 13−4).

The above diagram represents the transfer of data in a secure manner by the randomized encryption procedure over an insecure channel of communication, and the random bit

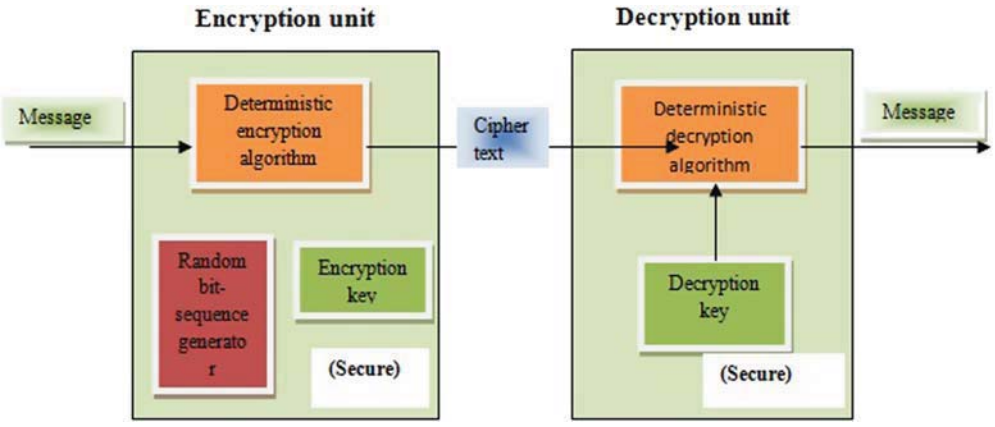


FIGURE 13–4 Diagram of randomized encryption procedure.

sequence generator is located inside the encryption unit in the secure area such that the enemy would be unaware about the output coming out of the random bit sequence generator (Kumar, Dadheech, Singh, Poonia, & Raja, 2019b; Kumar, Dadheech, Singh, Raja, & Poonia, 2019c).

The concept of randomization can be applied to both the block ciphers and the stream ciphers in different ways (Dadheech, Goyal, Srivastava, & Kumar, 2018). Now, let us have a look at some notations that would be used during the encryption procedure. Some notations are described in Table 13.1.

13.4 Proposed encryption approach

It has been already discussed that the asymmetric key encryption (or public key encryption) seems to be very slow and inefficient in performance as compared to the symmetric key encryption technique, although it is quite convenient to use it in exchanging the information securely as no secret sharing is required in an asymmetric key encryption technique. On the other hand, the symmetric key encryption method is quite efficient and faster to encrypt but it also lacks key distribution management which is a significant problem and also it is less secure than the asymmetric key encryption method as it involves the sharing of a private key between the sender and receiver (Kumar, Goyal, & Dadheech, 2018).

Therefore in order to achieve both the convenience and efficiency in encrypting the data or sensitive information in the cloud computing environment, an enhanced encryption technique is proposed which combines the convenient asymmetric key encryption method with

Table 13.1 Notations used in randomized encryption methods.

Symbol	Description
M	Message space
C	Ciphertext space
K	A set of keys
R	A set of bit sequences
M	A source of messages generated in M
R, R′	Source of random bits by which the sequence of bits are generated in R
k. K.′	Cryptographic keys in K
|M| and |R|	Length of messages and bit-sequences in M and R
E,E′	Deterministic encryption functions representing K*Domain(E) into Target(E) such that Domain(E) represents the message space, and Target(E) represents the ciphertext space.
D, D′	Decryption functions corresponding to E, E.′
P,P′	Pseudorandom-bit stream generators
$E_k(R)$	Encryption scheme in which the random bit-sequence are encrypted using encryption function E with key k.
||	Used for concatenating two or more variables, for example, A||B||C
⊕	Exclusive or (XOR)

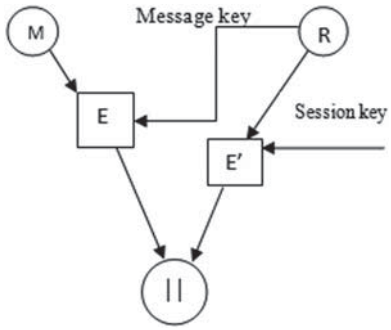


FIGURE 13–5 Enhanced encryption technique.

an efficient and fast symmetric -key encryption method to achieve strong security in the data migration process in the cloud (Kumar & Sinha, 2014, 2019b).

According to the concept of randomization within the block cipher, the given message is encrypted in the following manner: $\mathbf{E_R\ (M)\ ||\ E'R}$ (Fig. 13−5)

In this technique, the message M is encrypted by a random key (message key), and that random key is also encrypted (Kumar, Dadheech, Kumari, & Singh, 2019a) with a session key, and then both results are concatenated. Its major advantage is that it minimizes the malicious or brute force attack on its encryption function and also leads to a small bandwidth factor due to the longer length of the message than a random sequence. This can be further described in the form of an algorithm, as shown below.

13.4.1 Enhanced randomized encryption algorithm

13.4.1.1 For encrypting a message in the randomized hybrid system

1. Generate a random key by a randomized algorithm
2. Encrypt the data by using a random key
3. Encrypt the random key with a shared/session key
4. Forward both the data and key after the encryption process from step 2 and step 3 together to the recipient.

13.4.1.2 For decrypting the hybrid ciphertext

1. Now decrypt the encrypted random key with the receiver's private key.
2. Then, decrypt the encrypted data with the decrypted random key.

13.4.2 Operations involved in proposed algorithm

The proposed algorithm involves the following four operations:

- SetUp(): This operation is used to generate the master's key or shared key which will be used to encrypt the generated random key.

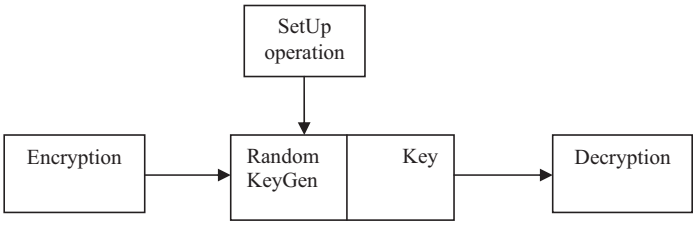


FIGURE 13–6 Enhanced randomized encryption.

