Secured Outsourcing Towards a Cloud Computing Environment Based on DNA Cryptography

Hamza Hammami
University of Tunis El Manar
Faculty of Sciences of Tunis
LIPAH-LR11ES14, 2092 Tunis, Tunisia
Email:hamza.hammami@aol.fr

Hanen Brahmi
University of Tunis El Manar
Faculty of Sciences of Tunis
LIPAH-LR11ES14, 2092 Tunis, Tunisia
Email: hanen.brahmi@yahoo.fr

Sadok Ben Yahia
University of Tunis El Manar
Faculty of Sciences of Tunis
LIPAH-LR11ES14, 2092 Tunis, Tunisia
Email: Sadok.benyahia@fst.rnu.tn

***Abstract*—Cloud computing denotes an IT infrastructure where data and software are stored and processed remotely in a data center of a cloud provider, which are accessible via an Internet service. This new paradigm is increasingly reaching the ears of companies and has revolutionized the marketplace of today owing to several factors, in particular its cost-effective architectures covering transmission, storage and intensive data computing. However, like any new technology, the cloud computing technology brings new problems of security, which represents the main restrain on turning to this paradigm. For this reason, users are reluctant to resort to the cloud because of security and protection of private data as well as lack of trust in cloud service providers. The work in this paper allows the readers to familiarize themselves with the field of security in the cloud computing paradigm while suggesting our contribution in this context. The security schema we propose allowing a distant user to ensure a completely secure migration of all their data anywhere in the cloud through DNA cryptography. Carried out experiments showed that our security solution outperforms its competitors in terms of integrity and confidentiality of data.**

***Keywords*-Cloud computing; storage; trust; security; protection; DNA cryptography; integrity; confidentiality**

I. Introduction

Cloud computing is the fruits of recent IT developments. It is a new technology that appears today to be a satisfactory response to the problem of storing and computing data, which is encountered by companies. It proposes to ensure processing and hosting their digital information via an entirely outsourced infrastructure [1], which enables users to benefit from a lot of online services without having to worry about the technical aspects of their uses, while amortizing the costs generated by covering all these data.

The objective of the appearance of this paradigm is to process very intensive computer workloads, to supply very large installations for data storage, to provide flexible and highly-performing services and progressive data storage for a daily increasing number of users, to potentially reduce the management and utilization costs, and to satisfy the exploitation of the services and spaces of computer storage available at a provider by clients who are external companies. The methods used to finalize the service between a supplier and an end customer appear in four forms [2]: application, platform, infrastructure, and data. First, the application remains in direct contact with the client. Second, the platform realizes the application. Third, the infrastructure supports the platform. Finally, the data are delivered on request.

Therefore, cloud computing looks a big opportunity for companies, but like any new technology, cloud services bring new security issues. [3] Data flows across the IP networks can also carry threats to the system core. Malicious users can take advantage of these security weaknesses to penetrate the system, through the cloud services, and to access the confidential data of any company. Thus, the security problem seems to be a major issue for companies intending to expand in the cloud. It has become one of the main restrains on turning to the cloud. That is why users are unwilling to resort to the cloud due to the security and protection of private data, the operational difficulties or the inability to control information when leaving the perimeter, and the lack of confidence in cloud service providers.

In order to provide an effective solution for the security of the data of remote users against any analysis aimed at their disclosure when migrating to the cloud, we propose a security scheme allowing a remote user to ensure a completely secure migration of all their data anywhere in the cloud through DNA cryptography. The latter is a new and promising field for information security. Indeed, it is possible to benefit from the advantages of biological DNA to store and hide data inside it. In addition, secret information is placed in a molecule of DNA and hidden among other DNA molecules.

This article is organized as follows. First, section 2 sets out some approaches conducted to overcome this problem, as well as a comparative study of these approaches from which we try to inspire ourselves by way of being able to work out and draw up our contribution in this field. After that, section 3 describes the main idea of our proposed solution to resolve security problems. The results of experiments showing the usefulness of the proposed approach are presented in section 4. Finally, section 5 describes a general conclusion where an evaluation of the developed work is carried out and prospects are suggested.

