Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

# Survey on Secure Cloud Data Sharing Services in Multi-level Environments

**K. Saravanankumar[1], Dr.B.Amutha[2]**
[1]Department of Computer Science and Engineering,SRMIST,Kattankulathur, India.
[2]Department of Computer Science and Engineering, SRMIST, Kattankulathur, India.

**ABSTRACT**
Free decentralized solutions allow real control of individuals over their data. By having the choice of hosting the platform, combined with the transparency of code executed, the owner ensures that his data is not analysed without his knowledge and reduces the risk of large-scale hacking. However, this does not solve a fundamental paradox: to be completely sovereign over his personal data, the owner must be able to control the server that stores them and understand the code that runs on it. However, this is only really possible if the owner is an expert. This paper presents a survey on secure cloud data sharing services in multi-level environments.
The paper details centralized, discretionary, mandatory and role-based access controls. The concepts of encryption and personalized cloud sharing is discussed.

**KEYWORDS**
Decentralized, Paradox, Cloud Data Sharing, Centralized, Discretionary.

## Introduction

As our lives become more and more digital and the **volume of datapersonal** explosion, it becomes vital to be able to protect yourself against surveillance and control that come dangerously close to the dystopias. Beyond the simple respect of private life, to which each individual can and must claim, the underlying issues go far beyond the individual sphere and ask real questions about the democratic impacts and the free arbitrator of individuals [1].

This centralization also poses **serious security problems**. Indeed, regroup data in one place provides significant gain to an attacker who manages to infiltrate the system. The cost / benefit ratio of an attack is therefore very interesting, and it's not surprising to see examples of data breaches on the rise in recent times years [2]. To name just a few: 68 million Dropbox accounts have been affected by an attack in 2012, against 117 million for LinkedIn and 360 million for MySpace in 2016. As for the Yahoo! platform, all the accounts have impacted by attacks that started in 2013, i.e., 3 billion accounts.

In 2015, extra-marital dating site Ashley Madison saw its database exposed publicly on the Web, with sometimes dramatic consequences for married. In 2017, the agency American credit union Equifax announces that it has been affected by a computer attack that compromised over 145 million accounts. Leaked data includes among other things mailing addresses, social security numbers and banking information. This same year, founder of the Pirate Party, announces that the Swedish government has accidentally leaked a database of personal information of his fellow citizens, as well as extremely sensitive military data [3]. It does not have was the result of an attack, but of simple negligence: The Ministry of Transport outsourced the processing of this data in an IBM Cloud and communicated access by e-mail in clear.

Finally, beyond negligence or attacks from malicious third parties, **abuse** can also occur directly by those in charge of **administration** of the service. Cases of harassment, stalking or blackmail by Google, Uber or Apple employees were notably reported, and no doubt from many others were overlooked. To face this situation, the World Economic Forum formulated the need to see emerge personal data management platforms allowing each individual to collect, manage and **share** their data in different contexts, with real **guarantees of security and protection of privacy [4]**. The principle of the **Cloudpersonal**, is precisely

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

that. In this approach, each individual has his own personal space, under his control, on which he can confidently store his digital heritage and manage it through applications. In this paradigm, the principle **of *empowerment*** is essential. Translated as "empowerment" in Quebec, it implies empowerment of individuals, so that they are no longer simple dependent users' system, but full players in their digital life.

Personal Cloud is **decentralized** by nature, i.e., the user has the choice of hosting and supplier. It is in direct opposition to traditional approaches, where the service provider imposes its centralized infrastructure on all individuals who then no longer have any control over how their data is stored and used, with the risks of abuse that we have detailed[5]. The **governance** changes so completely: the individual is no longer a prisoner of a service that holds his data and has the freedom to change if they are not satisfied with it: they can also choose a host commercial ina data-center than an associative hosting that puts ethics at the center of its concerns, for example the CHATONS 2 collective. In fact, the user can even do without it completely and decide to host your digital heritage at home, if it has the skills. In addition, decentralization also brings an additional guarantee resilience: if a centralized service becomes inaccessible, following a failure, attack, or simply a shutdown, all the data of all users becomes unreachable and are potentially lost forever [6]. Finally, the decentralized aspect mitigates greatly the risk of being the victim of a massive attack: the cost / benefit ratio of an attack is reversed compared to a system where everyone's data is grouped together in one place.

