Secure and Optimized Algorithm for Implementation of Digital Signature

Isha Jauhari[1], Jitendra Kumar[2]

[1]Lecturer, Computer Science and Information Technology Department
Krishna Institute of Management and Technology, Moradabad, Uttar Pradesh, India

[2]Associate Professor, Computer Science and Engineering Department
IFTM University, Moradabad, Uttar Pradesh, India

Abstract: ***This Research work is based on a very important aspect of today's world i.e. Cryptography. Cryptography consists of protecting information by transforming it (also known as encryption) into a scribbled format, called cipher text. Only those who own a secret key can decode (also known as decryption) the message into original (plain) text. Sometimes encrypted messages can also be broken by cryptanalysis although modern cryptography techniques are virtually unbreakable. In this paper we have tried to implement the concept of a very popular field in Cryptography known as Digital Signature in a new and efficient manner. Since message authentication can only protect two parties from the third interfering party but it cannot protect the two parties against each other. So, Digital Signature has been introduced to provide a means for Message Authentication, Non-Repudiation and to check Integrity of the message after it has been received by the receiver. We studied various algorithms that are currently available to solve the problem of Digital Signature implementation, and after the analysis we found out that more efficient and secure algorithm can be developed by using positive aspects of various algorithms with some enhancements and by doing some good technical implementations in them. Proposed algorithm mainly focuses on reducing the time required for digitally signing of the document simultaneously providing enough security. After designing of the algorithm in the best possible manner we came out with many positive results and found that the proposed algorithm is feasible and valuable in creating and verifying the digital signature.***

Keywords: message digest, hash function, padding, parsing

1. Introduction

Cryptography is the technique of providing security of information communicated between individuals or group of individuals. Cryptography can be defined as the process in which data is converted into a scrambled code that can be again converted to a readable form and sent across a public or private network. It guards against unauthorized individuals or group of individual by preventing unauthorized modification of use. It uses a cryptographic system to convert a plaintext into a cipher text, every time using a key. Message authentication is a term used in cryptography which can protect the two parties who exchanges messages from any third party. However, it cannot protect the two parties against each other. So, in situations where there is not complete trust between the sender & receiver, something more than authentication is required. The most important solution to this problem is Digital Signature [1].

There are various algorithms used for digital signature. For creating any digital signature two types of algorithms are used one is for calculating the message digest and the second one is for encrypting the message digest (or digital signature).There is a long list of hash algorithms such as MD5,SHA-1,SHA 256,SHA-384,SHA-512 etc [2]. In this paper we proposed an algorithm which less processing to be done and hence the time reduces to a large extent. The proposed All these algorithms works on a single dimensional array i.e. they work by creating blocks of message forming an array which makes the work more tedious and time consuming so large number of processing need to be made for the security of hash function. So, we have reduced all these complications in our algorithm. We have done the whole calculation for finding the hash function in two-dimensional array.

This paper firstly describes the working of the digital signature after that a brief literature review on digital signature algorithms is presented. The next section introduces related problem after that proposed ranking approach is discussed in detail. Proposed algorithm uses the two dimensional structure for computing hash of the message and some modifications in the encryption algorithms for creating the digital signature. Finally, the last section shows some results with the help of tables.

2. Digital Signature

A Digital Signature is a construct which helps achieve non-repudiation of who signs it assures that he is the author of the document or the message that was signed [3]. A digital signature can also be used to verify that information has not been altered after it was signed. Digital signatures ensures authentication by relying on certain types of encryption. Digital signature is a sort of Cryptography. This digital signature creation process consists of two steps:

1) Calculate The Message Digest

In the first step of the process, a hash-value of the message (often called the message digest) is calculated by applying some cryptographic hashing algorithm (e.g. MD2, MD4, MD5, SHA1, or other). The hash value of a message which is the result after calculation is a fixed length sequence of bits,, extracted in some manner from the message.

2) Calculate The Digital Signature

In the second step in which the message is signed digitally, the information obtained in the first step i.e. the calculated hash-value of the message (the message digest) is encrypted with the private key of the person who signs the message and thus digital signature is obtained which is the encrypted hash-value. For this purpose, there are some algorithm based on mathematical formulas also called cryptographic encrypting algorithm for calculating digital signatures from given message digest is used .Often, obtained digital signature is attached to the message in a special format to be verified later if it is necessary. The above process is shown in Fig given below.

