Different types of policies can be implemented by security sites that control access to the servers for processing and storage. A distributed approach to managing security policy can be applied in this context [49][50].

The FRS allows strengthening the data confidentiality, integrity and availability [24][41][46]:

- Confidentiality: an intruder will have to collect all the fragments in the correct order before attempting cryptanalysis.
- Integrity and availability: an intruder should alter or destroy consistently all replicas of a fragment to be able to modify or destroy data.

The application of the FRS technique involves the following operations:

- Encryption before fragmentation;
- Secure management of the encryption key;
- Fragmentation of encrypted fragments of fixed size data;
- Secure naming of fragments;
- Fragments diffusion from the user site to all storage sites;
- Implementation of storage sites in a distributed algorithm to select which sites will actually store each fragment.

The fragmentation and scattering technique, when applied to file storage, involves cutting every sensitive file into several fragments in such a way that one or several fragments (but not all) are insufficient to reconstitute the original file [48]. The number of copies depends on the file criticality defined by the user [44][48]. The user site displays in random order all the fragments of each page to all storage sites. Then, the fragments are stored in several copies on different distributed sites, which can be viewed as fragment server machines [51]. Figure 4 illustrates this processing. The name of each fragment is generated by a hash function from the encryption key, file name, index of page and the index of the fragment. It prevents knowledge of the correct order of fragments of a given page by an intruder [44][48][52]. Figure 5 shows the cycle of file processing during the fragmentation operation. Thus, no information about a fragment can be derived from its name.

At this stage, the question that arises is how to build a map of effective dispersion of minimum computer resources used during the reconstruction of the file. In [53], the article proposed a scattering technique based on a tree structure. The dispersion map proposed the use of a Huffman encoding process based on the use of the frequency of the file. In the same direction, another study proposed two algorithms developed to maintain a constant number of replicated fragments: one based on the game of life, and the other based on roaming ants [54]. Each of them respects the following criteria:

- To maintain an acceptable number of copies of fragments;
- To resist the malicious attacks and multiple node failures;

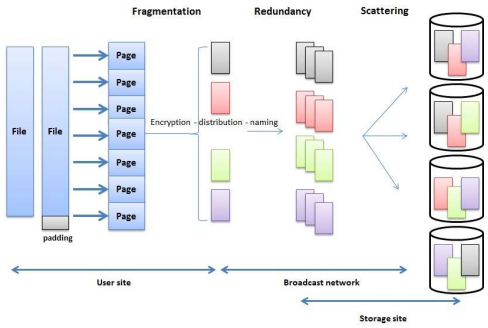

Fig. 4: FRS applied to persistent file storage

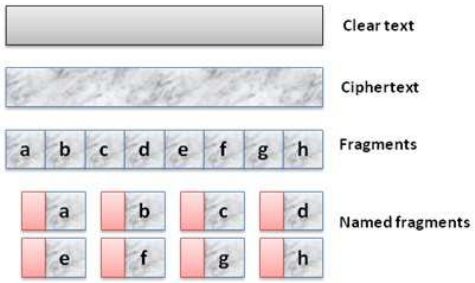

Fig. 5: Transformation cycle of data in the FRS

- To preserve the anonymity of data holders.

This study remarks that the number of replicas fragments generated by the game of life algorithm was higher than the ant swarm algorithm.

During the read operation, the user site reconstructs the names of all the fragments of pages that must be recovered (using a hash function), and diffuses in random read requests to all storage sites that possess a copy of the fragment. If all copies of each fragment are identical, the user site can easily restore encrypted pages, decrypt and verify the checksum. If different copies exist for a fragment, the user site recreates several encrypted pages and tries to decipher until it gets a correct checksum. Only the name of the fragments allows to find their location and this information is calculated during fragmentation operation (based on information as the key to fragmentation, file name, etc.) and dynamically recalculated using the same information at reassembly operation [41][43][44].

In [42], an approach was described, based on FRS in the context of a peer-to-peer architecture where each agent (client and/or server) has the ability to request data storage service from other agents to store elsewhere. The inconvenience here is that nodes (agents) constituting this system have a dynamic behavior (connectivity). The node can have all the fragments associated with a file without problem because the node cannot tell the difference between fragments belonging to a given file.

The effectiveness of FRS appears when the attacker is incompetent to differentiate between fragments of the same file across the transited network flow. This requires that the sending of fragments exchanged between archive sites and user sites should not be sequential in their normal order.

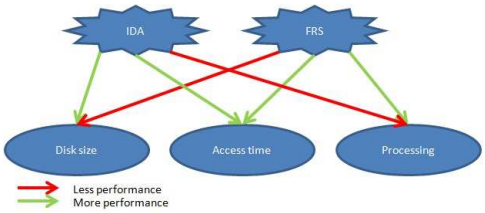

Fig. 6: FRS vs. IDA

The fragments should be transmitted at intervals sufficiently spaced. If necessary, this rate must be increased artificially [45]. The FRS performance depends on the fragmentation granularity. The FRS approach is a method to take advantage of the division file and distributed system to implement reliable applications that handle confidential data.

1) Comparison with Information Dispersal Algorithm technique: The IDA (n;m) algorithm is a tool for converting a file into multiple files and any m files from n files are sufficient to recover the original file. As a performance advantage, this technique provides an adequate level of security. Figure 6 illustrates in brief the comparison between the two techniques IDA and FRS. Based on the conclusions of various scientific research papers, it is argued that the IDA approach does not need the important disk space. With regard to the additional treatment, the FRS processing complexity is less than the IDA one. The fragmentation of a file (including encryption, distribution of bytes in fragments, naming) is faster than encrypting a file with conventional techniques [44]. The IDA technique necessitates high computation on large matrices especially if these calculations are made on traditional workstations. Finally, both techniques have the same access time to file by parallelization of access to different fragments on the storage sites [44][47][48].