In this type of environment, sharing is particularly complex to implement, because it is at the intersection of many issues, which we will discuss in the following: access control, identification and authentication of subjects, encryption, synchronization, revocation, etc. The challenge is to be able to give individuals the possibility of easily control which third parties can access which data, which third parties can be as well other individuals or online services [7]. Unlike digital safes which focus on data security to the detriment of uses, the personal Cloud gives pride of place to applications and is part of the social and collaborative dimension of Web: individuals are constantly seeking to share resources of all kinds, whether in their private or professional sphere. However, this dissemination does not go risk-free: Sharing policies can be bypassed in various ways, even be badly perceived by users who then lose control over their own data. He It is therefore essential to have the means to guarantee the security of sharing documents in the personal cloud.

## Secure Document Sharing in the Personal Cloud

### 1) Centralized Access Control

To ensure the safety (Cherdantseva et al., 2013) [8], the three fundamental properties are considered as **availability, data integrity, and confidentiality** in an information system. Based on the principles like unavailability, incorrect modification, and unauthorized observation, the data security faults are classified (Bertino 2005) [9].

Access control refers to all the techniques implemented to restrict access to resources. It is therefore positioned both on respect for **confidentiality** and data **integrity**.

The application of access control, also called enforcement, is ensured by a module called a **reference monitor** which intercepts all data access and determines whether they are legitimate or not. All the rights and conditions applied by the reference monitor constitutes the **access control policy** that can be represented and expressed in multiple ways. Note that a distinction is made between authentication and access control. The reference monitor generally considers that an access request is already correctly authenticated when evaluates it [Sandhu 1994], although this is not always true, typically in so - called *credential-based systems*. In these models, each user authenticates via a token that contains permissions.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

Finally, although access control and authentication are essential components, the security of an information system cannot be reduced to this. Application of techniques encryption, audit or denial of service protection (Sandhu and Pierangela1994) [10], (Bertino et al., 2011) [11] are all means to reduce attack surfaces. This also applies to sharing, which affects many aspects of information system security. The access control issue focuses in this paper as it is the key aspect, but other components of security don't neglect without contributing directly to it.

Some models of conventional access control are detailed and used extensively in the industrial world.

### A. Discretionary Access Control

Commonly referred to by the acronym **DAC**, for Discretionary Access Control, this model allows you to associate the identity of a subject with a set of permissions on objects. When a subject makes a request on an object, the action is granted if and only if an authorization allows it (Griffiths et al., 1976) [12].The key notion of CAR is the **discretionary** aspect which results in a delegation of rights: each user of the system can himself assign rights to other subjects on the objects for which it has permissions. On the other hand, it cannot assign rights that he does not have.
In most DAC models, permissions are represented as **listsaccess control**, or **LCD** for Access Control List. Each element of an ACL represents an authorization for a subject on an object, associated with an access mode (read, writing, etc) [13]. The evaluation of access control can then be reduced to going through the ACL lists and verify that the requested action has a registered authorization.

The vast majority of operating systems (Unix, Windows, etc.) use a DAC model for their access control: users can thus transfer their rights to others users on the data or services they control without needing any authority central which controls everything. This discretionary aspect makes it flexible and suitable for many multi-user systems. However, it makes it difficult to control the spread of data: Once an owner grants access for a user to an object, they do not have
no more control over future accesses that may be created. This is precisely what mandatory access control models seek to avoid.

### B. Mandatory Access Control

The Mandatory Access Control **(MAC)** model is used to define access policies by a predefined classification of objects and subjects. Along with the DAC model, it is one of the two major classes of access control and has inspired many works.

Historically associated with the US government, this model appeared in the years 1960 and was initially intended to ensure the protection of sensitive military data, by particularly against attacks by Trojan horse 4, which are particularly dangerous for discretionary systems like DAC where an infected station could delegate all its access to an attacker.

In the model, each object is associated with a level of sensitivity, and each subject, an authorization level. The values of these levels are defined in classes access, made up of two elements:

- A security label, for example, _Top Secret '(TS) or _Confidential' (C), where TS> C in terms of security.
- A category, for example _Aviation ', or _Army'. This allows to know the context in which data access takes place and define classes different access depending on the domain.