II. State of the art

Several works have been devoted to finding solutions to deal with this problem such as: the authors in [4] addressed the issue of how to secure information outsourced to the cloud from any hacker who knows the encryption key

and has as a matter of fact access to most of cipher-text blocks. For that purpose, they proposed a novel and efficient Bastion scheme as a solution that guarantees data confidentiality despite the leakage of the encryption key and in spite of the hackers access to nearly all cipher-text blocks. It is sure that the proposed scheme ensures high confidentiality of data. However, this scheme does not verify the integrity of these data when migrating to the cloud for storage. In this case, an adversary can intercept the encrypted data, and then they can modify them without the owner's knowledge of these blocks.

Furthermore, the authors in [5] put forward a model that would improve the security of cloud data by the incorporation of different cryptographic techniques. The suggested model performed some steps on the data before storing it in the cloud. First, it encrypted the files with the RSA algorithm. Then a hash calculation was carried out on the encrypted data. This calculation was generated by the md5 algorithm in combination with the fingerprint of the owner of these data. After combining the RSA and MD5 algorithms, the data became resistant to access or modifications by any third party and to intruders of cloud storage system. However, one of the major limitations of this model was that the owner of these data was the only entity that would manage their keys, so it could become dangerous. Indeed, if this owner lost or forgot their keys, the encrypted data would be indecipherable and the unencrypted data would be definitively lost. Besides, the owner used the same public and private keys for both encryption and decryption of all data. In addition, during the users registration and login steps, the proposed model utilized a single-factor authentication which made it susceptible to several types of attacks such as the man-in-the-middle and brute-force attacks.

Moreover, the authors in [6] suggested a data security technique. It was called a Bidirectional DNA Encryption Algorithm (BDEA). The latter was used to overcome cloud data security problems and it could encrypt and decrypt Unicode characters with the utilization of ASCII characters. It provided as well two security levels. Firstly, a table maintained the DNA digital A, G, T, C coding as well as their corresponding binary value. After that, there was DNA coding encryption and decryption. Encryption generally started using a plaintext Unicode message. The latter was then converted to an ASCII code that was another time converted to a hexadecimal code. The latter was then converted by a binary convertor into a binary code. After that, this code was divided into various parts in a way that corresponding DNA digital loading was carried out like the BDEA algorithm utilized for a Unicode character set. As a result, a great number of cloud-service users could be reached easily. Nevertheless, the disadvantage of the proposed algorithm was that it did not guarantee the integrity of the data encoded in DNA coding. As such, data destruction, distortion or data corruption attacks were applied to the encrypted data.

Besides, the authors in [7] proposed three-way architecture protection schemes. To begin with, the Diffie-Hellman algorithm was utilized so as to generate keys for the step of exchanging keys. Then a digital signature was authenticated. Afterwards, the AES encryption algorithm encrypted or decrypted the data file of the user. The limitation of the proposed solution was that the Diffie-Hellman key exchange algorithm was vulnerable to the man-in- the-middle attack. The most serious limitation was the lack of authentication.

Moreover, the authors in [8] presented the basic concept of homomorphic encryption as well as different encryption algorithms. Paillier preserved in this context the additive property of homomorphic encryption. This would enable getting results of operations done through cipher-texts without having any knowledge regarding the raw data, while respecting data confidentiality. Yet, the drawback of the homomorphic encryption in the context of cloud computing was the high computation time and complexity.

Furthermore, the authors in [9] put forward a new secure cloud storage system in the aim of protecting the data of organizations against cloud providers, third-party auditors, and several utilizers who might use their old accounts in the purpose of accessing data stored on the cloud. In fact, this suggested system would enhance the authentication level of security through the use of a time-based one-time password in order to verify cloud users and the use of an automatic blocker protocol so that we can protect the system completely against any non-authorized third-party auditors. The limitation of this proposed system was that it was vulnerable to the man-in-the-middle attack. During this attack, an attacker could intercept all the communications between the different entities that constituted the suggested system, like the organization admin, the cloud user, the third party auditor, and the cloud service provider.

In addition, the authors in [10] proposed a genetic-security cloud method implementing encryption based on human genetics, more particularly on protein biosynthesis. The attractive coupling between the encryption of content and biosynthesis protects data against unauthorized access. However, one of the major drawbacks of this approach was the high computation time to encrypt and decrypt the data during their migration.