C. Applying FRS for data storage and recovery

In cloud computing environment, the user needs some services related to data storage. Before using this service, the user needs some security mechanisms to data access (authentication and authorization). In fact, in this section, several required scenarios related to mono cloud are pesented in order to benefit from storage service.

1) User authentication and authorization: In order to enable security functions to tolerate faults and intrusions, despite the fact that these intrusions are made by security administrators, these functions are implemented as a distributed security system that contains several security sites managed by different administrators. The implementation is based on majority vote and threshold scheme algorithms. Therefore, any k of the n parts are sufficient to reconstruct the original secret [28][55][56][57]. As a majority of these sites are neither defective nor penetrated by an intruder, the security functions will be performed correctly, and no confidential data will be revealed.

The proposed framework is based on the use of two backup sites at least, belonging to a one or more of the clouds. It requires authentication and user authorization from cloud(s) contributing to serve the customer. This registration should be done separately in each cloud or each entity of the cloud. Authentication is perhaps the single, most common requirement of any application. First, the user must authenticate by the CSP to access the services. Then, he must be logged separately on the various security sites, and a user's authenticator is stored on each security site. This authenticator can be a secret shared by the user and site security (password) or biometric information characterizing the user (a fingerprint), or public information corresponding to a secret known only by the user. On the user side, the shared secrets should be stored in a specific device like smart cards or usb flash driver. Our framework requires a seperate registration for each security site, which significantly increases the level of security in our system; since a malicious security administrator cannot, without the help of other administrators, pass for a user, and he cannot create a new user (it would not be recognized by the other security sites). His authority is limited to his own security site and he has no power over other security sites. This approach provides some separation of powers. Figure 7 shows the process followed during authentication in our framework.

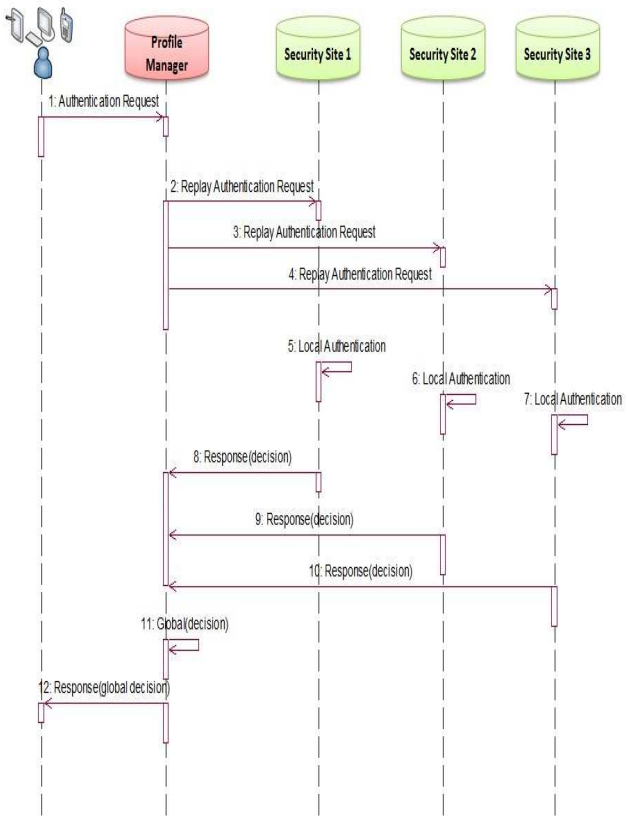

Fig. 7: Authentication scenario in our CCS framework

The user sends a request to an entity named Profile Manager to find the security sites to be able to authenticate (1). Certainly, there are many profile managers to ensure this security mission. Thereafter, the user sends authentication to multiple security sites (2, 3 and 4). Next, each site runs an independent security authentication protocol according to the local authentication scheme (5, 6 and 7) to verify the

