Decentralized access control mechanism with temporal dimension based on blockchain

Mayssa JEMEL[1] and Ahmed SERHROUCHNI[2]

[1] Sorbonne University, UPMC Paris 6, CNRS, LIP6 UMR 7606, Paris, France.
[2]Paris-Saclay University, TELECOM ParisTech, CNRS, LTCI 75013, Paris, France.
Email: mayssa.jemel@lip6.fr, ahmed.serhrouchni@telecom-paristech.fr

***Abstract*—Used mainly for the virtual money, the Blockchain technology adopts a decentralized network of peers to ensure a secure and transparent information storage and transmission. The basic use of Blockchain can be bypassed, and it is interesting to integrate it into other fields such as the Cloud storage. Cloud storage solutions ensure a continuous data synchronization and guarantee data sharing between different users. However, sharing user side encrypted data raises key and access control management challenges. Within this paper, we propose a novel access control model called Timely CP-ABE. Two main features come with our model. First, we introduce a decentralized access control mechanism where the user legitimacy is verified by Blockchain nodes. Second, we add temporal dimension to file sharing based on CP-ABE. In fact, we introduce a validity time to the access authorization without additional revocation cost. As a proof of concept, the implementation of the Timely CP-ABE based on Blockchain is performed on the CP-ABE toolkit and Multichain solution.**

***Index Terms*—File sharing, Access control, Key management, Blockchain, CP-ABE, Temporal dimension.**

I. Introduction

With the technological progress, multiple connected and smart devices are exploited and frequently used. Therefore, the cross-devices data management remains the concern of multiple research and industrial projets. In fact, various devices, owned by the same user or different ones, rely on synchronization protocols in order to maintain data consistency. In this context, several Cloud storage solutions are proposed to handle this issue. They offer the possibility to externalize data. Thus, users can upload, download, share and access their data whenever and wherever they want.

Remote synchronization has been the subject of significant works [1] [2] that depend on the purpose of the software and the nature of objects to synchronize. In fact, these objects can be stored in structured or unstructured databases, in files and folders. In this context, Gartner has published a report entitled "EFSS: Enterprise File

One of the main challenges of data sharing is the access control management, and it is obvious that additional one should be addressed when data are client-side encrypted. It is the encryption and decryption key management.

Access control management is usually based on centralized architecture where a single entity verifies users legitimacy. Nowadays, current solutions are migrating towards decentralized solutions to overcome the weakness of the centralized ones. Dealing with recent solutions, we find the Blockchain. The fundamental value of this technology is to adopt a decentralized network of peers to ensure a secure and transparent information storage and transmission through transactions. These informations are directly shared across multiple nodes without requiring a central administrator. In fact, the decentralization is ensured as each transaction holds its own validity proof and its own authorization proof. Blockchain was created first for the virtual money, more precisely for the bitcoin money [3] and it is interesting to integrate it into other fields such as the access control management for data sharing.

Three main issues must be point out when dealing with access control management of sahred data. First, neither the user who shares data nor the consumer who wants to access the owner data are usually connected at the same time. Second, the access control should cover users as well as third parties not authorized to access the data. Third, we should avoid a single entity that manages access control for security issues.

To emphasize these considerations, two novelties come with our proposal. First, we cover the two first points with the adoption of a fine-grained access control with a temporal dimension by adopting all necessary security measures. The idea behind is to introduce the notion of validity time to the authorization access without additional cost of revocation. Second, to deal with the third point, we bypass the basic use of Blockchain and adopt it for a decentralized aceess control management.

The rest of the paper is structured as follows. In section 2, we give an overview on existent work related to access control and blockchain for data sharing. In section 3, we present the access control with the integration of the temporal dimension and the blockchain. We deal with the implementation and the analysis of our solution in section 4. We end up with a conclusion and perspectives.

II. Related work

A. Access control for data sharing

In the context of data synchronization, we need to focus on the data sharing while highlighting the privacy. The data privacy is susceptible to be compromised as the storage and the protection are usually delegated to remote storage servers. The basic access control methods consider that the entities that manage the access control as fully trusted which is not

978-1-5386-1412-9/17 $31.00 © 2017 IEEE
DOI 10.1109/ICEBE.2017.35

the case in the Cloud storage services. For this purpose, advanced encryption methods are adopted to ensure the access control. It preserves the confidentiality in addition to the key management.