- KeyGen()or Rand():This operation involves the generation of keys (random key or session key) for performing the encryption and decryption of data, respectively.
- Encrypt (): This operation involves the main function, that is, the encryption of the data or message which is in the form of a block cipher as well as the random key.
- Decrypt (): Finally, this operation is used to decrypt the encrypted data and the encrypted random key (Fig. 13−6).

First, the SetUp() operation is used to generate the master key or shared key for encryption and decryption purposes and then the KeyGen operation is performed, which involves the algorithms for generating a random key which is then encrypted by the shared key.

13.4.3 Random number generator

Our first step involves the generation of a random key, therefore this requires a large number of random numbers which are needed to be strongly secure (Dadheech et al., 2019). The best solution for this would be to make use of *pseudorandom sequence generators* or *cryptographically secure pseudorandom number generators*.

The pseudorandom sequence generator produces a random sequence which is a uniformly distributed stream sequence. Moreover, while using such generators, one thing that should be taken into consideration is that the sequence should be nonperiodic in nature. Examples of pseudorandom generators include the Linear Congruently Generators and Mersenne Twisters.

On the other hand, the latter generators, that is, the cryptographically secure pseudorandom generator provides some more features suitable for enhancing security. It involves uses such as one time pads, nonce, etc. It makes use of a pseudorandom-sequence generator in order to generate a random key for our proposed encryption technique. In this way, a random key is generated using a Linear Congruential Generator (LCG), which is a pseudorandom-sequence generator.

To define the generator, a relation is used known as the recurrence relation. This relation is shown as below:

$$X_{n+1} = (pX_n + q)(\text{mod } z)$$

where X represents the pseudorandom sequence values, and the following constants define the generator as:

z, $0 < m$ is the modulus
p, $0 < p < z$ is the multiplier
q, $0 < q < z$ is the increment
X_o, $0 < X_o < z$ is the seed or salt value

Now, z defines the limit of a period in general LCG which means that the LCG generator has periods up to z and even less than this value for some p choices. Moreover, for the full period of LCG having q as nonzero, there are some conditions:

1. q and z should be relatively prime.
2. P−1 is divisible by all the prime factors of z.
3. Also, p−1 should be a multiple of 4 if z is a multiple of 4.

By using the LCG pseudorandom sequence generator, a large number of random numbers will be generated and then be used to create a random key.

A key of variable size is required for encrypting the random key that has been generated by a pseudorandom sequence generator in our proposed randomized encryption technique (Kumar, Dadheech, Beniwal, Agarwal, & Patidar, 2020a), generating a key-pair out of a large number of size Z such that this large number is considered as a modulus for the public and private key and is then calculated by multiplying the two different prime numbers, namely, x and y, that is, $Z = x^*y$. Now the Euler's totient function is calculated as $O(Z) = (x\text{-}1)(y\text{-}1)$. After the totient function is computed, now select an integer for p, such that it lies in the range of [1, O(Z)], that is, $1 < p < O(Z)$, and also p and O(Z) have only One as a common factor. Here, the exponent of the public key is p. After defining the public key exponent then the exponent of the private key is defined for satisfying the congruency $q^*p = 1 \bmod O(Z)$. In this way, the public key is obtained as (Z, p) and the private key is obtained as (Z, q) and all these values must be kept secret.

13.4.4 Algorithm for the generation of a key-pair

1. Select two different random prime numbers as x and y.
2. Calculate $Z = x^*y$, where Z is considered as the modulus for the public and private key.
3. Calculate the Euler's totient function, O (Z) as

$$O(Z) = (x - 1) * (y - 1)$$

4. Select an integer for p, such that $1 < p < O(Z)$, and also p and O (Z) have only One as a common factor. This integer p is the exponent of the public key.
5. Now calculate q in order to satisfy the congruency as $q^*p = 1 \bmod O(Z)$ where q is defined as the exponent of the private key.
6. Finally, get the public key as (Z, p) and private key as (Z, q) and all these values must remain secret.

After generating the public key and the private key by using the above algorithm and the random key by using the pseudorandom sequence generator algorithm, now the main task

is performed (Kumar, Dadheech, & Chaudhary, 2020b), that is, the encryption of the user's data by a random key using Advanced Encryption Standard (AES) encryption technique and the encryption of that random key by the shared key (receiver's public key) using the RSA encryption technique. Therefore the encryption algorithms involved are described as follows:

13.4.4.1 Encryption of data by random key using Advanced Encryption Standard encryption technique

The second step of our proposed encryption technique involves the encryption of user's data by a random key. Since this random key is used only one time for encrypting the message, it is also referred to as ***one-time message key***. The encryption of the message is performed by using the AES algorithm as follows: (Fig. 13−7)

Thus performing the AES encryption technique over a block of data (an email message as an example), obtains the encrypted message in the following form:

Eq. (13.1), where M_{Sym} refers to the message or the plain text to be encrypted.

$$C_{Sym} = E_{Sym}\left(M_{Sym}, K_{random}\right) \tag{13.1}$$

K_{random} refers to the random key used to encrypt the message M_{Sym}.

C_{Sym} refers to the cipher text obtained by symmetrically encrypting the message M_{Sym} with the random key using AES symmetric encryption technique.

AES encryption technique is a symmetric encryption technique which requires the use of a single key or a private key for both the encryption and decryption of data and is also referred to as private key encryption. It is a faster method to encrypt the data which is based on the substitution−permutation network. The AES encryption method is basically an extension of data encryption standard (DES), that is, data encryption standard technique, and it involves a number of rounds of transformation which are determined by the size of the key. It means that how many times these rounds of transformations should be repeated, would depend upon the size of the key used in the method. For example, if we use the key having a size of 128-bits, then the repetition cycles used would be 10. Similarly, for the key size of 192-bits, there are 12 repetition cycles to be used, and for the key size of 256-bits, it would be around 14 repetition cycles. By choosing the size of the key according to the security needs, the appropriate numbers of rounds are then performed in order to encrypt the plain text and similarly, for decrypting the ciphertext back into the plain-text, the same numbers of transformation rounds are performed in the reverse order.

FIGURE 13–7 Encryption of data using Advanced Encryption Standard method.

The basic steps involved in the AES encryption technique are as follows:

Step 1: Key Expansion in which the key is expanded by generating the round keys from the cipher key schedule mechanisms.