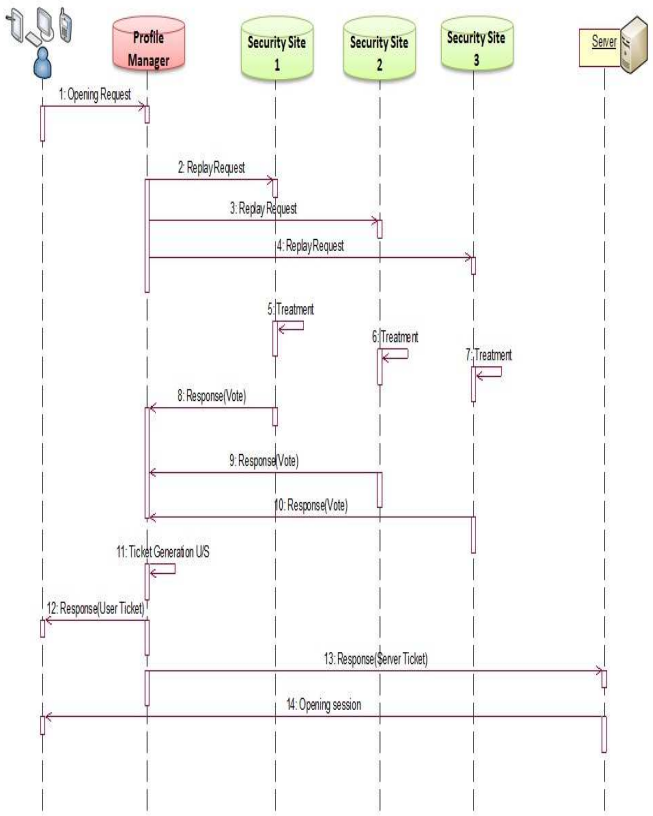

Fig. 8: Authorization scenario in our CCS framework

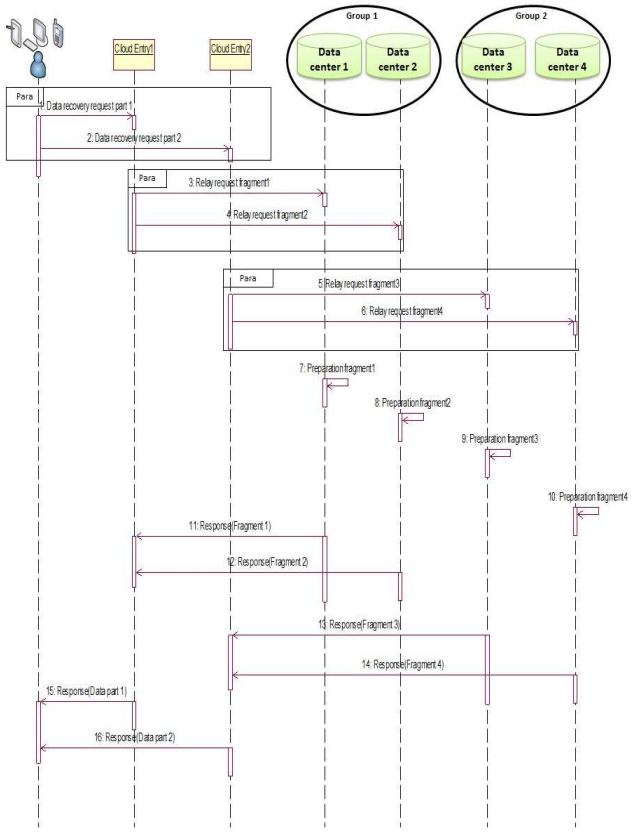

Fig. 9: Backup and data recovery scenario in our CCS framework

identity disclosed (password, biometrics, etc.). Then, these local decisions are shared to obtain an overall decision (8, 9, 10 and 11). Once the global decision is made, a response is sent to the user (12). Finally, the user can send queries to access other servers or objects of cloud computing. Thus, the user obtains session keys, one of them for site security for future accesses to every site security. All other requests will be encrypted by the session key. When there is a need to access an object, the authenticated user sends a request to the security server that allows or denies access. Figure 8 illustrates the protocol for authorization.

2) Storage and data recovery: In this scenario, two ingresses are used for access to cloud computing. The concept of the framework is to create two groups of data centers. Each group is built and assigned to ingress of the cloud (E1 or E2). To do this, during the authorization phase, the middleware (client side) must select two ingresses from the entry proposals of cloud (It is assumed that the cloud contains more than two entry points to its IT infrastructure). This can be done in a random or alternative manner in every time the user wants to establish a connection to a data center; he can choose a different entrance to his previous connection to a data center. Each data center is connected to the client via the two chosen entries (1 or 2) and each of them presents a group in our Framework. Figure 9 illustrates the proposed approach.

Therefore, any possible transaction (backup or data recovery) between the client and the cloud passes through the two communication channels between the client and the two chosen entries of cloud that serves as a bridge to the data center. These channels should be secured by establishing VPN (Virtual Private Network) connections. The figure 10 illustrates the process of data storage in our proposal.

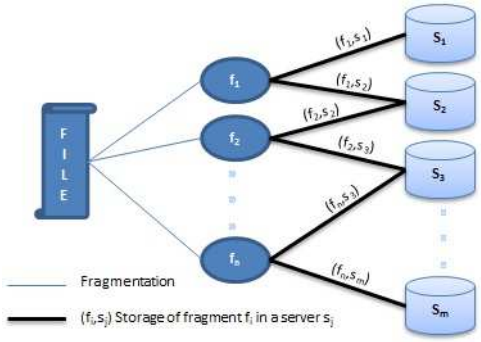

Fig. 10: Logical process of our proposal

Certainly, with many servers that present container of user's data, the confidentiality and availability are guaranteed and then security level rises relatively to actual strategy of clouds.

Let's suppose the probability $p(< 1)$ where an object is affected by an attack (integrity and/or confidentiality and/or availability) in a server. In fact, in the case when an entire file is

hosted in a server and is duplicated in another server, then the probability that this file is affected is $(p.p = p^2)$. However, in our proposal, a file is converted to n fragments that are stored in servers with duplication. Therefore, in this case, the probability that this file is affected is $(\prod_{i=1}^{n} p\ )(\prod_{i=1}^{n} p\ )$=$\prod_{i=1}^{n} p^2$. Consequently, our proposal offers more security than the traditional case because $\prod_{i=1}^{n} p^2 \ < p^2$.

D. The proposal in multi-clouds

This approach can be applied for the multi-clouds. Thus, several clouds can be used depending on the capacity and design of the overall architecture of the clouds. The multi-clouds architecture is the environment where there is cooperation between CSP (see Figure 11). The cooperation between clouds carries more benefits for the customer in terms of security and performance and this is achievable according to the SLA established between the client and the clouds.