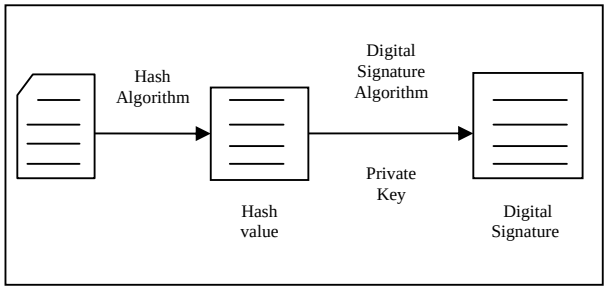

Figure 1: Digital Signing

After this the digital signature verification process is carried out. In this process again two steps are involved:

3) Calculate The Current Hash-Value

In the first step, the calculation for the hash-value of the signed message has been done. For this calculation, the same hashing algorithm is used as was used during the signing process. Since the obtained hash-value is calculated from the current state of the message, it is called the current hash-value.

4) Calculate The Original Hash-Value

In the second step in which the verification of digital signature is done, the decryption of digital signature has been carried out with the same encryption algorithm that was used during the signing process. At this time, the public key that corresponds to the private key used during the signing of the message is used for decryption. As a result, the original hash-value has been obtained that was calculated from the original message during the first step of the signing process (the original message digests). The above two steps are shown in Fig 2.

There are various purposes for which digital signature are used:

1. Signer Authentication
2. Message Authentication
3. Non-Repudiation
4. Integrity

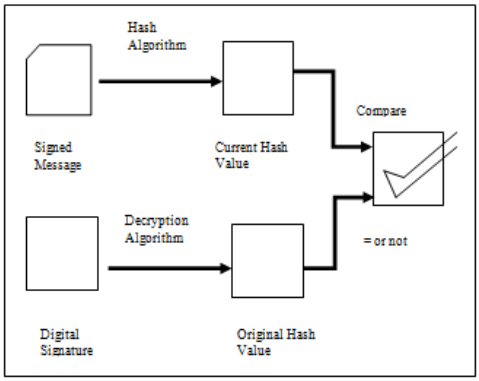

Figure 2: Verification of Digital Signature

3. Related work

3.1 Mc-Eliece Based Digital Signature Scheme

McEliece is one of the oldest known public key cryptosystems. The McEliece cryptographic scheme **is** based on error correcting codes. It consists in randomly adding errors to a codeword (as it would happen in a noisy channel) and uses this as a cipher. There are many shortcomings with this scheme. (a) It generates the Digital Signatures of 81-bits. Hence security factor is only of pow (2, 83) (b) There exists a problem of decoding an error correcting code called Syndrome Decoding (SD) and (c) In many cases, code-based cryptosystems like McEliece do not allow practical digital signatures [2].

3.2 DSA

A digital signature algorithm is used by a signatory to generate a digital signature on data and by a verifier to verify the authenticity of the signature. There is a public and private key owned by each signatory. The signature generation process uses private key and the signature verification process uses the public key [4]. The only known way to attack it is to perform a “brute-force" attack on the modulus. This attack can be easily defeated by simply increasing the key size. However, this technique can lead to a number of serious problems such as increased processing time, computational overhead and increased key storage requirement.

3.3 Elliptic Curve DSA

This approach is a variant of the Digital Signature Algorithm (DSA) which uses Elliptic curve Cryptography each other. Today’s most of the search engine works on this technique. The bit size of the public key believed to be needed for ECDSA is about twice the size of the security level, in bits. ECDSA has not been proven to be existentially non-forgeable against chosen message attack. Signature Generation in ECDSA is also quite complex. Various types of attacks are possible on ECDSA. They are: (a) Attacks on the elliptic curve discrete logarithm problem (b) Attacks on Attacks on the Hash function employed (c) Duplicate Signature key selection.

3.4 Morph Digital Signature

This signature is designed for a distributed usability. In this, The Distributor can manage and morph any document according to the specific transaction (e.g. price cost). So some authorized parts can be freely masked or blinded by the distributor without soliciting any producer intervention. In this manner, the consumer will own a sub-document from the original one, which only contains corresponding part for her transaction. The limitations of this algorithm are (a) The Distributor has the possibility to blind all the DP parts or a set of sub- parts inside the DP part. Therefore, it may be possible that he/she can forge the original document (b) Moreover; each DP can contain some other DP. Therefore, it may be somewhat difficult to verify the original message digest of the document.