Step 2: Initial round in which the AddRoundKey step is performed. It means that in the initial round, the round key is added with each state's byte with the help of an X-OR operation.

Step 3: Rounds in which four different steps are performed. These are SubBytes, ShiftRows, MixColumns, AddRoundKey

Step 4: Final round is similar to the step 3 rounds, but the only difference is that in the final round, the MixColumn step is not required. Thus the three substeps are: SubBytes, ShiftRows, AddRoundKey

The main reason for using AES rather than DES is that the larger size of the key (such as 128, 192, and 256 bits) and the maximum number of rounds for encrypting the data are used in the AES method by which it becomes strongly secure against any chosen ciphertext attack. In this way, the message is encrypted using the AES encryption method by the random key generated by a pseudorandom sequence generator, as shown in Eq. (13.1).

13.4.4.2 Encryption of random key by a shared key (or recipient's public key) using RSA asymmetric encryption technique

The key-pair, which involves a public−private key pair, is generated by using the key generation algorithm of the RSA technique, as stated above. Now it's the time to encrypt the random key (used to encrypt the data or message) by the shared key, that is, recipient's public key, using the RSA encryption technique. This can be shown in Fig. 13−8.

Therefore the encryption algorithm of the random key involves Eq. (13.2), where

$$E'_{Asym} = (K_{random})^{e} (\text{mod } Z) \tag{13.2}$$

E'_{Asym} refers to the ciphered random key (or encrypted random key) which is obtained by encrypting the random key by a shared (or public key) using the RSA asymmetric encryption technique. The reason behind using the *RSA Encryption Algorithm* here is the fact that it possesses the property of increased robustness in comparison with other encryption algorithms and is considered to be the toughest algorithm for cracking the various standards of encryption that are used for providing strong security in transmitting and storing the data. Hence it is safer in use than other encryption algorithms (Uma Maheshwari, Ratna Sirisha, & Sreenivas, 2018).

FIGURE 13–8 Encryption of random key using the RSA method.

The RSA encryption algorithm that has been used in our approach is a type of encryption which performs encryption on the block of data such that the block is obtained by breaking a chunk of data into a number of blocks of data. Moreover, it describes the *factorization concept* that causes the harder decryption of block cipher and also the *longer-sized keys* ranging from size 1024−2048 bits which can't be easily cracked by brute force attack.

The working of the RSA algorithm constitutes the multiplication of two numbers, say A and B, by a third number, say C, and it must be taken into consideration that the number C has been used by the one who already knows C, as the other person who tries to calculate the two numbers A and B without knowing the number C will face many difficulties. This is because the number C has a much larger value in RSA, thereby causing the tough calculations. Some protocols that support RSA for providing security are Pretty Good Privacy for securing email, IPSEC for securing the IP data, and Transport Layer Security/SSL for securing the data transported.

Since the user's data and the random key both are encrypted using the symmetric key encryption technique and the asymmetric key encryption technique, now according to the step 4 of our enhanced encryption algorithm as described above, send both the encrypted data and the encrypted key together to the recipient (Fig. 13−9).

This would be done by applying the concatenation operation over the Eqs. (13.1) and (13.2), respectively, as follows:

Concatenation of Eqs. (13.1) and (13.2) gives the following hybrid ciphered or encrypted result:

$$\begin{gathered} E_{Hybrid} = E_{Sym}\left(M_{Sym}, K_{random}\right)||E'_{Asym} \\ OR \\ C_{Hybrid} = C_{Sym}||C_{Asym} \end{gathered} \tag{13.3}$$

Therefore in this way, the encryption of user's data along with the encryption of the random key is performed in our proposed encryption technique using the concept of randomization.

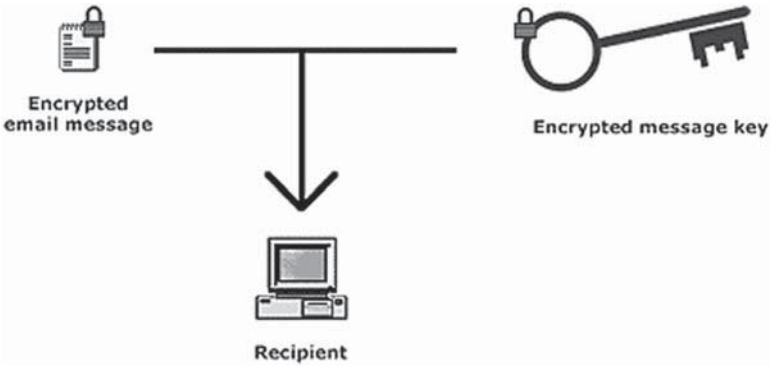


FIGURE 13–9 Sending of encrypted data and the encrypted key to recipient.

13.4.5 Decryption algorithm

Finally, the decryption of the encrypted result will be performed as soon as the signature made by the sender matches the signature by the receiver so as to verify the authenticity of the message. This would be performed as follows:

- Initially, the signature S, the encrypted message and the public−private key pair are sent to the receiver.
- Now, the random key is decrypted first using the receiver's public key, and then the message is decrypted by the decrypted random key. This is shown as follows:

$$\begin{aligned} \text{Decrypted Random key, } K_{random} &= (K_C)^d \pmod{Z} \text{ and,} \\ \text{Decrypted Message, } M_{Sym} &= D\left[C_{Sym}, K_{random}\right] \end{aligned}$$

- Signature is calculated using the sender's public key (Z, p), such that:

$$S1 = m^d \bmod Z$$

- Finally, two signatures S and S1 are matched, and if the match is successful, that is, S = S1, then the message is accepted, otherwise it is discarded.

In this way, the whole process of encryption and decryption of the message along with its random key is carried out as proposed by our randomized enhanced encryption technique and is shown by a suitable diagram as below: (Figs. 13−10 and 13−11)

13.4.6 Implementation of enhanced encryption technique in a cloud computing environment

Scenario 1: Embedded within a service

In this scenario, the randomized encryption scheme is incorporated within the already existing service of the cloud by the CSP. This allows the sharing of data among its users within the environment of a particular service. By embedding this scheme within a particular service of the cloud, it is completely integrated within the cloud environment itself. As soon as the users register themselves for the service, they get their decryption key.