Several cryptographic access control mechanisms were proposed to ensure both file sharing and privacy when stored in untrusted servers. Their approach is based on externalizing files encrypted by the owner. Thus, the decryption keys are retrieved only by authorized users. The retrieval of the decryption key can follow one of these three strategies:

Generated from user profile:

The encryption and decryption keys are designed for a specific or a group of users. Identity-based encryption (IBE) [4] is one of the first solutions which extends the public-key paradigm. The basic idea is to generate the public key based on the consumer identity to encrypt the data. A private key is then generated by a third trusted party and sent to the consumer to decrypt these data.

Different from the basic public key cryptography used by IBE, the Attribute-Based Encryption (ABE) is used when the encrypted messages are not intended for a specific destination. In fact, one to many encryptions is the target. Two kinds of ABE exist: Ciphertext Policy Attribute Based Encryption (CP-ABE) [5] and Policy Key Attribute Based Encryption [6] (PK-ABE). In CP-ABE, the access policy is integrated into the ciphertext, while it is linked to the decryption key in PK-ABE.

Transfered directly to the user:

Under this case, the owner acts as a key distribution entity. He transfers directly the decryption key to the consumer who is authorized to share data. Among the realized works, we find, Plutus [7] which divides the files into filegroups (group of files having similar sharing attributes). Each file is encrypted with a unique symmetric key then encrypted with the symmetric key of the filegroup to which it refers. The filegroup key is further delivered by the owner to authorized consumers. In case of revocation, all the keys should be updated, and whole files should be re-encrypted with the new keys. This strategy is not appropriate for fine-grained access control with a huge number of filegroups. In addition, the owner should be usually connected to send the decryption key.

Transfered from third parties:

Some solutions store the file as a couple of file metadata and file content. With Sirius [8], besides information related to the file, the metadata includes the access control policy and a series of data decryption keys encrypted with each authorized consumer public key. Thus, the size of the metadata is proportional to the number of authorized users.

A proxy can also be used such as the case in [9]. It is based on the proxy re-encryption to secure the file storage. Without retrieving the plaintext, a semi-trusted proxy converts a ciphertext encrypted under the owner public key into another ciphertext. The last ciphertext can be decrypted only by the consumer private key. Thus, the private and public keys are generated using a set of proxy re-encryption operations.

B. Blockchain and access control

Blockchain technology with the P2P networks was used first for virtual money exchange. The idea behind is to avoid relying on third parties to prove the identity of user and grant access to the system. In fact, it represents a database of the transactions history performed between users since the creation of the chain. This database is secured and shared between the different nodes of the blochchain where every node participates in the validation process of transactions. The name chosen for Blockchain comes from the mechanism adopted in the technology itself. In fact, the transactions are grouped into blocks, and each block is validated by supernodes called miners.

As a novel trend, many research and industrial works focus on Blockchain and adopt in areas other than Bitcoin and virtual money. For example, M.baldi and al in [10] use blockchain for certificate validation. Based on Blockchain, they propose a new PKI solution which is used to address the issues of reliability and security of certificate revocation information. In the field of access control, Blockchain is adopted in [11] to protect personal data. In this paper, the blockchain accepts two types of transaction the first one is used for access control management and the second for data storage and retrieval. However, they consider that the resource requester and owner should be both online, which is not usually the case in the data sharing context. Still in the same field, a framework called FairAccess is proposed [12]. This framework is used to ensure a decentralized access control model based on blocks following the IoT constraint. Thus, the blockchain is considered as a database that stores all access control policies for each pair (resource, requester) in the form of transactions. It serves also as logging databases that ensure auditing functions. However, the FairAccess is restricted to the case where the requester and ressource owner must be connected at the same time.

III. Decentralized access control with temporal dimension

In this section, we deal first with the CP-ABE with the temporal dimension. Second, we detail the integration of blockchain into the Timely CP-ABE.