The two central properties of access control are: *no read up* and *no write down*. This means that a subject can only read an object if its access class is higher than that of object and can only write an object if its

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

access class is lower.
In this very rigid model, users cannot under any circumstances assign rights by themselves, even though they are the "owner" of the data. A central administrator decides, a priori, the access policy and classification, which prevents users from be responsible for the degree of exposure of the data to which they have access. This system is particularly justified in organizations where access to data is extremely sensitive, typically in military or power plant contexts nuclear.

It should be noted that the MAC approach is not necessarily incompatible with DAC. In Indeed, some systems allow them to be combined to benefit from the best of both worlds.

### C. Role-based Access Control

The cost of maintaining access control can quickly become prohibitive in previously cited models. Particularly in industrial systems where access to objects can be done by hundreds or thousands of subjects. The management of these authorizations causes a combinatorial explosion that slows down the evaluation of access control and makes complex attribution or revocation of rights. In addition, neither the rigidity of the MAC models, nor the lack of control of DAC models are not really satisfactory for systems industries where authorizations must be able to be delegated, but also controlled.

It was in response to these issues that role-based access control appeared, referred to as Role-Based Access Control **(RBAC)** (Sandhu et al., 1996) [14]. The Figure 1 illustrates the mechanical RBAC: Subjects obtain permissions on resources through roles that give them are assigned, themselves associated with a set of permissions. As there are normally significantly fewer roles than users and resources, this simplifies greatly managing permissions. Different users can play the same role, for example _manager 'and the same user can play different roles, which allows mutualize the authorizations according to the tasks assigned to the subjects. In addition, a hierarchy system between roles simplifies its administration: each child role inherits permissions from its parents, for example, the role of 'IT manager' can inherit the manager 'role and its permissions, while benefiting from permissions specific to its child role. This avoids having to redefine permissions for each role and allows manufacturers to have a mental model that echoes their hierarchical organizations.

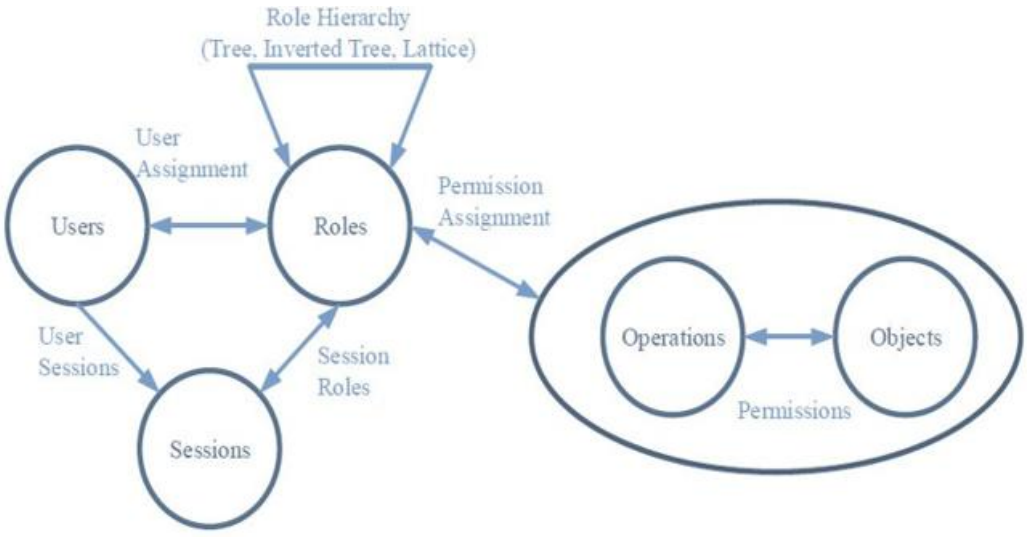


**Figure 1. RBAC model**

This access control mechanism is intended to be a credible alternative to DAC and MAC, in especially for large scale scalable systems. Nevertheless, it is important to point out that RBAC does not necessarily

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

oppose these models. Thanks to its flexibility and because it does not impose a single way to apply access control, it is possible to implement DAC (Sandhu et al., 1998)[15] and simulate MAC (Osborn et al., 2000) in RBAC.