The interaction within the given scenario is represented in Fig. 13−12, which shows that the encryption of data is performed by the service users (Enc. User) after putting data to the service, according to some access policies chosen by them. On the other hand, the availability of KA (Key Authority) within the service allows the service users to make their own access policy and through these access policies, the Enc. Users define that the CSP is also allowed to access the encrypted data. In this way, by embedding within the already existing service of the cloud, the randomized encryption scheme will be used within the cloud for transforming data more securely.

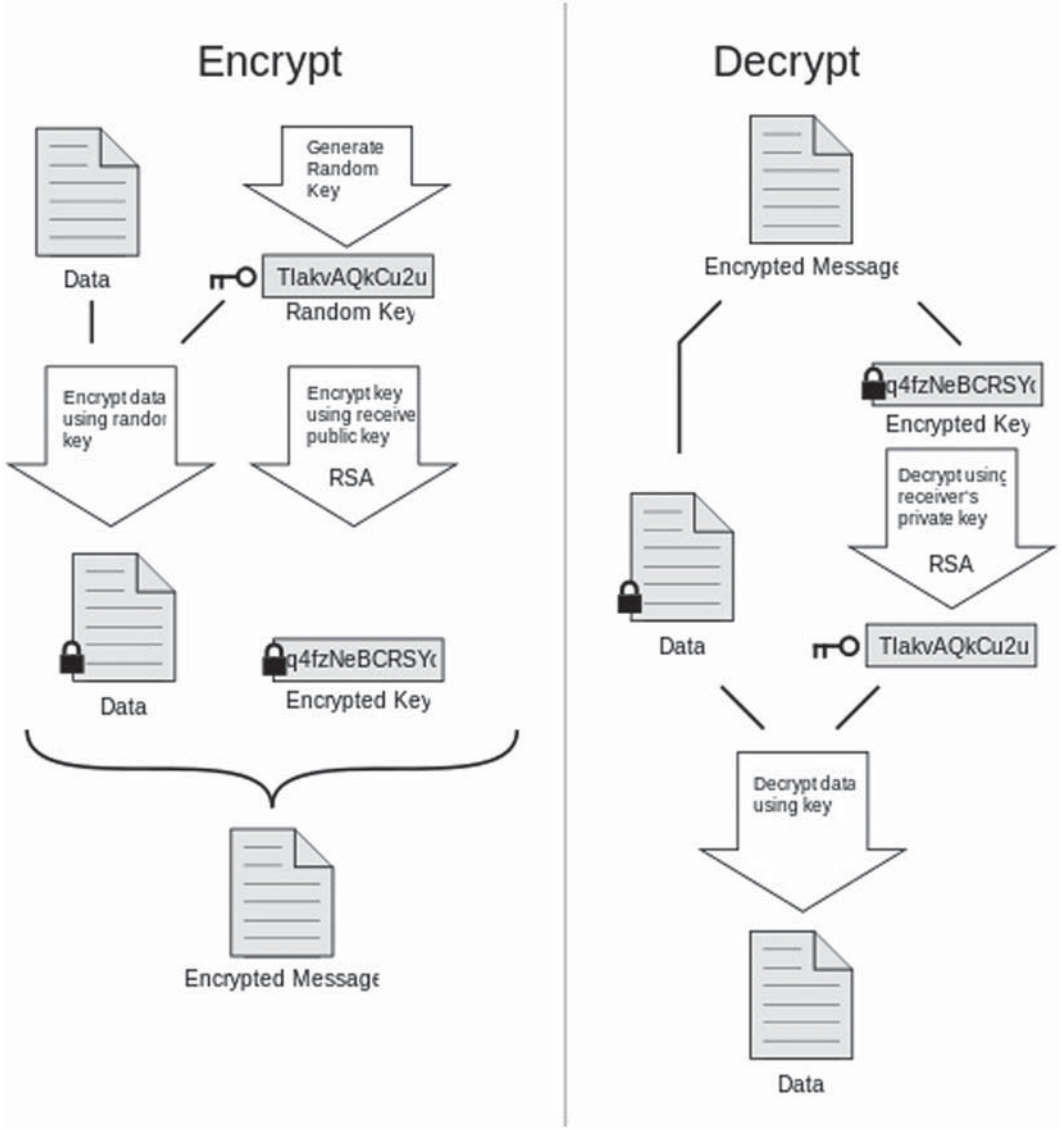


FIGURE 13–10 Proposed randomized encryption technique.

But the problem may arise in this scenario when the service users talk about making the trust in CSP. This is because the messages of the service users' can be decrypted by CSP's own constructed decryption keys as it has the key authority embedded within the service and hence according to their trust on CSP, it should not be malicious in this regard. Thus in order to maintain their trust in CSP, the development of some privacy policies and agreements based on the service-level is required (Alsharo, 2017).

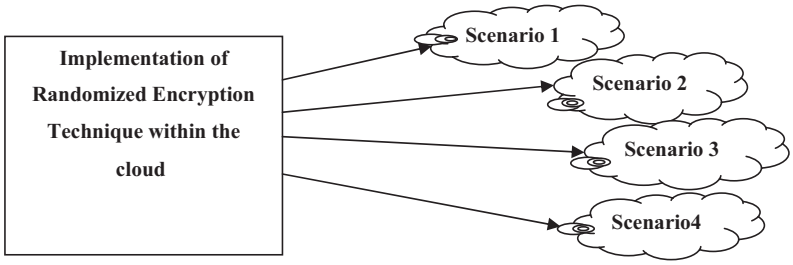


FIGURE 13–11 Different scenarios within cloud.

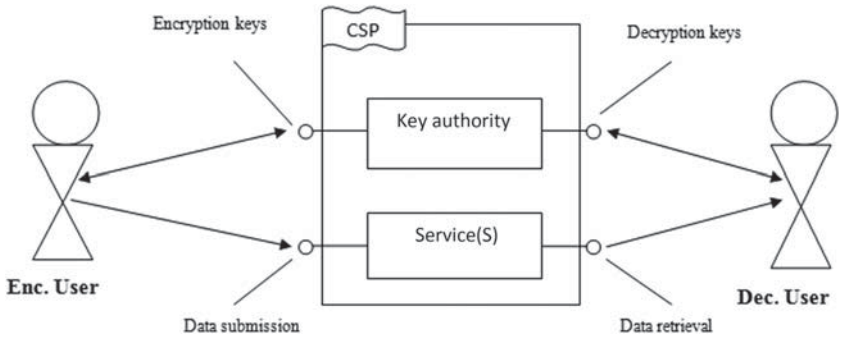


FIGURE 13–12 Outline of scenario 1.