A. Timely Ciphertext Policy Attribut Based Encryption

Among the known ABE methods, we choose to work with the CP-ABE. In fact, each user has a set of roles and properties. According to these properties, a user gets permission to access the files. These properties can be translated into a set of attributes. In addition, as we focus on a timely file sharing, the time of an access request can be merged with the user attributes to generate the encryption key.

In the centralized architecture of file synchronization and sharing, we find an entity responsible of change synchronization, access control management and conflict resolution. It is called Sychronization Manager [2]. In our decentralized access control model, this entity must be an integral part of the Blockchain network.

1) Timely CP-ABE: System Level operation:

The high level operation in our architecture can be designed as follows:

- System setup: This step focuses on the system initialization. In fact, the Synchronization Manager executes the CPABE_setup to generate the Public key and the Master Secret Key, and sends them to the Blockchain nodes.

- New File generation:

This procedure is ensured by the data owner. In fact, this entity:

- generates two keys K_1 and K_2;
- computes K = $K_1 \otimes K_2$ used to encrypt the file;
- encrypts K_1 using the public key of the consumer certificate and sends it securely to the consumer;
- encrypts K_2 using the Timely CP-ABE ($\{K_2\}_{CP-ABE}$) with the appropriate policy to be sent to the Synchronization Manager and the blockchain nodes;
- encrypts the file using the Key K and sends it to storage servers.

- Consumer access verification:

In our proposal, we consider the distinction between the first and continuous synchronization request. In fact, the consumer can be offline when the owner makes changes to his data or to the access policies. In addition, the owner will not necessarily be online when the consumer receives notification for synchronizing. However, we require that both parties are online for the first request of data sharing.

The different steps of consumer legitimacy verification are as follow:

- When a legitimate consumer asks for a file sharing for the first time, it receives, from the data owner, the key K_1 encrypted with his public key $\{K_1\}_{Kpub_{consumer}}$.
- The transaction of GetAccess is sent to different Blockchain nodes. The time of access is retrieved and sent too within this transaction.
- When received, the different nodes verify the legitimacy of the consumer by decrypting $\{K_2\}CP-ABE$ based on consumer attributes, current time and Timely CP-ABE decrypt operation. The transaction is validated when the nodes retrieve the key K_2.
- For a continuous synchronization request, only K_2 is sent to the consumer. In fact, the Synchronization Manager re-checks the access privilege of the consumer, and regenerates the new key K_2.

The legitimate consumer is the unique entity that can retrieve the data. Even if all Blockchain network nodes have the Key K_2, it lacks the key K_1 to compute K. In the other side, even if the consumer has the key K_1, without the access privilege (i.e. time expiration or faults attributes) he cannot retrieve the key K_2 from the Synchronization Manager to compute K and to get the authorization to access the data.

- Owner revocation:

When the owner needs to change the access policy, the key K_2 is encrypted according to the new policy. Even if the owner goes offline after changing the access policy, the Blockchain nodes still handle the access verification. When the consumer has no longer access to the owner data, he is not authorized to retrieve the K_2 and to retrieve the latest version of the file. The owner revocation in our case is performed without additional time.

- Data deletion:

When an owner wants to delete the data sharing, he changes the policy and sends the SetPolicy transaction to the Blockchain network. It is sent also to the Synchronization Manager to update the version of the Cloud.

2) Timely CP-ABE: Algorithm Level operation:

The different operations which need to be defined by our Timely CP-ABE are:

- Setup:

It generates the Public Key (PK) and the master secret key (MSK). This function is performed by the Synchronization Manager.

- Encryption(PK, K_2, Policy):

Used by the owner, the function encrypts K_2 using the Public Key (PK) and according to the chosen access Policy. It generates $\{K_2\}_{CP-ABE}$. This policy is defined by the access in the guise of a tree structure where the leafs are the different attributes, and the interior nodes are the threshold gates. To guarantee that the file can be accessed only for a specific period of time (equation 1), the owner introduces, besides the attributes that define his access policy (Policy'), the time of start and end of the access period interval.

$$Policy = Policy' \wedge (TStart \wedge TEnd) \tag{1}$$

In this case, when the access time is overtaken, the consumer is no more concerned by the synchronization.