Finally, although originally designed for large-scale systems that can accommodate many users, it appeared that the maintenance of access control can become extremely complex to manage hundreds of thousands or even millions of users. In particular, manual role assignment can become nightmarish. To palliate this problem of scale, solutions have been proposed to automatically assign roles to users, through the use of rules evaluated on user attributes (Al-Kahtani et al., 2002). This paved the way for a new family of control access that would take full advantage of the emergence of web services and their need for flexibility and dynamism.

### D. Access Control based on Attributes

Access control based on attributes, or Attribute-based Access Control **(ABAC)** appeared in the 2000s (Yuan and Tong 2005) [18] to meet certain limitations of access controls mentioned above, in particular vis-à-vis web services. These latter are dynamic in nature and not suitable for systems like MAC or RBAC where permissions are based on predefined identities or roles. In addition, they require a more powerful and refined semantics in order to express access to data.

The assignment of permissions in ABAC is based on the attributes that can be set on subjects, resources or environment. A subject can either be a user or an application, and a resource can be data as well as a web service (Sadia Kauser et al., (2018) [19]. The environment represents the context where access control is applied, and can relate to the date, the nature of the network, etc.

For example, the following rule is used to express access to my online music service for any topic that is part of my friend group:

$$\text{can\_access}(s, r, e) \longleftarrow \text{Group}(s) = \text{'Friends'} \wedge \text{ServiceName}(r) = \text{'MusicPlayer'}$$

Here, the rule relates both to the attributes of subjects *s* (by belonging to the group _ *Friends')* and on the attribute of the name of the service which is a resource *r*. ABAC allows a great flexibility of expression which does not exist in RBAC and thus avoids the definition and management of multiple roles, which becomes inevitable when services and contexts are multiply.

Two entities, called Policy Decision Point (PDP) and Policy Enforcement Point (PEP), are in charge of the access control application and play the role of reference monitor. When a subject seeks to access a resource, the request is intercepted by the PEP which is the essential entry point of the system. It then transmits the request to the PDP which retrieves the access rules and the attributes concerned in order to determine whether the request access is legitimate. It sends its decision to the PEP which grants or denies access to the resource depending on the response.

ABAC thus makes it possible to express many access rules with fine granularity, at attribute level. He is nonetheless agnostic regarding the format of access rules. He is therefore possible to use rule languages such as XACML (Anderson et al., 2005) or EPAL (Ashley et al., 2003) [20-21] which express a wide variety of control policies access, formatted in XML. These languages emphasize interoperability, so that heterogeneous systems are capable of transmitting access policies to one another and avoiding redefinition of *ad-hoc* rule formats for each use. Although EPAL has been defined first via a W3C recommendation, it was XACML that finally won, especially in industry.

Although more flexible than the above-mentioned approaches, ABAC can nevertheless make the

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

understanding and administration of access control quite complex: it is difficult,from a set of more or less elaborate rules, to apprehend the concrete implications of such a policy and knowing exactly who can access what, in what conditions. On-the-fly evaluation of requests to determine their legitimacy with respect to a set of predicates on attributes is beyond the scope of the understanding of most individuals and therefore not compatible with the property **of *empowerment*** personnel Cloud.

### E. Authentication OpenID

(Seong et al., 2010) and (Yeung et al., 2009) [22-23] propose sharing mechanisms which authenticate the topics by **OpenID** (Recordon and reed 2006) [24]. In this protocol, a user authenticates to third-party services using an identity subscribed with an authority that potentially has no link with the services accessed.

The authentication of a user Jon wishing for connecting to the *Favorite Knowthing* web application is illustrated. In order not to burden the reading, some details techniques are deliberately eclipsed:

1. Jon goes to *Knowthing's* login *page* which plays the role of *Relying Party* (RP), and enters its OpenID *jon.snow*, previously subscribed after a supplier Identity (IdP).
2. The OpenID identifier is a URI that points to a document indicating Jon's IdP Snow, *stark.corp*. The RP then redirects the latter to the relevant IdP.
3. The IdP asks Jon Snow to authenticate using the password provided during creation of its OpenID. If he knows it, the authentication is successful and Jon Snow is redirected again.

This protocol has greatly contributed to popularize decentralized authentication and has known growing interest since its first publication in the mid-2000s. Unfortunately, the need to use trusted central authorities, embodied by identity providers, leaves users vulnerable to shutdowns of services, symbolized by that of MyOpenID in 2014. Furthermore, various security weaknesses (Hossein et al., 2020) [25] have contributed to a gradual loss of confidence in the protocol.