3.5 The Korean Certificate-Based DSA

KCDSA is a variant of ElGamal, similar to DSA, and it is designed by incorporating several features from the recent cryptographic research and thus is believed to be secure and robust. But it has some disadvantages too: (a) It is prone to brute force attack (b) In this; public key is validated by means of a certificate issued by some trusted third party authority. Therefore, it may be possible that third party may issue wrong public key intentionally or unintentionally (c) It also requires a collision-resistant hash function which is very difficult and complex to produce. (d) In this, hash code is variable, it is not fixed.

4. Problem Description

Earlier all the hash algorithms like SHA-1, SHA-256, SHA-384, and SHA-512 works on a single dimensional array which makes the work more tedious and time consuming. The work was done on single dimension so large number of processing need to be made for the security of hash function. So, we have reduced all these complications in our algorithm. We have done the whole calculation for finding the hash function in two- dimensional array. Thus, it helped in decreasing the processing overhead and thus reduces the time taken to compute hash to a large extent. We have also studied various block ciphers such as Blowfish, DES, and Triple DES and tried to improve the processing time of proposed algorithm by adopting their methodology and their good characteristics and by using their ideas to some extent and incorporating some new ideas The algorithm we have designed uses the concept of private key cryptography for key generation because Public-key cryptography is relatively slow and is only suitable for encrypting small amounts of information while private key cryptography is much faster and is suitable for encrypting large amounts of information. Also we used hexadecimal S-box which implements a much faster processing while encryption.

5. Proposed Digital Signature Algorithm

In general, a digital signature scheme consists of two algorithms:

a) Algorithm for computing message digest (Hash function)

b) Algorithm for creating and verifying digital signature. (Encryption-Decryption algorithm)

5.1 Proposed Hash Algorithm

A hash function can be defined as an algorithm which takes as input the original message of any length and produces as output a fixed length string. Hash functions are also known as Digital Fingerprints. Examples of some popular hash algorithms are MD5 and Secure hash algorithm (SHA-1, SHA-2 etc…) [5]. A hash function H () should have the following properties:

1. **One-way:** Given any message m, it is easy to compute H (m). However the reverse is computationally infeasible.

2. **Collision resistant:** It is computationally infeasible to find m2 = m1 such that H (m1) = H (m2) , given any message m1.

3. **Strong collision resistant:** It is computationally infeasible to find m1 = m2 such that H (m1) = H (m2).

Hash computation consists of the following procedure as shown by the figure given below (See Fig. 3).

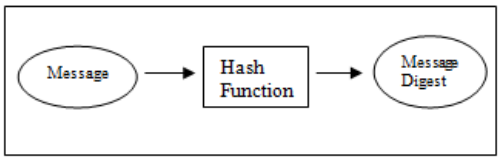

Figure 3: Computing Hash

So, for calculating the hash of the message we can use the Proposed Hash Algorithm which is highly influenced by the Secure Hash Algorithm (SHA-1, SHA-256, SHA-384 and SHA-512).This algorithm is divided into two phases:

5.1.1 Preprocessing

In this stage, firstly the message is padded then it is parsed into m-bit blocks and after that the initialization values are set which are to be used for hash computation.

5.1.2 Hash Computation

The hash computation generates a message schedule from the padded message and make use of that schedule, along with word operations , constants and functions to iteratively generate a series of hash values. The message digest can be determined from the final hash value generated by the hash computation. Proposed Algorithm for Hash Computation is as follows:

INPUT: A message x of length $l \geq 0$.

OUTPUT: A 128 bit (16-bytes) hash code of x.

Step 1: Take an array named text [] which will contain the input message or file of any length.

Step 2: Take a pointer variable *hash in which the final message digest will be stored.

Step 3: Define variables such as i, j to be used in for loop and initialize variables such as posi1=0 and posi2=0 which indicate the corresponding positions.

Step 4: Now two matrices are initialized named state1 [4] [4] and state2 [4] [4] which will contain the values contained in variable hash and in the input message respectively.