Several approaches have been devoted to find solutions to this problem, but each of these approaches is faced with one or more problems. For example, the proposed approaches in [4, 6] did not verify the integrity of data, so an attacker could intercept the encrypted data transmitted in the network, and can modify them without the data owners reaching their account. Furthermore, the proposed approaches in [5, 7, 9] were vulnerable to the man-in- the-middle attack, so an attacker could access information the remote data owners and the cloud were trying to transmit, and it could intercept the communications between these two parties in order to be able to spy on, capture and control it in complete transparency without the knowledge of external parties. The suggested approaches in [8, 10] required a high execution time so that the owners could encrypt the data before their migration to the cloud.

Thus, the approach put forward in [10] did not present any of the aforementioned defects, specially the attack of the man-in- the-middle. However, it necessitated the use of a high execution time to perform the encryption task. Consequently, we opt for this approach. Therefore, our approach will be based on the approach proposed in [10] in order to improve the performance in execution time. This approach will be detailed in the next section.

III. Our proposed approach

The main motivation of our approach is to put in place a solution to the problem of dematerialized storage to ensure the security of confidential data. In this section, we are reviewing the proposed security approach. We begin by identifying the different phases of data security for remote users when migrating from the original host (the data owner) to the hosts hosting that data (cloud computing). These phases are in the following order: (i) DNA encoding phase, (ii) encryption and migration phase. The details of each of these phases are as follows:

A. *Phase of DNA encoding*

First, in this step, we use a genetic-based coding method. This method consists in encoding the data of the remote user into DNA nucleotides forming genetic information. The type of coding used is that of the genetic codes of human beings. To be able to apply this type of coding, we follow the following steps:

- First, the user's data are transformed into an ASCII code giving a string of bits denoted "S0".
- The next step is to convert the "S0" string to an "S1" string of DNA. The latter consists of nucleotides indicated by the letters "A", "T", "C" and "G".

This conversion follows the next table:

Binary code	Genetic code
00	A
01	T
10	C
11	G

Table I
Conversion of binary code into DNA coding

B. *Encryption and migration phase*

Once the distant user's data are converted into genetic codes, the next step is to divide these genetic codes into blocks of 256 sub-blocks. Each sub-block is formed by four genetic codes. Subsequently, each sub-block will be encrypted by another sub-block. This encryption is done by an encryption Sbox generated to encrypt the 256 sub-blocks. Each Sbox is variable for each block. We note that this encryption process is not bijective; *i.e.*, there are several possible matches for each sub-block, and the attack by the frequency analysis is impossible.

Once the encryption step is performed, the next step is to substitute each encrypted sub-block with another sub-block. Thus, to be able to substitute a block of 256 sub-blocks, as illustrated previously as an example in Figure 1, there exist 256! possible choices. These 256! choices are as follows:

- To substitute the first block, there are 256 possible choices.
- To substitute the second one, there exist 255 possible choices.
- To substitute the third one, there are 254 possible choices.
- To substitute the last one, there is only one possibility.
- The final result is calculated as follows: 256 * 255 * 254 * ... * 1 = 256!

Figure 1 presents an example of a block of 256 sub-blocks.