Scenario 2: Randomized encryption scheme-as-a-service

Scenario 2 differs to scenario 1 in the sense that the proposed scheme is used as a service itself in this scenario rather than inculcating within the already existing service of the cloud as shown in scenario 1 above. It means that by acting as a separate service in the cloud, the proposed scheme provides its users with the interacting facility with other services of the cloud as well. Therefore the issue regarding key management can be reduced in this way.

On the other hand, this scenario is considered as an example of Security-as-a-Service and hence plays an essential role in providing protection at both the SaaS level as well as the PaaS level services.

The following figure describes the interactions that are performed within this scenario:

In this scenario, since the user comes to know that the plain text can be accessed by only those who are capable of satisfying the access policy, so the data is transmitted by the user during interaction with other services. Moreover, in this scenario, the CSP is treated like any other user by the service users. Also, a CSP in this scenario doesn't have the KA, therefore they have no authority to make their own decryption keys, and thus there is no issue regarding the trust upon the CSP in order to get service from it.

Scenario 3: Applying randomized encryption scheme within the database

Here comes the third scenario for incorporating the proposed randomized encryption scheme within the cloud during the data migration process. The main aim of using this

scenario is to allow the encryption of data within the cloud's database as soon as the data is placed into it. The further interaction within the given scenario is represented in Fig. 13−13. (Fig. 13−14)

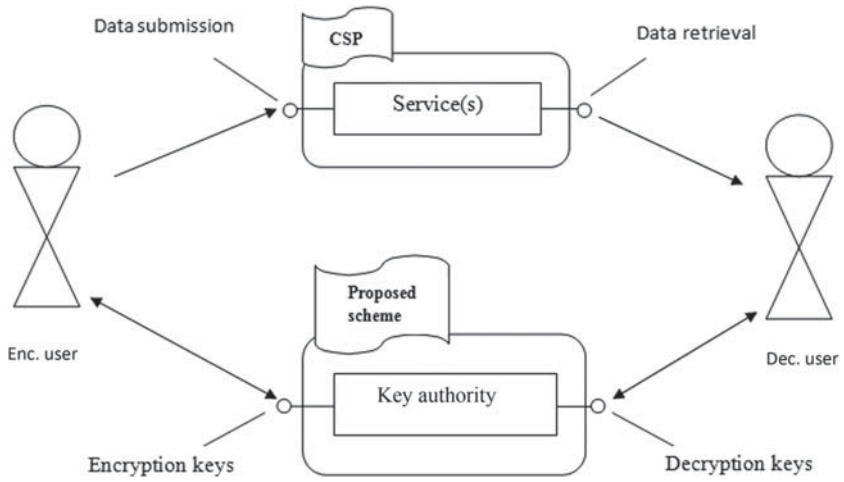


FIGURE 13–13 Outline of scenario 2.

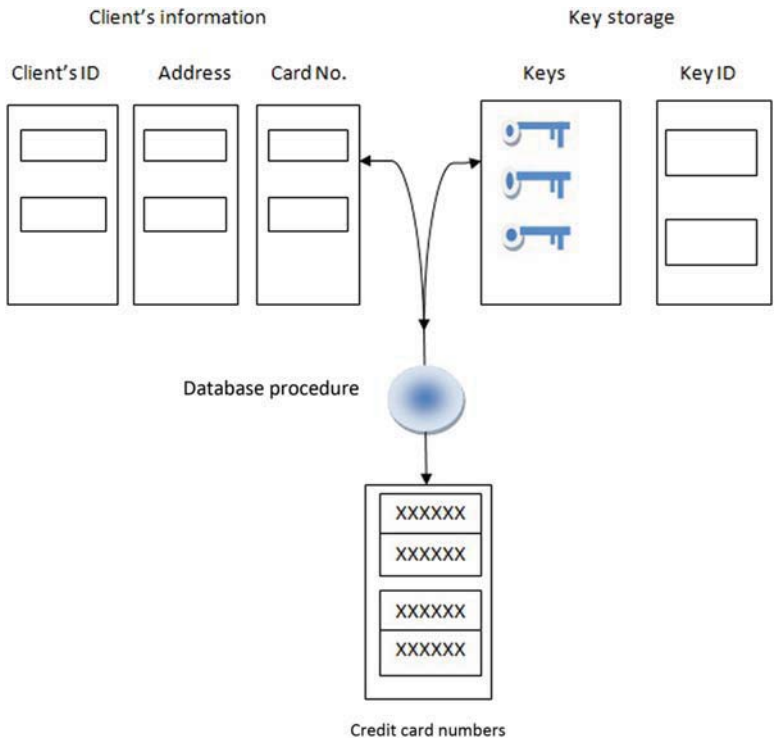


FIGURE 13–14 Outline of scenario 3.

Here, in this scenario, when the data enters into the database, it is encrypted through a "Database Procedure Call" which is responsible for discovering the key that is supplied to the database in the restricted table and then it is queried by the procedure. After discovering and querying it, the encryption key is then used by the procedure of the database to perform an encryption algorithm and gives out the final encrypted output. The main benefit of using this scenario is that it doesn't affect the applications of the cloud while performing a randomized encryption scheme within the database. But it is also lacking in that it involves the extra processing load that leads to the degradation in performance, and there is no security key management provided while using this kind of scenario.

Scenario 4: Applying randomized encryption technique outside the database

The fourth scenario focuses on performing the enhanced encryption (or randomized encryption) scheme outside the database, or it involves the encryption of information within the application such that it may be affected due to any change or modification to this application (Fig. 13−15).