### F. Web of Trust

Completely decentralized approaches, i.e., not based on any authority central. Many of them are inspired by the principle of ***Web of Trust*** **(WoT),** introduced in the early 1990s by PGP (Zimmermann and P.R1995) [26]. Originally, WoT allows users to exchange encrypted messages, while by having guarantees on their integrity and the identity of the issuer, thanks to a network trust established between the participants.

It is assumed here that each individual has a pair of private and public keys which are used to encrypt and sign the data. The **encryption** is used to ensure the **confidentiality** of data, while the signature ensures their **integrity** and is also used here to **authenticate** the transmitter. The public key is, as its name suggests, accessible to everyone, and is used to encrypt data and verify signatures. Conversely, the private key must only be known by its owner and is used to decrypt and sign the data.
For example, suppose that user Jon wants to communicate a private message to his friend Ygritte. The following steps, take place:

1. Jon generates a temporary session key and encrypts it with Ygritte's public key.
2. Jon encrypts his message with the session key and generates a hash with a hash function. Then, he encrypts the fingerprint with his private key to obtain the signature of the message.
3. Jon brings together the encrypted message, the encrypted session key and the message signature, and sends it all to Ygritte, via a potentially unprotected channel.
4. Ygritte decrypts the session key using his private key.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

5. Ygritte decrypts the message using the session key obtained.
6. Ygritte checks the validity of the signature by decrypting it with Jon's public key and compares the result with the fingerprint of the decrypted message. If she gets the same thing, everything went well.

This is the case of Lockr, (Tootoonchian et al., 2009) [27] who notes that sharing data via centralized and closed platforms make the user dependent on their access control and can hardly, or not at all, share with his contacts who use others platforms. In Lockr, users share their data by creating ACLs that indicate the relationship required to access the object. A relationship, for example "family" is composed of a set of users where each member of the relation has the key members' public and mutually sign their keys. A link between two members is proven by a social certificate which contains the public key, the wording of the relationship and a relationship key shared by all members and the signing of the certificate.

(Van Kleek et al., 2012) presents WebBox, a system that also allows users to share data between them in a decentralized manner. Each user of the system has a personal server on which he hosts his data, as well as a public profile, accessible via a semantic URI 5, called **WebID**. Besides information on the owner's identity (last name, first name, e-mail, etc.), this profile contains his public key and the list of contacts, in FOAF 6 format. There is no need to sign the public key here with other third parties as in the original WoT because it is linked to the user's public profile referenced by the WebID. The owner signs his key with the content of his public profile and its WebID; it therefore becomes much more complicated for a malicious third party to spoof an identity, as long as the contacts correctly verify the signature and URI. In WebBox, all permissions are represented by ACLs, as RDF 7 triples *<subject, predicate, object>*, where subject is represented by its WebID, the predicate matches the HTTP action it can perform on the object (GET, POST, PUT, etc), and the object is the granted resource, also in the form of a semantic URI. All data are thus represented in RDF and accessible via the SPARQL 8 query language.

## G. Encryption

In traditional the *enforcement* is responsibility of information system administrators who have tools and specific skills to help them in this task. It is much more difficult to do in decentralized approaches centered around the user, since the latter often does not have control over the cloud server and does not have control anyway necessary to control what is happening there. However, service providers may reveal unscrupulous and **confidentiality** attacks are numerous 9,10. A way to protect against it is to make the application of access control inseparable from encryption.

(Thilakanathan et al., 2014) [29] and (Wang et al., 2016) [30] focus on the protection of data in centralized clouds where little or no trust can be placed in the server. Despite centralized data storage, access control remains decentralized because it is defined and maintained from the owner's device (s). The data is thus kept encrypted in the Cloud, while allowing the user to define sharing rules on it without exposing them to the provider. [29] and [30] propose to use Attribute-Based encryption Encryption **(ABE)** (Sahai et al., 2005) [31] to protect and access documents based on an access control by attributes. The owner places tags on his documents, for example example *type = 'Medical'* or *date = '2017'*, and uses them to express its sharing policy.