1	AAAA	33	CAAA	65	GAAA	97	TAAA	129	AGAA	161	CGAA	193	GGAA	225	TGAA
2	AAAC	34	CAAC	66	GAAC	98	TAAC	130	AGAC	162	CGAC	194	GGAC	226	TGAC
3	AAAG	35	CAAG	67	GAAG	99	TAAG	131	AGAG	163	CGAG	195	GGAG	227	TGAG
4	AAAT	36	CAAT	68	GAAT	100	TAAT	132	AGAT	164	CGAT	196	GGAT	228	TGAT
5	AACA	37	CACA	69	GACA	101	TACA	133	AGCA	165	CGCA	197	GGCA	229	TGCA
6	AACC	38	CACC	70	GACC	102	TACC	134	AGCC	166	CGCC	198	GGCC	230	TGCC
7	AACG	39	CACG	71	GACG	103	TACG	135	AGCG	167	CGCG	199	GGCG	231	TGCG
8	AACT	40	CACT	72	GACT	104	TACT	136	AGCT	168	CGCT	200	GGCT	232	TGCT
9	AAGA	41	CAGA	73	GAGA	105	TAGA	137	AGGA	169	CGGA	201	GGGA	233	TGGA
10	AAGC	42	CAGC	74	GAGC	106	TAGC	138	AGGC	170	CGGC	202	GGGC	234	TGGC
11	AAGG	43	CAGG	75	GAGG	107	TAGG	139	AGGG	171	CGGG	203	GGGG	235	TGGG
12	AAGT	44	CAGT	76	GAGT	108	TAGT	140	AGGT	172	CGGT	204	GGGT	236	TGGT
13	AATA	45	CATA	77	GATA	109	TATA	141	AGTA	173	CGTA	205	GGTA	237	TGTA
14	AATC	46	CATC	78	GATC	110	TATC	142	AGTC	174	CGTC	206	GGTC	238	TGTC
15	AATG	47	CATG	79	GATG	111	TATG	143	AGTG	175	CGTG	207	GGTG	239	TGTG
16	AATT	48	CATT	80	GATT	112	TATT	144	AGTT	176	CGTT	208	GGTT	240	TGTT
17	ACAA	49	CCAA	81	GCAA	113	TCAA	145	ATAA	177	CTAA	209	GTAA	241	TTAA
18	ACAC	50	CCAC	82	GCAC	114	TCAC	146	ATAC	178	CTAC	210	GTAC	242	TTAC
19	ACAG	51	CCAG	83	GCAG	115	TCAG	147	ATAG	179	CTAG	211	GTAG	243	TTAG
20	ACAT	52	CCAT	84	GCAT	116	TCAT	148	ATAT	180	CTAT	212	GTAT	244	TTAT
21	ACCA	53	CCCA	85	GCCA	117	TCCA	149	ATCA	181	CTCA	213	GTCA	245	TTCA
22	ACCC	54	CCCC	86	GCCC	118	TCCC	150	ATCC	182	CTCC	214	GTCC	246	TTCC
23	ACCG	55	CCCG	87	GCCG	119	TCCG	151	ATCG	183	CTCG	215	GTCG	247	TTCG
24	ACCT	56	CCCT	88	GCCT	120	TCCT	152	ATCT	184	CTCT	216	GTCT	248	TTCT
25	ACGA	57	CCGA	89	GCGA	121	TCGA	153	ATGA	185	CTGA	217	GTGA	249	TTGA
26	ACGC	58	CCGC	90	GCGC	122	TCGC	154	ATGC	186	CTGC	218	GTGC	250	TTGC
27	ACGG	59	CCGG	91	GCGG	123	TCGG	155	ATGG	187	CTGG	219	GTGG	251	TTGG
28	ACGT	60	CCGT	92	GCGT	124	TCGT	156	ATGT	188	CTGT	220	GTGT	252	TTGT
29	ACTA	61	CCTA	93	GCTA	125	TCTA	157	ATTA	189	CTTA	221	GTTA	253	TTTA
30	ACTC	62	CCTC	94	GCTC	126	TCTC	158	ATTC	190	CTTC	222	GTTC	254	TTTC
31	ACTG	63	CCTG	95	GCTG	127	TCTG	159	ATTG	191	CTTG	223	GTTG	255	TTTG
32	ACTT	64	CCTT	96	GCTT	128	TCTT	160	ATTT	192	CTTT	224	GTTT	256	TTTT

Figure 1. An example of one block of 256 sub-blocks

According to statisticians who have done a study to calculate the elapsed time to test all possible cases as for 26! possible choices, with a computer able to test 1,000,000 keys per second, it would then take 12 million years to enumerate everything. Therefore, to be able to calculate the time necessary to enumerate all our 256! possible choices, we apply the rule of three. This same computer would then need the time:
*T = (256! * 12 million) / 26!* to enumerate all possible substitutions.

We can say from the last two steps (encryption and substitution) that our approach has a gigantic number of keys equal to $[(256^{n*256}) * (n * 256 !)]$, where (256^{n*256}) represents the space of keys used in the encryption step, *(n * 256 !)* is the space of keys utilized in the substitution step, and n stands for the number of used blocks.

Once the substitution step is complete, we will compute the *"HMAC"* hash function in combination with a *"trustkey"* on the encrypted and substituted data before being migrated to the cloud. We then lock the data and their HMAC code with the *"trustkey"* and send them to the cloud for storage. We note that, in this phase, we locked all the information transmitted to the cloud by a *"trustkey"*. This latter is generated between the distant user and cloud computing.