Step 5: Now for every 16-bytes block of the input message consider the following implementation procedure:

```
Void hash (char text [], char *hash)
{
```

```
int i,j,posi1=0,posi2=0;
for ( i=0;i<4;i++)
{
for ( j=0;j<4;j++)
{
state2 [i][j] = text[posi1];
posi1++;
}
}
for ( i=0;i<4;i++)
{
for ( j=0;j<4;j++)
{
state1 [i][j] = hash[posi2];
posi2++;
}
}

int k=0;
for (i=0;i<4;i++)
{
for (j=0;j<4;j++)
{
hash[k]=state2[i][j]^state1[i][j];
if (k<8)
rotateleft(hash[k],k);
else
rotateright(hash[k],k);
                                k++;
                                }
                                }}
```

Description: The whole message is divided into a number of blocks. Each block is of 16-bytes.The operation for computing hash is applied on each 16-byte block. Firstly, we store the whole message contained in an array named text [] into a 4x4 matrix state2 [4][4].Each position of state2[4][4] matrix contains the value at the corresponding position of text[] which is incremented by a counter named posi1.The same procedure is applied for storing the values contained in the variable named hash into another 4x4 matrix named state1[4][4] by incrementing another counter named posi2.

Now we take a random variable *k* which serves as the counter in the next for loop used for the calculation of message digest. In this loop each and every value of matrix state2 [4] [4] at position[i] [j] is XORed with the value of matrix state1 [4] [4] at the matching position and stored in hash[k] and thereby incrementing k. Now we perform circular left shift on the values stored in hash[k] until k<8(can be any number) i.e. for the two rows of the earlier matrices and circular right shift for the next two rows and thereby finding the message digest of the original message. This whole process is shown by the diagram below (See Fig. 4).

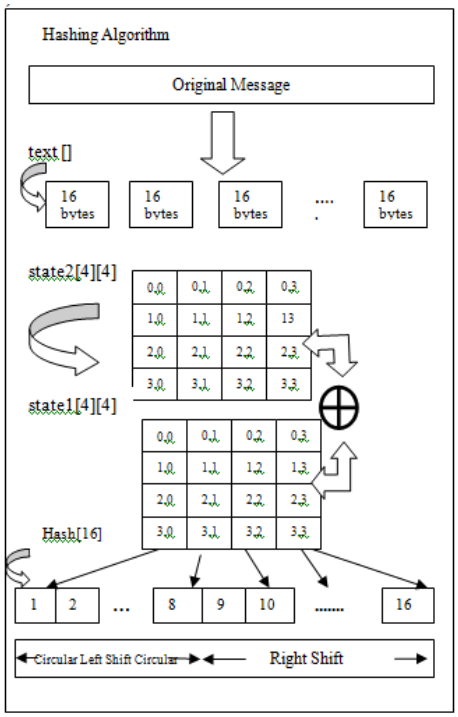

Figure 4: Proposed Hashing Algorithm

After that the next 16 bytes of the message will be processed with the same operation.

5.2 Enhanced Encryption Standard

Proposed Algorithm for Creating and Verifying Digital Signature also acknowledged as *ENHANCED ENCRYPTION STANDARD (EES)* is as follows:

This algorithm that we are using derives its ideas from the earlier block ciphers and consists of some modifications in them. It specifies a symmetric block cipher that can process data blocks of 128 bits, using cipher key with length of 128 bits. The algorithm consists of ten rounds of encryption. First the 128-bit key is expanded into eleven so-called round keys, each of them 128 bits in size [6]. Each round includes a transformation using the corresponding cipher key to ensure the security of the encryption. After an initial round, during which the round key is XORed to the plaintext (Per-Round Key Operation), ten equally structured rounds follow. The following operations are carried out in each round [7]:

1) Replace bytes: This operation is a non-linear byte substitution which operates independently on each state byte using a substitution table (S-box). This invertible S-box, is constructed by composing two transformations:

a) Take the multiplicative inverse; the element {00} is mapped to itself.
b) Apply an affine transformation which can be found in Rijndael documentation.

Since S-box does not depend on any input pre-calculated forms are used. Each byte of the state is then substituted by the value in the S-Box whose index corresponds to the value in the state [8, 9]: a (i,j) = SBox[a(i,j)]
The same operation which is the inverse of Replace Bytes is performed, using the inversed S-Box, which is also pre calculated.