Typically, to share all cardiological documents from before 2017, a rule could be expressed like this: *(type = 'Cardio' AND date <2017)*. For each rule, a decryption key is communicated to the subjects concerned which contains the expression of the rule and allows them to decipher the documents that go into its assessment. Persona [Baden 2009] also uses ABE encryption, but applied to social networks. The owner creates groups via attributes ("family", "rock band", etc.) and distributes them to them associated encryption keys, encrypted with their public key. Each group can accessto certain resources, specified by ACLs, produced and stored in applicationsnetwork. Each application can define its own access control

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

semantics, byexample *post* for a microblogging application or *chat* for a microblogging applicationmessaging. Each application has a centralized home on which the dataare stored encrypted and permissions are applied. Users must thereforehave confidence in their applications to correctly apply their access policy.

Besides performance, a weakness of these ABE encryption-based jobs is that the attributes are not protected from a **privacy** attack on the server. That means that an outside observer could infer sharing policies and determine what is shared, with whom. Predicate encryption-based tracks could address this problem, but they also significantly impact performance.

In (Ali et al., 2017) [33], the trust is deported to a cryptographic server, in charge of encryption operations and access control assessment. Unlike the server of Cloud, the crypto server must have the full confidence of the user. Unfortunately, the honesty and robustness of the server cannot be verified by the owner who must blindly trust in the correct application of encryption and access control.

Another possible way to share encrypted data from a non-server trust is the use of a *re-encryption proxy*, or *proxy re-encryption* (PRE). Again, the data is kept stored on a Centralized cloud. For each share with a subject, the owner generates a re-encryption and sends it to the proxy, so that it re-encrypts the data concerned, so that that the subject can decipher them, without the proxy ever having access to the clear. Unlike ABE approaches, the proxy does not have any information, except know what subjects the owner communicates with.

Finally, (Yuan et al., 2015) and (Guha et al., 2008) [34-35] prefer to use obfuscation strategies, in order to be able to use existing sharing and social networking platforms, transparent. These two systems assume that the keys are exchanged beforehand. between owners and their subjects through secure channels. implements algorithms for scrambling photos at various levels depending on their degree of sensitivity. The unencrypted viewing is only possible if the recipient has the decryption keys and may be only partial depending on the owner's preferences. He can for example decide that a subject can only see his face in clear, in a group photo. uses data substitution techniques on social media platforms where privacy is not guaranteed, like Facebook. However, these platforms generally prohibit obfuscation in their terms of use and may go until ban in case of infringement. This is explained both for reasons user experience, to avoid having unreadable content in the news, but especially for commercial reasons, given that their *business model* is very largely on data collection and user *profiling*. the private data of an owner is thus randomly replaced by data from other people. Only authorized contacts can view the data real thanks to their decryption keys, which is done automatically thanks to a dedicated web extension. Unfortunately, this makes mobility complicated as it becomes impossible to use the platform from devices without the extension, in particular on mobile.

Finally, the heaviness and complexity of access control discourage users who end up seeing it as a burden. They thus find themselves most of the time to define access policies that are much more permissive than they would like, sometimes even without realizing it (Liu et al., 2011)(Mazurek et al., 2010) [36-37] and using services known to have few scruples about respect for private life. So that the personal cloud can establish itself as a credible alternative, it is essential to put the individual back control and decision-making center, while providing it with the appropriate tools to can perform these tasks as intuitively as possible, while maintaining smooth user experience.

### H. Administration of Objects

Today, most of the platforms that offer their users the possibility of sharing their data offers a very manual and not very dynamic approach. Well Often, users are forced to express their shares on a case-by-case basis, with little possibilities of automation and even less of **transversality**: it is typically very

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

complicated to express the sharing of all the documents associated with a particular context, whatever their type. For example, it would be practical to be able to express everyone's sharing documents relating to a *road trip* with friends (photos, expenses, GPS tracks, etc.) or for a business project (agenda entries, contact of each participant, report of meetings, etc) without having to redo a manual action for each new document concerned by this type of sharing. This transversality is specific to the personal Cloud which aggregates data of all types and does not have a fixed data model: the possibility adding and removing applications and new uses inevitably lead to **versatility** that does not exist in traditional information systems. Moreover, the data can be added at a high rate, for example a time series of power consumption: in this case, the sharing cannot be static but must be able to update to capture any new document or legitimate update to enter the permissions.