- Key_generation(PK, Consumer_public_key, MSK, A):

Used by the Blockchain nodes, this function generates the Consumer_public_key. This key is customized to the consumer according to the set of his attributes. In order to ensure a timely file sharing, the access depends heavily on the period predefined by the owner in his access policy.

Consequently, the request timestamping should be performed at this level, and this time should be added to the user attributes when generating the decryption key (equation2).

$$A = UserAttributes \cup Time\_Access\_Request \tag{2}$$

- Decryption (PK, Consumer_Private_Key, $\{K_2\}_{CP-ABE}$):

This function verifies the consumer legitimacy. In fact, it verifies first that the consumer has the attributes that meet the owner access policy. Second, it verifies that this request is performed during the period chosen by the owner. When these conditions are satisfied, this function can decrypt $\{K_2\}CP-ABE$ using the Consumer_Private_Key. Otherwise, an access denied alert is raised. This function is performed by the Blockchain network nodes.

B. Decentralized access control management based on Blockchain

Blockchain was created first for the Bitcoins. In our architectural, we adopt it mainly for a decentralized access control management within the context of file sharing.

To adapt the blockchain to our context, we consider that:

- The blockchain nodes are users who are subscribed to our data storage service. Each node connected to the network uses an application called wallet.
- The Synchronization Manager is a part of the chain as it is the front end of the synchronization infrastructure, and as it is the entity that controls the access to stored data.
- Two main transactions of the Blockchain are considered: SetPolicy and GetAccess

In the Blockchain, a transaction is a sign piece of data which identifies the action performed from one account to another. A transaction should have a unique ID to be identified and a set of inputs used by the transaction to produce a set of output as results.

One of the main constraints that we consider in data sharing is that neither the owner nor the consumer are connected at the same time. Therefore, both transactions should be generated separately.

1) SetPolicy transaction: The different steps related to the SetPolicy transaction are shown in the figure 1.

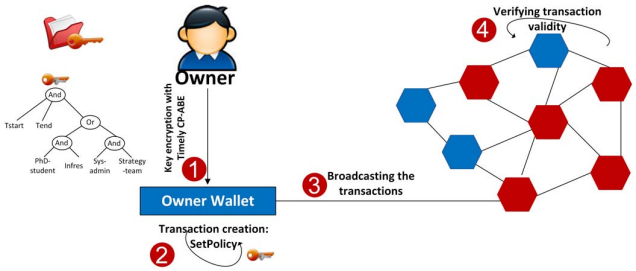

Fig. 1. SetPolicy transaction

- (1) When the owner decides to store an object in the Cloud, he defines an appropriate access control policy. This policy follows a tree structure where the leafs are the different attributes, and the interior nodes are the threshold gates. The period of access is also added into the tree as additional attributes as detailed in the previous part. The objects are encrypted before their sent to the Cloud. Hence, the Timely CP-ABE will be adopted to encrypt a part of the key used in object encryption.
- (2) The Owner Wallet creates the SetPolicy transaction. This transaction follows this form:

$$\begin{aligned} &Tx : (tr, sign(tr)) \\ tr = (&IDx, \\ &input(ObjectID), \\ &output(Owner, Encryptedkey) \end{aligned} \tag{3}$$

In the transaction input, we find the ID of the concerned file. In addition, this transaction encapsulates in its output the encrypted key $\{K_2\}_{CP-ABE}$ (i.e. the key $\{K_2\}$ is used to encrypt the file, and this key is encrypted using the Timely CP-ABE).

- (3) The Owner Wallet, as a part of the blockchain network, broadcasts the transaction between the different nodes.
- (4) In the blockchain network, the transaction is validated and stored in blocks. In this case, the nodes witnesses that the owner imposes a certain policy to this object and that only legitimate consumers can access its content.