2) Reallocate rows: As implied by its name, the Reallocate rows operation processes different rows. A simple rotate is performed with a different rotate width. There is a shift of one byte position to the left in the second row, two byte positions to the left in the third row and three byte positions to the left in the fourth row of the 4x4 byte input data (the state) matrix. The first row is not changed.

3) Per round key: In the PerRound_Key () transformation, a Round Key is added to the State by a simple bitwise XOR operation. Each Round Key consists of 4 words from the key schedule. Those 4 words are each added into the columns of the State, the application of the PerRound_Key () transformation to the 10 rounds of the Cipher occurs. The inverse of Shift Row is the same cyclically shift but to the right. It will be required later for deciphering.

5.2. 1 Algorithm Specification:

For this algorithm**,** the length of the input block, the output block and the State is 128 bits. And the number of 32-bit words (number of columns) in the State. For this algorithm, the length of the Cipher Key, K, is 128 bits. The key length reflects the number of 32-bit words (number of columns) in the Cipher Key.

5.2. 2 Inputs and Outputs:

The input and output for the algorithm each consist of sequences of 128 bits (digits having values 0 or 1). These sequences will sometimes be referred to as blocks and the number of bits they contain will be referred to as their length.

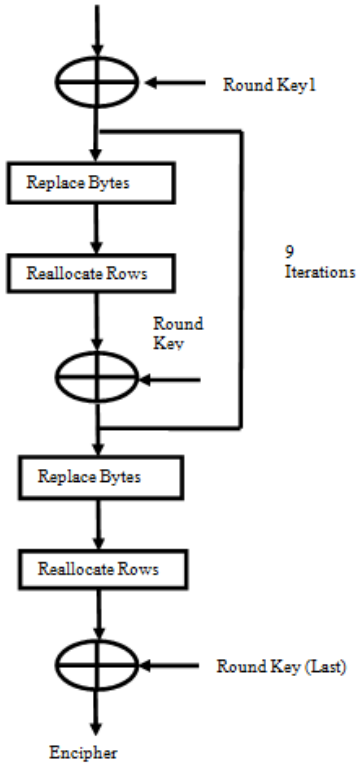

Figure 5: Enhanced Encryption Algorithm

The Cipher Key for the algorithm is a sequence of 128 bits. Other input, output and Cipher Key lengths are not permitted by this standard. The bits will be numbered starting at zero within such sequences and ending at one less than the sequence length (block length or key length).

5.2. 3 The State

Internally, the algorithm's operations are performed on a two-dimensional array of bytes called the State. There are four rows of bytes (block length divided by 32) in the State. In this State array denoted by the symbol s each individual byte has two indices, with its column number c in the range $0< c < 4$ and its row number r in the range $0 < r < 4$. At the start of the Cipher and Inverse Cipher, the input – the array of bytes in0, in1… in15 – is copied into the State array. The inverse Cipher or Cipher operations are then conducted on this State array, after which the final value of the state is copied to the output.

5.2. 4 Cipher

At the start of the Cipher, the input is copied to the State array using the conventions described above. After an initial addition of the Round-Key, the State array is transformed by the implementation of a round function 10 times. The final State is then copied to the output .The round function is parameterized using a key schedule that consists of a one-dimensional array of four-byte words derived using the Key Expansion routine. The individual transformations– Rep_Bytes (), Realloc_Rows (), PerRound_Key () process the State. Thus instead of using too many operations for encryption decryption process we adopted only three operations for signing the document thus providing enough security and simultaneously reducing the time taken for creating and verifying the digital signature.

6. Experiments and Results

In this work, implementation of proposed algorithm is done for the creation of digital signature in as much as reduced time and providing enough security against cryptanalysts attacks. For performance evaluation of proposed algorithm we took files of varying sizes and types (PDF files, Video files, Audio files, and Text files), signed, verified and analyzed the time taken by this algorithm and found that it takes less time as compared with algorithms such as data encryption standard as well as it is more secure than these algorithms since it contains more operation and its based on the theory of confusion and diffusion.

Details of results obtained by proposed algorithm are as follows:

Earlier algorithms takes huge amount of time for implementing digital signature, our algorithm implements digital signature in almost one-fourth time as taken by the DES and this is shown in following tables. We have shown the result considering 4 types of files for creation of digital signature and same amount of time is also taken at the receiver end for verification. In these tables we have shown time corresponding to various size files.