(Geambasu et al., 2007) [38] offers a peer-to-peer solution to easily express shares transversal and dynamic. An SQL-based language allows applications to create views on owner data and to pass *capabilities* to relevant subjects which are authentication tokens giving access to the results of a view. The subjects thus directly request views by passing in their *capabilities* to retrieve the data with the owner. The permissions are therefore associated with the results of views, which allows you to manage the dynamic aspect of personal data. If data are added, modified or deleted, and the result of a view changes, the subjects can restart a request to retrieve the updates, without the owner having to redefine accesses manually.

[37] introduces Penumbra, a distributed, tag-based file system that aims to simplify both file administration and sharing for people not techniques. The use of tags is justified by the fact that users would have one more intuitive understanding and use than traditional file systems hierarchical (Seltzer et al., 2009)(Klemperer et al., 2012) [39-40]. It also gives the possibility of having a cross-sectional view of the data which can be files, notes, agenda, etc. The owner of a device therefore classifies his data by assigning them tags that can be queried and combined into logical expressions. For example, *type = photo* ∧ *album = Iceland17* returns the 2017 summer vacation photo album.

Similarly, the owner shares his data by assigning access to his tags. As the tags are at the heart of the access control policy, they are signed with the key owner's privacy to ensure that no unauthorized apps or third parties can modify them. Access rules are also signed and verified by a *software* system.*based* (Wobber et al., 1994) [41] which makes it possible to ensure that only an accredited subject is capable of forge the evidence necessary to execute his request.

(Riva et al., 2011) [42] defines a rule semantics adapted to the context of the personal Cloud, seen here like the union of an owner's devices. These rules make it possible to replicate data between devices, depending on their type (photos, contacts, etc.), or items contextual (importance of data, device memory space, etc.). The rules of sharing are expressed in Prolog 12, a logic programming language. It is also the case of which allows to express access control rules for distributed data, in a language inspired by Datalog 13, itself based on Prolog.

The systems of sharing rules utilize in this work that allow the owner of easily expressing transverse views on his data, whether by languages inherited from SQL (Geambasu 2007) [38], from Prolog (Riva 2011) [42] or by tags (Mazurek etal., 2014) [37]. In any case, this avoids the owner having to Manually express each share for each type of data, as is mostly the case today. Based on the automation of creation of rule, if it’s not possible to go even further is being informed.

## I. Rules Administration

Creating access rules can be a tedious task. Although the works previously cited considerably reduce this burden by simplifying the expression of subjects and objects, this requires manually activating rules to

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

represent the sharing policy. However, the progress made in artificial intelligence leaves to think that these rules could be directly inferred from habits and preferences of the user. (Fang et al., 2010) [43] thus propose to use algorithms of machine learning to automatically infer sharing policies from user data. (Fang et al., 2010) [43] starts from the observation that individuals tend to design their privacy preferences in the form of implicit rules, sharing certain data on social networks with certain types of people. In order to build a sharing model adapted to each user, some initialization data is manually requested from the owner. The more information the latter provides, the more classification system is accurate. In addition, all the social data of the owner are analyzed in order to extract communities of individuals with whom the owner has common release preferences. Each new data added by the owner is then classified and exposed to certain communities, in depending on the construction of the model. Note that advanced users can also visualize their model in the form of a decision tree and modify them to their as you wish.

(Squicciarini et al., 2011) [44] uses an image classification model trained over a wide set of photos, in order to be able to infer the desired degree of exposure for each image added by the owner, from what it represents and its metadata. He must also specify its privacy preferences so that the classifier can adapt to its specificities. Rules semantics are provided for the expression of preferences which allows combining simple predicates.

These studies claim to obtain good results in terms of the precision of their models, greater than 90%. At the same time, *machine learning* algorithms are increasingly powerful and see the flourishing of multiple applications, ranging from music recommendation to automatic tag creation on photo albums. Nevertheless, applied to control access, the consequences of misclassification can be much greater a song in bad taste or an imprecise tag: this can lead to unshared shares. wanted on potentially sensitive data. In addition, it is not certain that the users trust algorithms they have no understanding of nor real control.

### 2) Personal Cloud and Sharing

An explosion in the volume of personal data is witnessed as human lives are digitized. It is becoming more and more important to have tools to store, manage and share its data in a secure and sustainable way. Little by little, the traditional file systems have given way to cloud solutions, where data is no longer stored on a hard drive of the family computer, but in dedicated infrastructures that ensure data synchronization between the different owner's devices and guaranteed automatic backup.