2) Getaccess transaction:

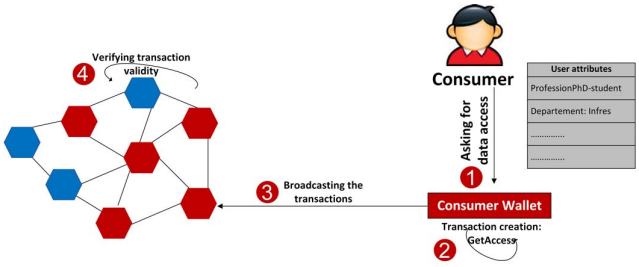

Fig. 2. GetAccess transaction

The different steps of this transaction are illustrated in the figure 2.

- (1) The second phase starts when the consumer asks for accessing a particular owner file;
- (2) The Consumer Wallet creates a transaction called GetAccess shown bellow:

$$\begin{aligned} &Tx : (tr, sign(tr)) \\ tr = (&IDx, \\ &input(ObjectID, Attributes, Current\_time), \\ &output(Consumer) \end{aligned} \tag{4}$$

The basic inputs of this transaction are the ObjectID, the different attributes of the consumer and the current time (i.e. the time of legitimacy verification to access the owner data);

- (3) The GetAccess transaction is disseminated throughout the Blockchain network instantaneously;
- (4) Based on the attributes and the time of an access request, Clockchain nodes verify the transaction. Based on the Timely CP-ABE, if the encrypted key was decrypted, the consumer is legitimated to access owner data otherwise the access request is rejected.

IV. Proof of concept and solution analysis

A. Implementation of the decentralized Timely CP-ABE

To prove the concept of our proposed access control model for data sharing, we use both the CP-ABE toolkit [13] for

access control management and the Multichain solution [14] for the Blockchain implementation. CP-ABE toolkit provides a set of cryptographic operations used to implement the Ciphertext-Policy Attribute-Based Encryption scheme. Regarding Mulichain, it is an open source platform used to create a private blockchain within different nodes. This solution is mainly designed to handle transactions related to virtual money. Hence, it handles quantitative transaction, which is not our case with the SetPolicy and GetAcess transactions. To use Multichain in our context, we use the metadata field sent with each created transaction. In this field, we introduce as a string the inputs of the SetPolicy and GetAccess transactions. Thus, for each received transaction, the metadata field is retrieved and analysed by nodes.

We detail, thereafter, the Timely CP-ABE and Blockchain instructions that should be followed by our architecture entities.

- We consider, in our example, that the consumer is a PhDStudent and a part of the NPA team:

(0) Consumer: *attributes(PhDStudent, NPA)*

- The owner generates the Keys K_1 and K_{objID} and uses them to compute the Key K = $K_1 \otimes K_{objID}$

(1) Owner: *Generate(K_1)*

(2) Owner: *Generate(K_{objID})*

(3) Owner: *Compute(K, K_2, K_{objID})*

- The owner encrypts the key K_{objID} using the Timely CP-ABE. In this instruction, he specifies, besides the policy, the validity time (between 1497526129 and 1497525120 in ms),

(4) Owner: *cpabe-enc pub_key K_{objID} current_date $<$ 1497526129 and current_date $>$ 1497525120 and ((PhDStudent and NPA) or (sys_admin and 2 of (strategy_team, phare, DSI)))*

- The owner generates a SetPolicy transaction, and its input is the file name where the encrypted K_{objID} is stored

(5) Owner: multichain-cli chain sendwithdata @wallet 'SetPolicy,0' K_{objID}.cpabe

- When the consumer needs to access to owner data, his Wallet generates a GetPolicy. The outputs of this transaction are consumer attributes (i.e. PhDStudent, NPA), current time and the file name where the encrypted K_{objID} is stored. In our case, we do not need to introduce the Object ID as input since this information is part of the file name.