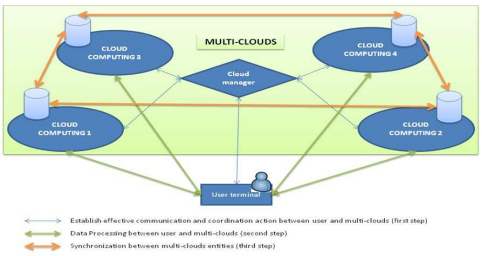

Fig. 11: Multi-clouds architecture in our CCS framework

Our projection of the Framework on the multi-cloud requires the addition of a new entity called cloud Manager (CM) that manages the communications between the client and the clouds. When constructing a communication context with clouds, the user must provide the list of clouds to the CM when he wants to communicate. Then, the scenario of creating a communication context with clouds begins between the user and the clouds selected, using the CM and then, it follows the same procedure as in the case of a single cloud. By the way, many possibilities of CM placement can be proposed:

- CM is implemented in user terminal. In this case, if the CM crashes, the user can restart, re-install or update the CM if there is a need.
- CM is a role that can be assigned in any cloud computing based on SLA (Service Level Agreement). In this case, if the CM crashes in one CC, another cloud computing can be chosen to operate as CM. The choice between cloud computing and CM can be based on priorities, service type oroperation's time, etc.
- CM can be implemented in a company's proxy. In this case, if the CM crashes, the IT department switches to another proxy as backup for the first one.

Independently of previous choices in CM's places, the CM should be available to orchestrate the communications between user and clouds. Certainly, policy manager for access to different clouds is required. The work in [49] treated a framework called Policy Management as a Service (PMaaS) that offers customers the ability to manage access policies to services running on a cloud infrastructure that can be accessed via user interfaces. The framework consists of four main components: cloud user, Policy Management Service Provider (PMSP), CSP and requester.

Other solutions can be integrated to increase the security level discussed in [58]. One consists of establishing a collaborative access control framework called PolyOrBAC. This approach provides each organization belonging to the CII (Critical Information Infrastructure) the ability to collaborate with others while maintaining control on its resources and its internal security policy. A contract is signed between the service provider and the service user to clarify certain degree of interaction and performance. The contract describes the parameters, functions of Web Service (WS), responsibility of each party and the security rules to control interactions. When running, respect for all interactions with these security rules is verified.

Another interesting point to discuss in the multi-cloud is about the impact of this framework on latency. This paper does not cover all QoS problem. However, a brief discuss about latency for this proposal in multi-clouds is necessary. The papers [59][60][61][62] analyzed some of the problems and challenges for achieving real-time cloud computing. QoS management in the cloud computing is linked to the problem of allocating resources to the application to guarantee a service level along dimensions such as performance and reliability [59].

QoS is considered in every side of the network - the user, the backbone network access, and the IP core network. In fact, the QoS depends on both cloud computing and the operator network. The paper [60] investigated if it is possible to use latency as an indicator for the other QoS parameters as throughput and jitter. It concluded that it was not possible to find a consistent relationship between latency and the other parameters. Also, the paper [61] presented PriorityMeister as a system that combines priorities and rate limits to provide tail latency QoS for shared networked storage in CC, even with bursty workloads. The paper [62] presented RT-VMs as a technology allowing virtualized applications to meet QoS constraints as stated in contractual agreements among customers and providers, formalized in proper SLAs.

In our framework, the requests for fragments increase compared to the case where the requests are for one file. However, the overall size of the received user will be the same for the case of one single file. The reconstitution of the original file is faster given the great performance of terminal processing. The problem is the amount of queries to request the fragments. In this case, a compromise is:

- File size,
- Fragments number,
- Number of storage servers,
- Location of servers.

By the way, the impact of the proposal can be decrease by reducing the fragments number and requests number. Normally, when the fragments number increases, the security level

raises. Certainly, this parameter depends on data security degree. Also, the requests number can be reduced by aggregation of fragments' names in few requests in each channel, and this entails a reduction of network traffic.

Finally, before the multi-cloud can offer a service to customer, it should verify everything about the QoS management via monitoring of some parameters that include latency, jitter, packet loss, and bandwidth.

IV. SIMULATIONS AND EVALUATIONS OF OUR PROPOSAL

A. Introduction

In this section, The simulation of the proposed CCS framework is presented for evaluating the performance of the EFRS technique as a dependable and secure solution for cloud storage. To evaluate the robustness, security (availability) and performance of the proposal during data exchange in cloud computing, Netlogo has been chosen as a modeling tool [63][64]. This allowed us to investigate the performance of our framework under various operational conditions.

Basically, during the simulation, some parameters are changed to evaluate the robustness of the framework and make a comparison with a classic model of storage data in the cloud computing. The Classic model is the case where the total file is transferred and saved in one place without cutting in fragments. Likewise, the other copies of this file are transferred and saved in other servers. In the proposal, the classic model is obtained if the fragments number is equal to 1. The figure 12 illustrates the general rule during simulation.

Subsequently, the target is to prove the strength of our framework in difficult conditions (increasing loss percentage of fragments) and have a performance's cartography of all simulations. Also, some conditions are made in automatic routine to evaluate the proposal.

In the simulation, there are:

- Five hundred files, each of them has a size of 150 Mo.
- One hundred servers, each of them has as maximum size of 1 To.
- The replicas number of fragments (or files for Classic model) is equal two.

The AES (Advanced Encryption Standard) is utilized as the symmetric cryptography algorithm before fragmentation. Because the length of plaintext and ciphertext are the same for AES, the advantages of the scheme are evaluated without adding any constraints about encryption operation complexity during the simulation.