The generation of this key is created by exchanging messages on a non-secure channel. When this key is

set up, the messages are actually sent unencrypted on the network, and anyone who intercepts the transmitted messages cannot deduce the generated key. Thanks to this key, secure communication is guaranteed by the fact that a possible attacker, who intercepts communications between the distant user and the cloud, would have no way of locating the *"trustkey"* generated from the publically transmitted information. This key is generated using the Diffie Hellman key agreement with authentication algorithm [11]. The latter is an algorithm of exchanging an anonymous key that allows two pairs to establish a shared secret through a non-secure communication channel. This shared secret is directly utilized as a *"trustkey"*. The detailed algorithm is listed as follows:

- Prior to execution of the algorithm, the two parties original host and remote host each obtain a public/private key pair and a certificate for the public key.
- Original host chooses x, $0 \leq x \leq p - 2$, at random, sets $c := g^x$ and sends c to the remote host.
- Remote host chooses y, $0 \leq y \leq p - 2$, at random, computes the shared secret key $k = g^{xy}$, takes its secret key S_{RH} and signs the concatenation of g^x and g^y to get $S := Sign_{S_{RH}}(g^x \,||\, g^y)$. Then, it sends $(g^y, E_k(S))$ to the original host.
- Original host computes the shared key $k = g^{xy}$, decrypts $E_k(S)$ and verifies the remote host signature. If this verification succeeds, the original host is convinced that the opposite party is the remote host. It takes its secret key S_{OH}, generates the signature $s = Sign_{S_{OH}}(g^y \,||\, g^x)$ and sends $E_k(S)$ to the remote host.
- Remote host decrypts $E_k(S)$ and verifies the original host signature. If the verification succeeds, the remote host accepts that it actually shares k with the original host.

IV. Approach validation and results exploitation

This last section is reserved for the experimental part of our approach. It has two main parts. First, we begin by describing experimentally the performance of our approach to that proposed in [10] while showing the effectiveness of our approach in terms of execution time and in terms of resistance against the man-in- the-middle attack. Second, we validate our approach by evaluating its performance based on confidentiality, authentication, and data integrity.

A. *Tests and experimental results*

In this section, we show, on the one hand, the performance of our approach in terms of execution time compared to that proposed in [10] since it is in the same context as ours, and on the other hand, we describe the ability of our approach to stem the man-in- the-middle attack.

To show the performance in terms of execution time, we carry out an experiment during which we increase the size of the data migrating to the cloud in order to have the influence of this rise on the execution time. Figure 2 depicts the comparison of our approach with the approach suggested in [10] in terms of execution time.

The study of Figure 2 allows us to realize that our approach at the time of execution is less than the approach proposed in [10]. This is probably due to the fact that in the approach suggested in [10], the two phases of translation and fragmentation have a temporal complexity equal to O(n2), while our approach contains two phases of DNA encoding and encryption and migration. Each of these steps needs a temporal complexity equal to O(n). We can therefore conclude that our approach is better than that proposed in [10].

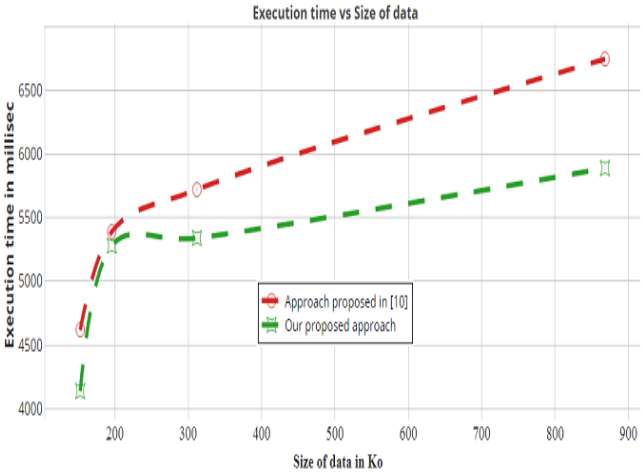

Figure 2. Comparison of our approach vs the approach proposed in[10]

Now let us return to the demonstration of the resistance of our approach against the man-in- the-middle attack. This latter is a type of cyber attack where a malevolent actor is inserted in a conversation between two parties that borrows the identity of both parties and accesses information that the latter two are trying to convey. With this attack, an attacker can intercept the communications between the originating host (the data owner) and the remote host (the cloud) so as to be able to spy on, capture and control it seamlessly without the knowledge of external parties. Figure 3 shows the attack scenario of the man- in- the- middle.