In this scenario, the encryption server is developed and utilized so as to offer centralized services of encryption to the database. This server performs various cryptographic operations that an application requests. Unlike in scenario 3, the encryption keys are separated from the encrypted data in this scenario, since the keys are kept in the encryption server, and the data is inserted into the database. In this way, the given scenario offers the best security

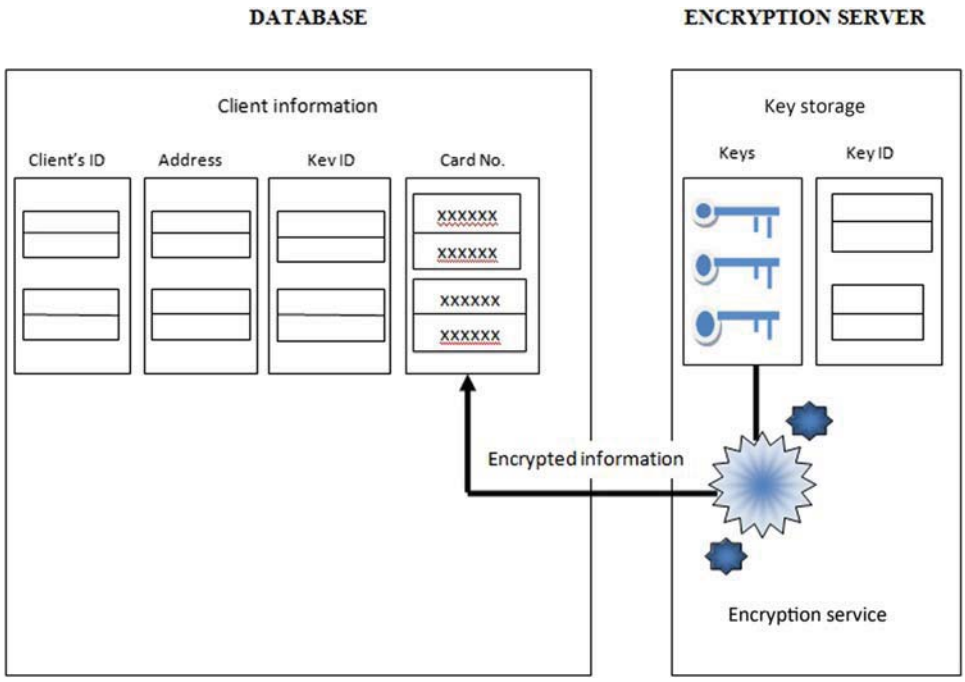


FIGURE 13–15 Outline of scenario 4.

management of the key by providing a layer of protection between the database and the attacker due to the availability of the encryption server. Other benefits of this scenario are:

- High performance
- Role of administrator is separated
- End-to-end encryption from client to encryption server is provided.

These are the different scenarios as discussed above that can be used within the cloud with our proposed randomized encryption technique so as to improve security while migrating data to the cloud.

13.5 Result and conclusion

This chapter aimed at improving the security of data within the cloud when the data is migrated from one source to cloud, or vice versa, using an enhanced randomized encryption technique.

During the analysis of providing security to a large amount of data in the cloud environment using various encryption techniques, it was formulated that the asymmetric algorithms are incapable of encrypting data in bulk or in large amounts when used singly. This is because of the large number of difficult mathematical computations of asymmetric algorithms which make the encryption procedure quite complex and lengthy. Hence, to overcome this problem, an ineffective asymmetric algorithm (such as RSA) combines with an efficient symmetric algorithm (such as AES) along with the concept of randomization such that the symmetric algorithm (AES) will be used to encrypt/decrypt the bulk of data and the asymmetric algorithm (RSA) will be used to encrypt/decrypt the short key (i.e., a random or session key).

Here for the experimental results, consider some useful data amounts required for performing the enhanced encryption technique, which are in the form of the key size and the data or message size to be encrypted and decrypted by using the proposed algorithm of encryption as discussed in this chapter. Therefore the requirements for performing an experiment are as follows:

13.5.1 Data required for performing Advanced Encryption Standard algorithm are

1. The random key size that has been generated by using a pseudo-random generator: 32- bits
2. Message length or size of data to be encrypted: 32- bits or 16 bytes.

13.5.2 Data required for performing RSA algorithm are

Key-pair: 1024 bits (for designing public key and private key of RSA algorithm)

Now, by taking all these data, an experiment will be performed to implement the enhanced randomized encryption technique within the cloud environment as follows:

1. Initially, a random key of 32 bits is used to encrypt the message or data of size equals to 32 bits using the symmetric AES encryption algorithm.
2. After encrypting the data of 32 bits, the random key of size 32 bits is then encrypted by the recipient's public key using the asymmetric RSA encryption technique.
3. Now, the encrypted data and the encrypted random key are then concatenated or combined together and stored in the CSP, such that when the user requests the encrypted result, then it will be delivered to the users/clients by the CSP.
4. Here, the user or client also is provided with a public key of RSA encryption algorithm by the CSP so as to store their data and hence this key is shared by both the users and the cloud service provider, whereas the random key is used only once for encrypting the data and then can be removed or destroyed. In this experiment, when the user wants their encrypted data, then the CSP first checks for its authentication and then delivers the encrypted result to its real user.

Figs. 13−16 and 13−17 are snapshots of the outcomes obtained while implementing the proposed encryption algorithm using MATLAB software

Similarly, the decryption process will be carried out in which the ciphered random key will be decrypted first by the RSA method using the recipient's private key. Then this decrypted random key will be used further to decrypt the ciphered text using AES method.