(6) Consumer multichain-cli chain sendwithdata @wallet 'GetPolicy' "Attributes, '$date+\%s$', K_{objID}.cpabe"

- The GetPolicy is broadcasted to the network. The nodes generate the decryption keys consumer_public_key. Then, they verify and validate the transaction when decrypting successfully the K_{objID}.cpabe file

(7) Blockchain nodes *cpabe-keygen -o consumer_public_key pub_key master_key PhDSturdent NPA 'current_date = '$date+\%s$'*

(8) Blockchain nodes: *cpabe-dec pub_key consumer_priv_key K_{objID}.cpabe*

- The synchronization manager sends the retrieved K_{objID} to the consumer. The consumer when receiving this key, computes the key which will be used for file decryption

(9) Consumer: Compute(K, K_2, K_{objID})

To evaluate the performances of our proposal, we compare the introduction of the temporal dimension to the Ciphertext Policy Attribue Based Encrypion and its integration to the simple access control list while adopting the Blockchain.

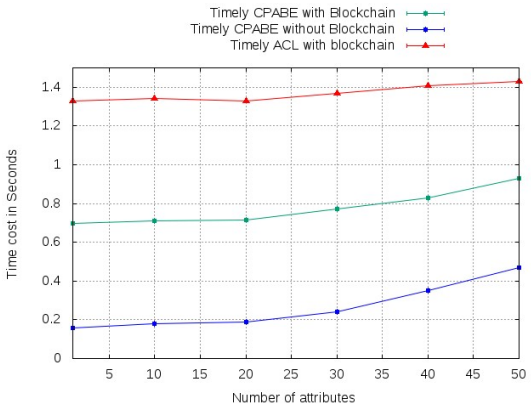

Fig. 3. Performances of the Timely CP-ABE with blockchain

In the Figure 3, the axis of X and Y match respectively to the total number of attributes used in the access policy and the time cost of access verification in seconds. We analyze three different curves: the Timely CP-ABE using a decentralized access control management based on Blockchain, the Timely CP-ABE based on a centralized access control verification and the Access control list with the temporal dimension using Blockchain. The results show that adopting the decentralized architecture comes with an additional cost. It is expected especially with the additional time brought by the exchanged Blockchain transactions. Interestingly, the Timely CP-ABE is more efficient than the Timely Access Control List. More precisely, the Timely CP-ABE reduces the verification cost by the fifth compared with ACL. It is even more efficient as it handles the key management besides the access verification which is not done by the ACL.

B. Security and privacy analysis

In our proposed access control model, two novelties are covered: the access control and key management with a temporal dimension based on the Timely CP-ABE and the decentralized access control management based on Blockchain. The integration of these two technologies offers even greater benefits for the security of our data sharing service. These benefits are detailed as follows:

- **Avoiding a single point of failure**
  One of the main challenging and interesting contributions that come with the blockchain in our data sharing architecture is the avoidance of a single point of failure. In fact, the system does not require a unique trusted third part as each node of the network saves the different transaction history. In the basic Bitcoin architecture, the miners are rewarded with coins when they succeed to validate the blocks of transactions. In our case, as we do not deal with virtual money, users who choose to be miners are rewarded with additional free storage space.
- **Non-repudiation**
  The blockchain guarantees that no one of the owner and consumer denies the contract of data storage, access and share. In fact, the different transactions of policy setting and data retrieval are stored every where in the network.
- **Auditing**
  The histories of transaction are stored in blocks and conserved in each node of the network. Therefore, the Blockchain guarantees auditing and thus following the different actions performed by the users.
- **Confidentiality against the Synchronization Manager and Blockchain nodes**:
  While the synchronization Manager is supposed to harm the confidentiality of data, it must be honest to follow the data and key management instructions and to process the request of data access. Even if the Synchronization Manager and the other Blockchain nodes handle the key managements and verifies the access legitimacy, they hold only a part of the secret (i.e. the key K_{ObjID}). This secret is used only to verify that the consumer and the time of access request fit the access policy imposed by the owner. Thus, they cannot retrieve the user data with this key.
- **Confidentiality against the access of unauthorized consumer**:
  In our proposed protocol, the legitimate consumer who has the access privilege is the unique entity that can retrieve and decrypt the data. Even if the Synchronization Manager and the other Blockchain nodes have the Key K_{ObjID}, the key K_1 misses to compute the file encryption key K. In addition, even if the consumer has the key K_1 and he has not the privilege of accessing the owner data, he cannot retrieve the key K_2 from the Synchronization Manager.
  Another case is possible. The consumer retrieved previously the both keys K_1 and K_{ObjID} and actually he has no longer the privilege of accessing the data due to time expiration or policy changes. In this case, the Synchronization Manager and Blockchain nodes, when computing the new key K_{ObjID} does not allow him to grant authorization to retrieve and synchronize the new data version from the storage servers.
- **No user forgery**
  Using the blockchain and especially with the GetPolicy and SetPolicy transactions, the attributes of each user subscribed to our storage service, are known by all the network nodes. Therefore, there is no risk of attributes tampering and thus user identity forgery.