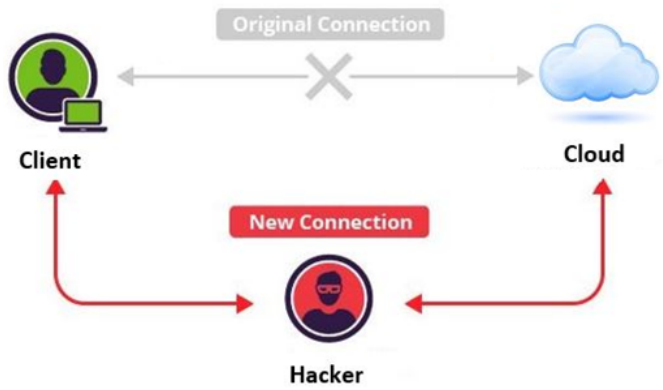

Figure 3. Man-in-the-middle attack

Firstly, in our approach a hacker can capture publicly exchanged information between a user and the cloud in order to generate a *trustKey*, but this hacker would have no means of locating the *trustKey* generated from publicly transmitted information. Nevertheless, the information exchanges very large numbers signed by the two correspondents. Thus, the *trustKey* generation technique resists the attack by the middle man because the hacker cannot send their own information instead of one of the information exchanged between the distant user and the cloud.

Secondly, a hacker can intercept the data of the user migrating to the cloud provider's data center. However, this hacker can not exploit anything because the data are at the same time locked by the trusted key *trustKey* and encrypted in the form of the nucleotides designated by the letters "A" "C" "G" and "T". Therefore, the hacker must have both the trusted key *trustKey*, all encryption Sboxes of each block of 256 sub-blocks, and all substitution Sboxes of each block of 256 sub-blocks, in order to unlock and decrypt the data migrating in the network. And even if this hacker has the trusted key, he can only see data encrypted in nucleotides. Furthermore, if the hacker has all the decryption and substitution Sboxes of each block of 256 encrypted sub-blocks, he can not decrypt them unless he has all the identifiers of all the blocks. These are scattered in the network, so it is impossible to have them all by the hacker. We can conclude, therefore, through the exposition of this discussion, that we can easily contain the man-in- the-middle attack.

B. Security Analysis

As in any information system, to ensure the security of the cloud paradigm, it is necessary to respond to the needs that the *"ISO*"* (International Standard Organization) organization has highlighted, in its studies, on the security of networks, which are confidentiality, integrity, authentication and availability. This subsection presents a brief overview of these security properties and discusses at the same time the validation of the proposed approach with respect to each of these properties:

Confidentiality: It is the ability to make a message incomprehensible, and therefore not accessible, to any attacker or potential spy. Only the author of the message and its recipients can access the content of the message. This property is assured by our approach thanks to the DNA cryptography, presented in section 3, during the encryption and migration phase. We recall that, in this phase, we carried out two processes: a first encryption process with a key space = (256^{n*256}), and a second substitution process applied to the results of the first process. The latter lasts *(n * 256 !)*. Hence, for a hacker to try to decrypt the data of users during their migration, they need, first of all, the *"trustkey"* to unlock them. Moreover, they will then require a number of keys equal to $[(256^{n*256})$ * *(n * 256 !)]* to enumerate all possible cases and to be able to subsequently decrypt them. We can confirm by this that our approach checks the property of data confidentiality.

Integrity: After verifying the confidentiality of cloud user data, notwithstanding, it may also be important for us to be able to ensure their integrity. This property guarantees that data from distant users have not been tampered with by unauthorized people between the time they were transmitted and the time they are received.

Using a calculation of a Hash Message Authentication Code (HMAC) function, in combination with a *"trustkey"* generated between both parties, the remote user and the cloud, allows us to verify the integrity of the data transmitted to the cloud after the step of substituting the encrypted data. Once the transmitted data are received by the cloud, this latter will first unlock the result of the hash calculation using the *"trustkey"*. After that, there will be a matching measure on the data from the time they were transmitted by the distant user and the time they are received by the cloud, to determine if they have been tampered with during their transmission to the cloud. In case the codes compared after the matching measure are different, there will be an alteration message. Then a retransmission request is automatically made. Consequently, we can affirm that our approach ensures that the data exchanged between the cloud and its users has not been tampered with by unauthorized utilizers.