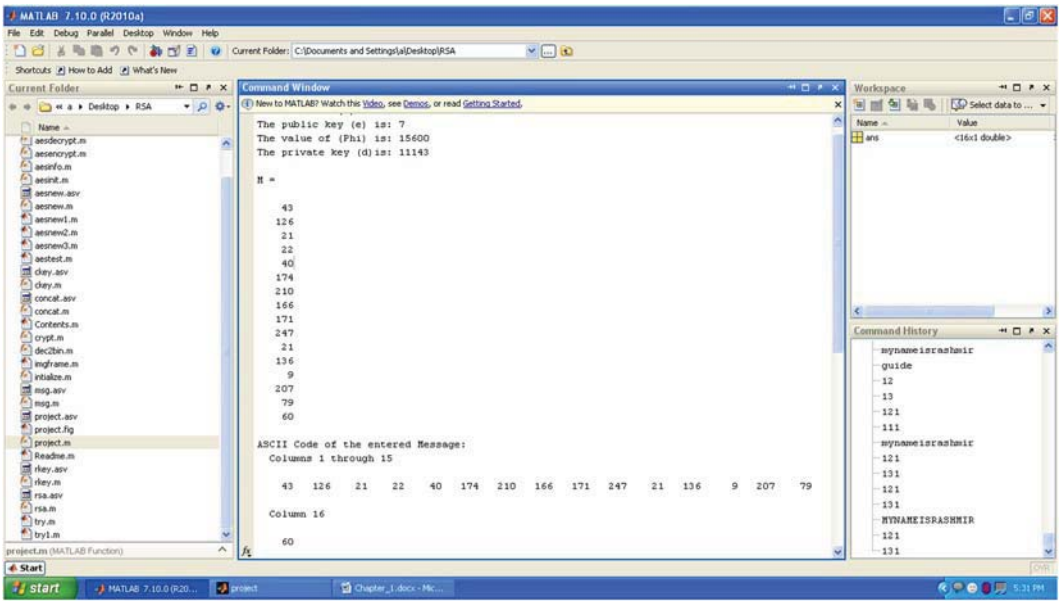


FIGURE 13–16 Snapshot of entering a random key as input for RSA encryption.

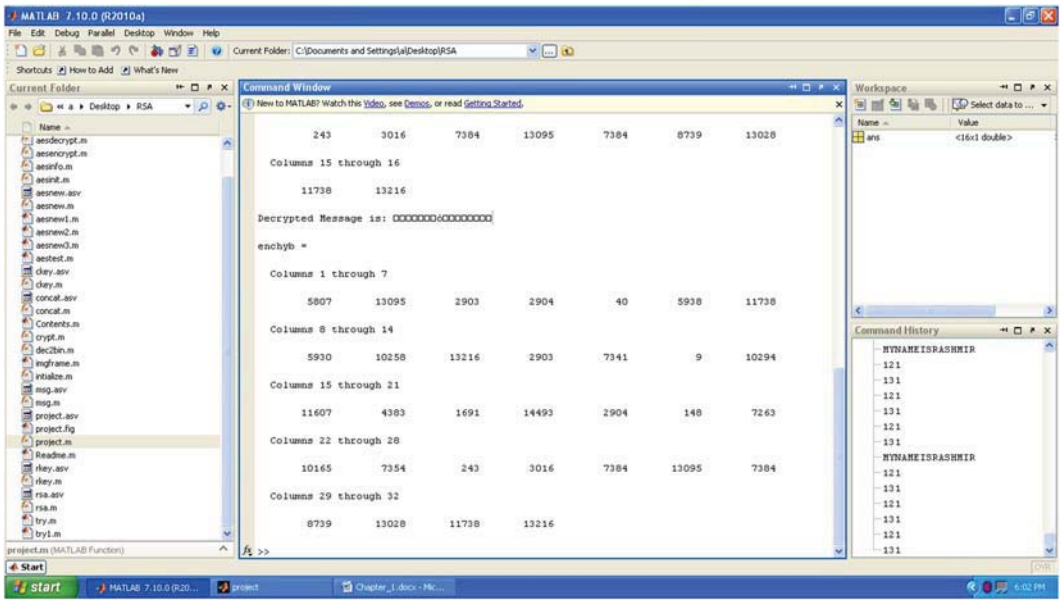


FIGURE 13–17 Snapshot of concatenated final encrypted output by combining encrypted text and encrypted random key.

In this way, this experiment aimed to provide confidentiality, the integrity of data, and the authentication of the origin of data. That is, the information or data should remain private while being migrated, the integrity of data should be maintained so as to verify whether it has been attacked by an intruder or not, and the origin of the data is authenticated so as to know from where the data came. Also, there are more computations involved in the proposed randomized encryption (Hybrid) technique in comparison to AES or RSA singly. Hence, it has been concluded that it will take time to encrypt the data as compared to the time taken by either AES or RSA alone and thus it will be challenging for cryptanalysis to break the randomized encryption (Hybrid) technique.

Moreover, it has been concluded that the enhanced (Hybrid) randomized encryption technique can be used easily and efficiently for providing electronic security as compared to other traditional encryption techniques. Electronic communication involves online banking, shopping on the internet, and email systems, which can be made strongly secure by using the proposed enhanced randomized encryption technique.

References

Alsharo, M. (2017). Attitudes towards cloud computing adoption in emerging economies. *International Journal of Cloud Applications and Computing, 7*(3), 44−58. Available from: https://doi.org/10.4018/ijcac.2017070102.

Bermbach, D., & Tai, S. (2014, March). Benchmarking eventual consistency: Lessons learned from long-term experimental studies. In *Proceedings of the IEEE international conference on cloud engineering* (pp. 47−56).

Coelho F., Paulo J., Vilaça R., Pereira J. & Oliveira R. (2017). HTAP bench. In *Proceedings of the eighth ACM/SPEC on international conference on performance engineering - ICPE '17*. Available from: https://doi.org/10.1145/3030207.3030228.

Dadheech, P., Goyal, D., Srivastava, S., & Kumar, A. (2018). A scalable data processing using Hadoop & MapReduce for big data. *Journal of Advanced Research in Dynamical and Control Systems*, *10*, 2099−2109.

Dadheech, P., Kumar, A., Choudhary, C., Beniwal, M. K., Dogiwal, S. R., & Agarwal, B. (2019). An enhanced 4-way technique using cookies for robust authentication process in wireless network. *Journal of Statistics and Management Systems*, *22*(4), 773−782. Available from: https://doi.org/10.1080/09720510.2019.1609557.

Domaschka, J., Griesinger, F., Baur, D., & Rossini, A. (2015). Beyond mere application structure thoughts on the future of cloud orchestration tools. *Procedia Computer Science*, *68*, 151−162. Available from: https://doi.org/10.1016/j.procs.2015.09.231.

Grolinger, K., Higashino, W., Tiwari, A., & Capretz, M. (2013). Data management in cloud environments: NoSQL and NewSQL data stores. *Journal of Cloud Computing: Advances, Systems and Applications*, *2*(1), 22. Available from: https://doi.org/10.1186/2192-113x-2-22.

Hong, T., Yao, J., Liu, C., & Qi, F. (2018). mmWave measurement of RF reflectors for 5G green communications. *Wireless Communications and Mobile Computing*, *2018*, 1−10. Available from: https://doi.org/10.1155/2018/8217839.