V. Conclusion

One of the main challenges of file sharing is the access control and key management. We present in this paper a novel decentralized access control model with a temporal dimension. First, the access control management in our model is based on an enhanced CP-ABE. Second, the decentralized access verification is based on Blockchain technology. Third, the temporal dimension is introduced without additional revocation cost.

We implement our Timely CP-ABE using the CP-ABE toolkit and the Multichain solution. We strongly prove that our model is in line with all the safety requirements in the context of file externalization and sharing.

As future work, we expect to focus in detail, besides the access control challenges, on the whole architecture of file sharing and synchronization with the security requirement.

References

[1] I. Drago, M. Mellia, M. M. Munafo, A. Sperotto, R. Sadre, and A. Pras, "Inside dropbox: Understanding personal cloud storage services," in *Proceedings of the 2012 ACM Conference on Internet Measurement Conference*, ser. IMC '12, 2012.

[2] M. Jemel, M. Msahli, and A. Serhrouchni, "Towards an efficient file synchronization between digital safes," in *2017 IEEE 31st International Conference on Advanced Information Networking and Applications (AINA)*, March 2017.

[3] "Bitcoin: an innovative payment network and a new kind of money," in *https://bitcoin.org/en/*.

[4] S. Adi, "Identity-based cyrptosystems and signature schemes," in *Advances in Cryptology-CRYPTO, Springer-Verlag Press*, 1984.

[5] J.Bethencourt and a. B. A. Sahai, "Ciphertext-policy attribute-based encryption," in *IEEE Symposium on Security and Privacy, IEEE Computer Society, Los Alam*, 2007.

[6] V.Goyal, A. O.Pandey and, and B. Waters, "Attribute based encryption for fine-grained access control of encrypted data," in *In Proceedings of ACM Conference on Computer and Communications Security (ACM CCS)*, 2006.

[7] M. Kallahalla, E. Riedel, Q. W. R. Swaminathan, and K. Fu, "Plutus: Scalable secure file sharing on untrusted storage," in *Proc. USENIX Conf. File and Storage Technologies*, 2003.

[8] E. Goh, H. Shacham, N. Modadugu, and D. Boneh, "Sirius: Securing remote untrusted storage," in *Proc. Network and Distributed Systems Security Symp. (NDSS)*, 2003.

[9] G. Ateniese, K. Fu, M. Green, and S. Hohenberger, "Improved proxy re-encryption schemes with applications to secure distributed storage," in *Proc. Network and Distributed Systems Security Symp. (NDSS)*, 2005.

[10] M. Baldi, F. Chiaraluce, E. Frontoni, G. Gottardi, D. Sciarroni, and L. Spalazzi, "Certificate validation through public ledgers and blockchains," *First Italian Conference on Cybersecurity (ITASEC17)*, 2015.

[11] G. Zyskind, O. Nathan, and A. S. Pentland, "Decentralizing privacy: Using blockchain to protect personal data," *IEEE CS Security and Privacy Workshops*, 2015.

[12] A. Ouaddah, rAnas Abou Elkalam, and A. A. Ouahman, "Towards a novel privacy-preserving access control model based on blockchain technology in iot," *Europe and MENA Cooperation Advances in Information and Communication Technologies*, 2017.

[13] "Advanced crypto software collection: Ciphertext-policy attribute-based encryption," *http://acsc.cs.utexas.edu/cpabe/, 2011*, 2017.

[14] "Multichain: Open platform for blockchain applications," *http://www.multichain.com/*.