6.1 PDF Files

Table 1: For PDF files

S. No	Size of File (KB/MB)	Time taken (min:sec:hund)

1	824KB	0.007395833
2	6.87MB	1:31:01
3	19.1MB	0.182951389
4	54.3MB	9:07:44

6.2 Image Files

Table 2: For Image files

S. No	Size of File (KB/MB)	Time taken (min:sec:hund)
1	566KB	0:05:11
2	1.42MB	0:13:10
3	4.56MB	0:42:30
4	6.92MB	1:07:06

6.3 Audio Files

Table 3: For Audio Files

S. No	Size of File (KB/MB)	Time taken (min:sec:hund)
1	1.38KB	0.009814815
2	9.93MB	1:45:18
3	22.1MB	4:11:50
4	41.3MB	0.302488426

6.4 Video Files

Table 4: For Video Files

S. No	Size of File (KB/MB)	Time taken (min:sec:hund)
1	3.85MB	0.027303241
2	21.7MB	0.153032407
3	33.1MB	0.217025463
4	77.3MB	16:37:45

7. Conclusion

We have developed a digital signature implementation algorithm which exhibits processing power of high performance, efficiency and in minimum amount of time. In this we used private key cryptography for key generation because Public-key cryptography is relatively slow and is only suitable for encrypting small amounts of information while private key cryptography is much faster and is suitable for encrypting large amounts of information. Also we used hexadecimal S-box which implements a much faster processing while encryption and all the work are done in two-dimension rather than one-dimension. We have applied the most efficient way for providing the highest security but we can further increase by taking larger key size and by using the concept of certificate authority for the distribution of keys. And for the concept of multiuser we can use the concept of different keys at senders' or receivers' end but we have to compromise with the confidentiality.

8. Acknowledgment

We would like to thank the anonymous referees for their helpful comments and suggestions that have improved the quality of this manuscript.

References

[1] W.Stallings, "Cryptography and Network Security: Principles and Practices", Prentice Hall, 1999.

[2] Min-Shiang Hwang, Cheng-Chi Lee (2005), "Research Issues and Challenges for Multiple Digital Signatures", International Journal of Network Security.

[3] Swati Verma, and Birendra Kumar Sharma (2011),"A New Digital Signature Scheme Based on Two Hard Problems", International Journal of Pure and Applied Sciences and Technology.

[4] Swati Paliwal, Ravindra Gupta (Volume 3, Issue 2, February 2013), "A Review of Some Popular Encryption Techniques", International Journal of Advanced Research in Computer Science and Software Engineering.

[5] Erfaneh Noorouzi1,Amir Reza, Estakhrian Haghighi, Farzad Peyravi, Ahmad Khadem Zadeh(2011),"A New Digital Signature Algorithm", International Conference on Machine Learning and Computing IPCSIT vol.3 (2011) © (2011) IACSIT Press, Singapore.IPCSIT vol.3 (2011) © (2011) IACSIT Press, Singapore.

[6] V. Sumathy,C. Navaneethan (2012),"Enhanced AES Algorithm For Strong Encryption", International Journal of Advances in Engineering & Technology, Sept 2012.Johan Oudinet,Search Engine Ranking.Technical Report no0617, July 2007,revision 1126.

[7] Amit Chaturvedi, Damodar Tiwari (2012), "Analysis on AES Algorithm using symmetric cryptography".

[8] Gabriela Moise (2009), "A Survey on the Usage of Substitution Tables in DES and AES Algorithms, Petroleum –Gas University of Ploieşti, Informatics Department.

[9] Faiz Yousif Mohammad, Alaa Eldin Rohiem, Ashraf Diaa Elbayoumy (2009), "A Novel S-Box Of AES Algorithm Using Variable Mapping Technique", Aerospace Sciences & Aviation Technology.

Author Profile

Isha Jauhari received the B.Tech degree in Computer Science and Engineering from Moradabad Institute of Technology in 2010. She is working as a lecturer in Krishna Institute of Management & Technology, Moradabad since 2010.

Jitendra Kumar received the M.C.A. degree from AIT Greater Noida in 2001 and M.Tech degree in Computer Science and Engineering from B.B.D.N.I.T (UPTU) in 2007.Currently he is working as an associate professor in IFTM University, Moradabad.