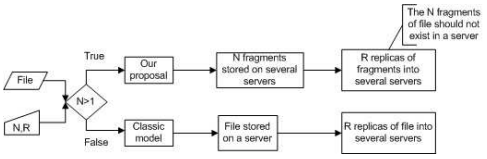

Fig. 12: The general schema of simulation

Some parameters are proposed and that can be changed in the simulation to have a result in output file. The parameters are:

- Number of servers: the number of storage server.
- Server size: the maximum size in the storage server.
- File size: the size file; to simplify, all created files have the same size (150 Mo).
- Fragments number: presents the number of fragments generated by the EFRS technique from the file can be imposed.
- Replicas requisite: the number of replicas fragments that should be normally kept in the cloud computing.
- Maximum buffer: the total size of the treatment data by second.
- Maximum buffer reference: the rest of the buffer during time unit of treatment (here, it is one second).
- FRS (OnOff): switches between our proposal and the classic model.

Here in the simulation, a tick is a time unit, so in this case, one tick is considered as one second.

Firstly, the tests adopted for the evaluation of the Framework are:

- One for classic storage strategy in the cloud computing.
- Second for new storage strategy in our CCS framework. In this case, five simulations are made. In each one, the fragments number is changed: 5, 10, 15, 20 and 25.

Next, the results and synthesis of these tests are presented to valorize the framework. Three options exist in all these simulation studies and that concern the number of servers failed in each trigger event of server failure.

B. Scenarios and results

Under the same experimental conditions, the global target of these scenarios is to show the difference between our proposal and the classic model by measuring some indicators for performance and security of cloud computing under some difficulties. Here, some events are made in automatic routine. Each 60 seconds, a failure server event is simulated. In fact, after each 60 seconds, a number of servers is shut down. The target is to evaluate the behavior of our framework and classic model. Thus, the Mean Time Between Failures (MTBF) for the simulation is 60 seconds.

There are three cases:

- Each 60 seconds, a server is shut down.
- Each 60 seconds, two servers are shut down.
- Each 60 seconds, three servers are shut down.

In each case, six simulations are made:

- Simulation 1: classic model (Sim1).

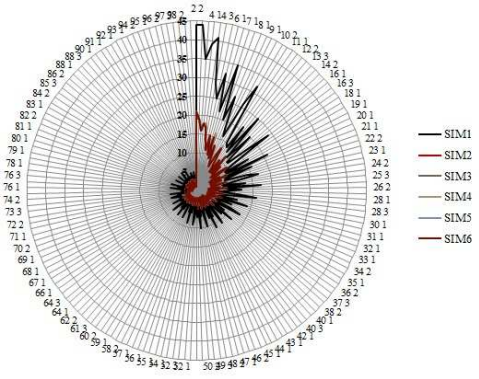

Fig. 13: MTTR comparaison between classic model and our proposal

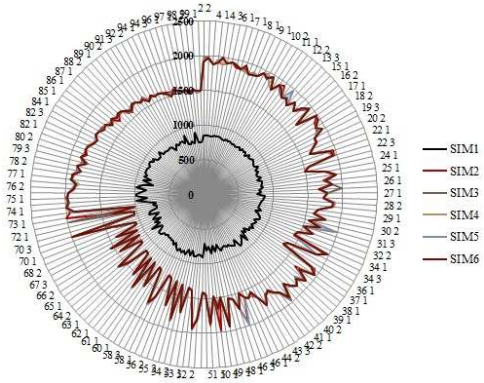

Fig. 14: RE comparaison between classic model and our proposal

- Simulation 2: our proposal with fragments number equal to 5 (Sim2).
- Simulation 3: our proposal with fragments number equal to 10 (Sim3).
- Simulation 4: our proposal with fragments number equal to 15 (Sim4).
- Simulation 5: our proposal with fragments number equal to 20 (Sim5).
- Simulation 6: our proposal with fragments number equal to 25 (Sim6).

Then, eighteen simulations are made in order to describe more the behavior of the model. So the simulations period is between thirty three and one hundred minutes. A radar chart is utilized to compare the aggregation values of multiple data of simulations for three cases.

When the failure event of a server is triggered, immediately the recovery processing of data tries to recover data from other servers that contain a copy of loss data. The target of this processing is to return to initial state before the server failure event. Also, Recovery Efficiency (RE) is defined as Maximum Data Size (MDS), that can be recovered, during Mean Time To Repair (MTTR) of the system. Then:

$$RE = \frac{MDS}{MTTR}$$

Figure 13 illustrates the variation of MTTR for all simulations. Here, the proposal reacts faster than the classic model to stabilize the global situation of the environment. Also, independently of fragments number, the proposal (sim2, sim3, sim4, sim5 and sim6), for three cases, has a stable behavior and approximately the same MTTR during all simulation. Finally, it is remarked that the maximum delay of reaction of the proposal is 20s whereas 45s for the classic model.

Figure 14 illustrates the variation of RE for all simulations. The RE of our proposal is higher than the classic model. Also, independently of fragments number, the proposal has a stable behavior and approximately the same RE during all simulation. As previous metric of MTTR, the RE of the proposal is always more than the classic model. Then, this proposal can be beneficial for the critical applications.

In the proposal, the confidentiality is guaranteed contrary to classic model. In fact, this proposal forbids storing all fragments of a file in one place. Indeed, the storage cloud security is based on encryption. The encryption and decryption keys are saved in memory in some places like USB flash drive. Some attacks, against the computer's memory, provide full access to protected (encrypted) information stored in the most popular types of crypto containers. The encryption keys can be derived from hibernation files or memory dump files. For example, while BitLocker may indeed protect against opportunistic stealing of a computer that is turned off at the time, there are several plausible scenarios for targeted attacks [65][66]. There are many ways available to acquire the original encryption keys. Then, the recovery of keys is easy and then the data are in danger of being lost or falsified for classic model.