Authentication: It is the ability to verify the identity of the data owner. Unlike the identification case, the author here must not only provide information relating to their identity but also give proof of that identity. In our approach, we verify this property by the encryption and migration phase. In the two-party communication case of the original and the remote hosts, data authentication can be achieved through a HMAC function in combination with a trustkey. During this communication, the two hosts share a secret key (*"trustkey"*) to compute a Hash Message Authentication Code (HMAC) of all migrated data. When the migrated data with a correct HMAC code arrives, the remote host knows that it must have been sent by the original host.

Given the study we have done in this section, we can say that the approach we have just proposed can face the man in the middle attack on users' data and can respond exhaustively and efficiently to all security needs.

V. Conclusion

The work in this paper has been to develop a security plan taking advantage of combining the benefits given by the recently appearing security mechanisms to provide sufficiently robust data confidentiality, while adapting easily to emerging and converging technologies. The solution we have presented in this paper contributes to the data security problem of the cloud paradigm. This solution has the following characteristics: *(i)* applying a biological system to secure the data of distant users, *(ii)* ensuring a protected migration of data in the form of data encrypted into DNA nucleotides forming genetic information, *(iii)* offering a mechanism based on the DNA test whose aim is to verify the integrity of migrated data during their reception, by

means of matching calculation between the hash result of the obtained data and that of the sent ones, and *(iv)* providing an efficient performance with regard to other previously studied security mechanisms as it ensures a completely secured hosting of their data in the cloud.

The various work presented in this paper can be summarized in the following points:

- A synthesis study of research in the literature on proposed security mechanisms for data security, especially data confidentiality management and access control in the cloud. Through this study, we have searched the recently published work to find a solution for this problem, and from which we are inspired to develop and present our contribution in this area.
- A Development of a security solution to adapt a robust encryption technology based on DNA cryptography in order to ensure the security of users' data migrating to the cloud to be hosted and to allow to securely access their data hosted in the cloud.
- Finally, we have completed our work by exposing some experiments from which we have tried to present the advantages and performances of our solution. Then we have validated our approach regarding security properties.

REFERENCES

[1] Ratten, V. (2015). "Cloud Computing Technology Innovation Advances : A Set of Research Propositions", International Journal of Cloud Applications and Computing, 5(1), 71-78.

[2] Zhangjie, F., S. Xingming, L. Qi, Z. Lu, & S. H. Jiangang, (2015). "Achieving efficient cloud search services: multi-keyword ranked search over encrypted cloud data supporting parallel computing". IEICE Transactions on Communications, 98(1), 190-200.

[3] Ali, M., S. U. Khan, & A. V. Vasilakos, (2015). "Security in cloud computing: Opportunities and challenges". Information Sciences, 305, 357-383.

[4] Karame, G. O., Soriente, C., Lichota, K., & Capkun, S. (2017). Securing Cloud Data under Key Exposure. IEEE Transactions on Cloud Computing.

[5] Chidambaram, N., Raj, P., Thenmozhi, K., & Amirtharajan, R. (2016). Enhancing the Security of Customer Data in Cloud Environments Using a Novel Digital Fingerprinting Technique. International Journal of Digital Multimedia Broadcasting, 2016, 1.

[6] Barkha, P. (2016, January). Implementation of DNA cryptography in cloud computing and using socket programming. In Computer Communication and Informatics (ICCCI), 2016 International Conference on (pp. 1-6). IEEE.

[7] Rewagad, P., & Pawar, Y. (2013, April). Use of digital signature with diffie hellman key exchange and AES encryption algorithm to enhance data security in cloud computing. In Communication Systems and Network Technologies (CSNT), 2013 International Conference on (pp. 437-439). IEEE.

[8] Msilini, N., Laouamer, L., Alaya, B., & Hamrouni, C. (2017). Homomorphic Cryptosystems for Securing Data in Public Cloud Computing. In Multimedia Forensics and Security (pp. 59-75). Springer International Publishing.

[9] El-Booz, S. A., Attiya, G., & El-Fishawy, N. (2016). A secure cloud storage system combining time-based one-time password and automatic blocker protocol. EURASIP Journal on Information Security, 2016(1), 13.

[10] Hammami, H., Brahmi, H., Brahmi, I., & Yahia, S. B. (2017). A Security Approach for Data Migration in Cloud Computing Based on Human Genetics. In European, Mediterranean, and Middle Eastern Conference on Information Systems (pp. 384-396). Springer, Cham.

[11] Choo, K. K. R., Nam, J., & Won, D. (2014). A mechanical approach to derive identity-based protocols from DiffieHellman-based protocols. Information Sciences, 281, 182-200.