Ibrahim, H., Aburukba, R., & El-Fakih, K. (2018). An integer linear programming model and adaptive genetic algorithm approach to minimize energy consumption of cloud computing data centers. *Computers and Electrical Engineering*, *67*, 551−565. Available from: https://doi.org/10.1016/j.compeleceng.2018.02.028.

Kaur, S., & Aggarwal, M. (2018). Extended balanced scheduler with clustering and replication for data intensive scientific workflow applications in cloud computing. *Journal of Electronic Research and Application*, *2*(3). Available from: https://doi.org/10.26689/jera.v2i3.380.

Krintz, C. (2013). The AppScale cloud platform: Enabling portable, scalable web application deployment. *IEEE Internet Computing*, *17*(2), 72−75. Available from: https://doi.org/10.1109/mic.2013.38.

Kuhlenkamp, J., Klems, M., & Röss, O. (2014). Benchmarking scalability and elasticity of distributed database systems. *Proceedings of the VLDB Endowment*, *7*(12), 1219−1230. Available from: https://doi.org/10.14778/2732977.2732995.

Kumar A. & Sinha M. (2014, March). Overview on vehicular ad hoc network and its security issues. In *Proceedings of the international conference on computing for sustainable global development (INDIACom)* (pp. 792−797).

Kumar, A., & Sinha, M. (2019a). Design and analysis of an improved AODV protocol for black hole and flooding attack in vehicular ad-hoc network (VANET). *Journal of Discrete Mathematical Sciences and Cryptography*, *22*(4), 453−463. Available from: https://doi.org/10.1080/09720529.2019.1637151.

Kumar, A., & Sinha, M. (2019b). Design and development of new framework for detection and mitigation of wormhole and black hole attacks in VANET. *Journal of Statistics and Management Systems*, *22*(4), 753−761. Available from: https://doi.org/10.1080/09720510.2019.1609555.

Kumar, A., Dadheech, P., Beniwal, M. K., Agarwal, B., & Patidar, P. K. (2020a). *A Fuzzy logic-based control system for detection and mitigation of blackhole attack in vehicular ad hoc network. Microservices in Big Data Analytics* (pp. 163−178). Singapore: Springer.

Kumar A., Dadheech P. & Chaudhary U. (2020b, February). Energy conservation in WSN: A review of current techniques. In *Proceedings of the third international conference on emerging technologies in computer engineering: machine learning and internet of things (ICETCE)* (pp. 1−8).

Kumar, A., Dadheech, P., Kumari, R., & Singh, V. (2019a). An enhanced energy efficient routing protocol for VANET using special cross over in genetic algorithm. *Journal of Statistics and Management Systems*, *22*(7), 1349−1364. Available from: https://doi.org/10.1080/09720510.2019.1618519.

Kumar, A., Dadheech, P., Singh, V., Poonia, R., & Raja, L. (2019b). An improved quantum key distribution protocol for verification. *Journal of Discrete Mathematical Sciences and Cryptography*, *22*(4), 491−498. Available from: https://doi.org/10.1080/09720529.2019.1637153.

Kumar, A., Dadheech, P., Singh, V., Raja, L., & Poonia, R. (2019c). An enhanced quantum key distribution protocol for security authentication. *Journal of Discrete Mathematical Sciences and Cryptography*, *22*(4), 499−507. Available from: https://doi.org/10.1080/09720529.2019.1637154.

Kumar, A., Goyal, D., & Dadheech, P. (2018). A novel framework for performance optimization of routing protocol in VANET network. *Journal of Advanced Research in Dynamical and Control Systems, 10,* 2110−2121.

Lan, K., Fong, S., Song, W., Vasilakos, A., & Millham, R. (2017). Self-adaptive pre-processing methodology for big data stream mining in internet of things environmental sensor monitoring. *Symmetry*, *9*(10), 244. Available from: https://doi.org/10.3390/sym9100244.

Li, J., Yan, H., & Zhang, Y. (2019). Efficient identity-based provable multi-copy data possession in multi-cloud storage. *IEEE Transactions on Cloud Computing*, 1. Available from: https://doi.org/10.1109/tcc.2019.2929045.

Li, S., Zhai, D., Du, P., & Han, T. (2018). Energy-efficient task offloading, load balancing, and resource allocation in mobile edge computing enabled IoT networks. *Science China Information Sciences*, *62*(2). Available from: https://doi.org/10.1007/s11432-017-9440-x.

Pathan, R., Voudouris, P., & Stenstrom, P. (2018). Scheduling parallel real-time recurrent tasks on multicore platforms. *IEEE Transactions on Parallel and Distributed Systems*, *29*(4), 915−928. Available from: https://doi.org/10.1109/tpds.2017.2777449.

Uma Maheshwari, P., Ratna Sirisha, M., & Sreenivas, V. (2018). A panoramic design of cloud computing implementation elastic utility computing architecture. *International Journal of Modern Trends in Engineering and Research*, *5*(2), 79−84. Available from: https://doi.org/10.21884/ijmter.2018.5040.01lxk.

Whaiduzzaman, M., Naveed, A., & Gani, A. (2018). MobiCoRE: Mobile device based cloudlet resource enhancement for optimal task response. *IEEE Transactions on Services Computing*, *11*(1), 144−154. Available from: https://doi.org/10.1109/tsc.2016.2564407.

Yonghong, L., Na, Z., & Guofeng, Z. (2016). Cloud storage optimization analysis with energy consumption and life cycle. *International Journal of Grid and Distributed Computing*, *9*(10), 43−52. Available from: https://doi.org/10.14257/ijgdc.2016.9.10.04.

Zhang, J., Chen, J., Luo, J., & Song, A. (2016a). Efficient location-aware data placement for data-intensive applications in geo-distributed scientific data centers. *Tsinghua Science and Technology*, *21*(5), 471−481. Available from: https://doi.org/10.1109/tst.2016.7590316.

Zhang, W., Tan, S., Xia, F., Chen, X., Li, Z., Lu, Q., ... Yang, S. (2016b). A survey on decision making for task migration in mobile cloud environments. *Personal and Ubiquitous Computing*, *20*(3), 295−309. Available from: https://doi.org/10.1007/s00779-016-0915-y.

This page intentionally left blank