In this respect, it is worth nothing that it seems difficult to break our proposal model, many obstacles exist:

- Knowing and access to different servers that contain the fragments.
- Search and collect fragments of file among existent fragments in the cloud; for example find ten fragments of a file among billions of fragments.
- Knowing the order of fragments; if n fragments are need it to generate the original file, so $(n!)$ operations are necessary to find the correct order of fragments.
- Knowing the encryption keys.

In conclusion, the proposal shows superiority in these comparisons. Also, it is remarked that the classic model needs more time than the proposal to stabilize the system situation. The synthesis is:

- Confidentiality in our proposal is higher than the classical model.
- Availability is approximately same in the two models.
- Consummation of memory is generally the same in the both models.
- Recovery efficiency of our proposal is more important than the classic model.

Furthermore, these results also indicate the important role of fragmentation operation, because when there are more fragments, the risk of loss data decreases.

V. Conclusion

In this paper, some issues related to data security have been discussed, also some concepts related to intrusion tolerance have been quoted. A technique IDA and the encryption data in the cloud have been discussed. Furthermore, the proposed CCS Framework based on the FRS technique have been presented in several situations to satisfy most user needs in terms of cloud computing operating, describing some scenarios (Authentication, Authorization, data backup and recovery). Furthermore, the robustness of the proposal have been evaluated by making a comparison with existing classical scheme. According to the results, our CCS framework presents an advantage over the classic model in terms of robustness. This study sheds some light on some advances in terms of data security and performance of cloud environment. The results demonstrate that this architecture can thus optimistically withstand a series of multiple failures.

Currently, other scenarios are being implemented in the real environment to assess the CCS framework in terms of security and performance compared to the current state of CCS. Consequently, as an extension of this work, a middleware will be developped and will be deployed in user terminals to handle all communications easily between the infrastructure provider and the final user. Future works should also examine other potential factors that might influence the global performance of cloud computing like the fragments number and dispersion algorithms. Also, It is planned to develop a dynamic approach in multi-cloud to increase performance especially QoS, based on the CCS framework.

References

[1] Vic (J.R) Winkler. *Securing the cloud - cloud Computer Security Techniques and Tactics*. Elsevier Inc, 2011.

[2] Abdul Nasir Khan, M.L. Mat Kiah, Samee U. Khan, Sajjad A. Madani. *Towards secure mobile cloud computing: A survey*. Future Generation Computer Systems (2012), doi:10.1016/j.future.2012.08.003.

[3] J. Sinduja, S. Prathiba. *Modified Security FrameWork for PIR cloud Computing Environment*. International Journal of Computer Science andMobile Computing-2013.

[4] Clara Leonard. *PRISM : la NSA argumente, le Guardian fait de nouvelles révélations*. From http://www.zdnet.fr/actualites/prism-la-nsaargumente-le-guardian-fait-de-nouvelles-revelations-39791924.htm. ZDNet, Jun 28,.2013. consulted Nov 20, 2013.

[5] Glenn Greenwald, Ewen MacAskill, Laura Poitras. *Edward Snowden: the whistleblower behind the NSA surveillance revelations*. http://www.theguardian.com/world/2013/jun/09/edwardsnowden-nsa-whistleblower-surveillance. The Guardian, jun 10,2013.

[6] Almokhtar Ait El Mrabti, Anas Abou El Kalam, Abdellah Ait Ouahman. *Data Security In The Multi-Cloud*. The International Conference On Networked Systems May 2-4, 2013, Marrakech, Morocco. The First International Workshop on Security Policies in cloud Environment (PoliCE2013)

[7] Keiko Hashizume, David G Rosado, Eduardo Fernndez-Medina, Eduardo B Fernandez. *An analysis of security issues for cloud computing*. Hashizume et al. Journal of Internet Services and Applications. SpringerOpen Journal. 2013, 4:5

[8] A.B. Chougule, G.A. Patil. *Implementation and Analysis of EFRS Technique for Intrusion Tolerance in Distributed Systems*. IJCSI International Journal of Computer Science Issues, Vol. 8, Issue 4, No 1, July 2011.

[9] P. Mell and T. Grance. *The NIST definition of cloud computing*. Special Publication 800-145. Retrieved September 2011, from http://csrc.nist.gov/publications/PubsSPs.html.

[10] Joshna S, Manjula P. *Challenges and Security Issues in cloud Computing*. International Journal of Computer Science and Mobile Computing, Vol.3 Issue.4, April- 2014, pg. 558-563.

[11] Rajkumar Buyya, James Broberg, Andrzej M.Goscinski. *Cloud Computing: Principles and Paradigms*. John Wiley & Sons, 17 dc. 2010.

[12] K. Sudha, M.Tech. MISTE, B. Anusuya, P.Nivedha, A. Kokila. *A Survey on Encrypted Data Retrieval in cloud Computing*. International Journal of Advanced Research in Computer Science and Software Engineering. Volume 5, Issue 1, January 2015.

[13] Almokhtar Ait El Mrabti, Anas Abou El Kalam, Abdellah Ait Ouahman. *Les défis de sécurité dans le cloud Computing - Problèmes et solutions de la sécurité en cloud Computing*. 2012 National Days of Network Security and Systems, IEEE Catalog Number CFP1239S-PRT

[14] Wenjun Luo, Guojing Bai, *Ensuring the data integrity in cloud data storage*. International Conference on cloud Computing and Intelligence Systems (CCIS), IEEE,240 243,15-17, 2011.

[15] Prashant Srivastava, Satyam Singh, Ashwin Alfred Pinto, Shvetank Verma, Vijay K. Chaurasiya, Rahul Gupta. *An architecture based on proactive model for security in cloud computing*. IEEE 2011.

[16] Christian Baun, Marcel Kunze, Jens Nimis, Stefan Tai. *Cloud Computing Web-Based Dynamic IT Services*. Springer, 2011.

[17] Thomas Haeberlen, Lionel Dupr. *Cloud Computing Benefits, risks and recommendations for information security*. The European Network and Information Security Agency (ENISA). Rev.B December 2012. from https://www.enisa.europa.eu.

[18] Cloud Security Alliance. *The Notorious Nine: cloud Computing Threats in 2013*. February 2013. from http://www.cloudsecurityalliance.org/topthreats/

[19] B. Rex Cyril, DR. S. Britto Ramesh Kumar. *Cloud Computing Data Security Issues, Challenges, Architecture and Methods- A Survey*. International Research Journal of Engineering and Technology (IRJET). Volume 02 Issue 04. July-2015.

[20] Mingqiang Li. *On the Confidentiality of Information Dispersal Algorithms and Their Erasure Codes*. the IBM China Research Laborato. 2013.

[21] Rabin, M.O. *Efficient dispersal of information for security, load balancing, and fault tolerance*. In: Journal of The ACM 36(2), pp. 335348 (1989)

[22] Abdelsalam A. Helal, Abdelsalam A. Heddaya, Bharat B. Bhargava. *Replication Techniques in Distributed Systems*. The Kluwer International Series on ADVANCES IN DATABASE SYSTEMS; KLUWER ACADEMIC PUBLISHERS 1996.

[23] M. TOUT Rabih. *Sauvegarde des données dans les réseaux P2P*. Thèse de l'université de Lyon - 2010.

[24] Sian-Jheng Lin and Wei-Ho Chung. *An Efficient (n; k) Information Dispersal Algorithm for High Code Rate System over Fermat Fields*. IEEE COMMUNICATIONS LETTERS, VOL. 16, NO. 12, DECEMBER 2012.

[25] Dongfang Zhao, Kent Burlingame, Corentin Debains, Pedro Alvarez-Tabio and Ioan Raicu. *Towards High-Performance and Cost-Effective Distributed Storage Systems with Information Dispersal Algorithms*. 2013 IEEE.

[26] Honglu Liu, Shugang Zhang, Xiaolan Guan. *Integration study of high quality teaching resources in universities*. Journal of Industrial Engineering and Management-2013.

[27] A. Shamir. *How to share a secret*. Communications of ACM, vol. 22, no. 11, pp. 612613 (1979).

[28] Martin Tompa. *How to Share a Secret with Cheaters*. 1998, Springer-Verlag.

[29] Hugo Krawczyk. *Secret Sharing Made Short*. 1998, Springer-Verlag.

[30] Chetan Bansal, Karthikeyan Bhargavan, Antoine Delignat-Lavaud, Sergio Maffeis. *Keys to the Cloud: Formal Analysis and Concrete Attacks on Encrypted Web Storage*. Second International Conference, POST 2013, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2013, Rome, Italy, March 16-24, 2013. Proceedings. Springer Berlin Heidelberg.

[31] *Guidance Implementing the Cloud Security Principles*. From https://www.gov.uk/government/publications/implementing-the-cloud-security-principles/implementing-the-cloud-security-principles. Consulted May 10, 2016.

[32] Claire Broadley. *Copy.com Cloud Storage: the Solution to Fair Sharing?*. From http://www.cloudwards.net/copy-com-cloud-storage-the-solution-to-fair-sharing/. Cloudwards.net (04 Feb 2016) consulted Jun 1, 2016.

[33] *Encryption key management is vital to securing enterprise data storage*. From http://www.computerweekly.com/feature/Encryption-key-management-is-vital-to-securing-enterprise-data-storage. Computer Weekly consulted Jun 1, 2016.

[34] Margaret Rouse. *Cloud encryption (cloud storage encryption) definition*. From http://searchcloudstorage.techtarget.com/definition/cloudstorage-encryption. consulted Dec 12,2015.

[35] Stephen Lawton. *Cloud Encryption: Using Data Encryption In The Cloud*. From http://www.tomsitpro.com/articles/cloud-data-encryption,2-913.html. Tom's IT Pro (APRIL 30, 2015) consulted Jun 01, 2016.

[36] *Security Guidance for Critical Areas of Focus in Cloud Computing V3.0*. From https://cloudsecurityalliance.org/download/security-guidance-for-critical-areas-of-focus-in-cloud-computing-v3/. Nov 14,2011

[37] Marten van Dijk, Craig Gentry, Shai Halevi, Vinod Vaikuntanathan. *Fully Homomorphic Encryption over the Integers*. International Association for Cryptologic Research. H. Gilbert (Ed.): EUROCRYPT 2010, LNCS 6110, pp. 2443, 2010.

[38] Craig Gentry, Dan Boneh. *A fully homomorphic encryption scheme*. Stanford University, Stanford, CA, 2009.

[39] Iain Thomson. *Microsoft researchers smash homomorphic encryption speed barrier*. From http://www.theregister.co.uk/2016/02/09/researchers_break_homomorphic_encryption. The Register (Feb 9,2016) consulted JUN 1, 2016.

[40] Nathan Dowlin, Ran Gilad-Bachrach, Kim Laine, Kristin Lauter, Michael Naehrig, and John Wernsing. *CryptoNets: Applying Neural Networks to Encrypted Data with High Throughput and Accuracy*. Microsoft Research (February 24, 2016).

[41] J-C Fabre, Y. Deswarte and B. Randell. *A Framework for the Design of Secure and Reliable Applications by Fragmentation-Redundancy-Scattering*. Technical Report Report Series No. 410 February 1993.

[42] Rudi Ball, Vicki Spurrett and Rogério de Lemos. *Dependable and Secure Storage in Pervasive Peerto-Peer Systems*. Technical Report September 2006. University of Kent, Canterbury, Kent CT2 7NF, UK.

[43] Jean-Charles Fabre, Yves Deswarte and Brian Randell. *Designing Secure and Reliable Applications using Fragmentation-Redundancy-Scattering: an Object-Oriented Approach*. EDCC-1 Proceedings of the First European Dependable Computing Conference on Dependable Computing. Pages 21-38. Springer-Verlag London, UK 1994

[44] Jean-Charles Fabre, Yves Deswarte, Laurent Blain. *Tolérance aux fautes et sécurité par fragmentationredondance-dissémination*. Rapport de recherche - LAAS-CNRS& INRIA.

[45] Jean-Michel Fray, Yves Deswarte, David Powell. *Intrusion-tolerance using fine-grain fragmentation scattering*. 1986 IEEE.

[46] Rabih Zbib, Farooq Anjum, Abhrajit Ghosh, Amjad Umar. *Intrusion Tolerance in Distributed Middleware*. Information Systems Frontiers 6:1, 6775, 2004 Kluwer Academic Publishers. Manufactured in The Netherlands.

[47] J.-C. Fabre and T. Pérennou. *Processing of confidential information in distributed systems by fragmentation*. Computer Communications, vol. 20, pp.177 -188 1997.

[48] Yves Deswarte, Laurent Blain and Jean-Charles Fabre. *Intrusion Tolerance in Distributed Computing Systems*. 1991 IEEE.

[49] Hassan Takabi and James B. D. Joshi. *Policy Management as a Service: An Approach to Manage Policy Heterogeneity in cloud Computing Environment*. 2012 45th Hawaii International Conference on System Sciences. 2012 IEEE.

[50] Hassan Takabi, James B.D.Joshi and Gail-Joon Ahn. *Security and Privacy Challenges in cloud Computing Environments*. The IEEE Computer and Reliability Societies 2010.

[51] Yves Deswarte. *Fragmentation-Redundancy-Scattering: a means to tolerate accidental faults and intrusions in distributed systems*. Proceedings of the ERCIM Workshops, INESC, Lisbonne (Portugal), 14-15 novembre 1991, pp. 31-34.

[52] Bruno Martin. *Codage, cryptologie et applications*. Collection technique et scientifique des télécommunications 2004.

[53] Farooq Anjum and Amjad Umur. *Agent Based Intrusion Tolerance using Fragmentation-Redundancy-Scattering Technique*. 2000 IEEE.

[54] Rudi Ball, James Grant, Jonathan So, Victoria Spurrett, Rogério de Lemos. *Dependable and secure distributed storage system for ad hoc networks*. Springer-Verlag Berlin Heidelberg 2007. LNCS 4686, pp. 142-152.

[55] Jun Kurihara, Shinsaku Kiyomoto, Kazuhide Fukushima, Toshiaki Tanaka. *A New (k,n)-Threshold Secret Sharing Scheme and Its Extension*. Proceeding ISC 2008 Proceedings of the 11th international conference on Information Security, Taipei, Taiwan, September 15-18, 2008 Springer.

[56] A. Shamir. *How to share a secret*. Communications of ACM, vol. 22, no. 11, pp. 612613 (1979)

[57] Hugo Krawczyk. *Secret Sharing Made Short*. 1998, Springer-Verlag

[58] A.Abou El Kalam, Y.Deswart, A.Bana, M.Kaniche. *PolyOrBAC:A security framework for Critical Infrastructures*. International Journal Of Critical Infrastructure Protection 2(2009)154-69.

[59] Danilo Ardagna, Giuliano Casale, Michele Ciavotta, Juan F Prez and Weikun Wang. *Quality-of-service in cloud computing: modeling techniques and their applications*. Springer 2014.

[60] Jens Myrup Pedersen, M. Tahir Riaz, Bozydar Dubalski, Damian Ledzinski, Joaquim Celestino Jnior and Ahmed Patel. *Using latency as a QoS indicator for global cloud computing services*. John Wiley& Son 2013.

[61] Timothy Zhu, Alexey Tumanov, Michael A. Kozuch. *PriorityMeister: Tail Latency QoS for Shared Networked Storage*. ACM 2014.

[62] Marisol Garcia-Valls, Tommaso Cucinotta, Chenyang Lu. *Challenges in real-time virtualization and predictable cloud computing*. Elsevier 2014.

[63] S. Tisue and U. Wilensky,*Netlogo: A simple Environment for Modeling Complexity. International Conference on Complex System*. Boston, May 2004.

[64] Wilensky, U. (1999). *NetLogo*. From http://ccl.northwestern.edu/netlogo/. Center for Connected Learning and Computer-Based Modeling, Northwestern University, Evanston.

[65] *Researcher Demonstrates Simple BitLocker Bypass*. By SecurityWeek News on November 18, 2015. from http://www.securityweek.com/researcherdemonstrates-simple-bitlocker-bypass. consulted MAR 10,2016.

[66] Sven Trpe, Andreas Poller, Jan Steffan, Jan-Peter Stotz and Jan Trukenmller. AAA. *Attacking the BitLocker Boot Process*. Second International Conference, Trust 2009, Oxford, UK, April 6-8, 2009, Proceedings. Pages 183 - 196. Springer Berlin Heidelberg.