### A. Centralized Solutions

As the centralization of data poses many problems in terms of security and privacy. However, the popularity of solutions current centralized systems makes them essential. (Bocchi et al., 2015) [45] analyzed the traffic internet of over 20,000 households and showed that around 25% of them used a customer Dropbox 15 or Microsoft OneDrive 16 and 10% for Google Drive 17. All these solutions offer a classic file sharing system, that is to say either public which gives access to the resource pointed to by anyone with the generated URL, be restricted to users of the solution used. This corresponds to a DAC share, which lacks flexibility in the expression of sharing. However, the centralization of data provides an undeniable advantage for the sharing: any shared document corresponds to an access created for a recipient who receives a version synchronous with that of the owner, since it is the same data.

This therefore allows **collaborative** sharing, where any update is propagated between participants of a share. For these Cloud services, the notion of *personnel* is quite relative. First, all data stored by the individual ends up in a single silo, owned by the company which edit the service.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

### B. Decentralized Solutions

Many tools exist which allow technical users to store and use their data while ensuring their privacy. For example, it is quite possible to configure an (S) FTP 19 server on a home computer and use clients from synchronization on terminals, based on rsync 20 scripts and cron 21. Sharing can be done by creating separate users and manually configuring their rights. Forget to assign a static IP to the server so that it is accessible from the outside, give it a domain name, and configure a NAT port 22 forwarding from the local router, if it is not possible to use a public IP.

The solutions are considered that only require the utilization of a server to store data and use applications for manipulation and sharing of it. It is right to specify that other approaches, known as *serverless*, exist, which allow storage fully distributed among peers, like SyncThing 23, MaidSafe 24 and Storj 25, but limit the uses that can be made on the data and focus exclusively on the files.

In addition, these initiatives defend the use of free software 28 which is a condition important, if not essential, to the digital emancipation of the individual. Difficult indeed to trusting software that uses code that is not accessible, which amounts to executing a black box on his personal data. Transparency is an important component self-accommodation and the resulting trust; even if the end user does not verify not the code himself, he can reasonably trust the community to do this work and ensure that the software can be trusted. Finally, at- beyond the confidence that free projects can bring, the freedom to be able to recover the code, run it yourself and even modify it provides guarantees on the sustainability of the system. In fact, users of centralized solutions are both dependent on the software and service infrastructure. If the entity behind the service decides to shut it down, as it happens regularly (D. C. Nguyen et al., 2019)(Hermannet al., 2009) [46-47], it becomes impossible to reuse it and all data becomes inaccessible. Conversely, if the maintainers of a free and decentralized project decide to throw in the towel, it is always possible for other people to retrieve it. And like the data is stored in a place under the control of the individual, the latter does not have to fear a sudden shutdown that would deprive it of its data forever. It is however important to specify that self-hosting is not necessarily synonymous with software free. For example, Lima 29 is a nonfree project that consists of storing all of its data on a hard drive at home, connected to a connected box that provides access to the drive from any device. Likewise, free and open-source should not be confused.

(Stallman et al., 2016) [48], the second being a more lax notion of the first. Even if initiatives seek to simplify it, self-hosting remains a complex subject which can only reasonably be achieved with the intervention of an expert in the field. Projects like ownCloud 30, nextCloud 31, Sandstorm 32, Cozy 33, Camlistore 34, Pydio 35, Seafile 36 or (X. Zheng et al., 2018) [49] have a pragmatic approach: the location of the storage and the place of execution of the software are left free to the individual: this means that can either install it on his own machine, if he has the necessary knowledge, or delegate administration to a specialized host, possibly for a fee. Own Cloud is a pioneer in the field of personal cloud. Created in 2010, it had for vocation to offer the same file storage and synchronization functionalities as the centralized giants, without compromising on individual privacy.

Following internal disagreements (Karlitschek et al., 2016) [50] and thanks to the free license of the project, nextCloud was launched in 2016 as a *fork* of ownCloud, i.e., a new software branch based on the same source code. Both projects use a common API (GEANT 017) for file sharing, based on WebDAV 37 and developed by a federation of partners 38. This specification aims to allow interoperable sharing between different Decentralized personal cloud platforms, eg Pydio. Indeed, power sharing between heterogeneous systems is essential to hope to compete with giants centralized and allows the pooling of efforts. Unfortunately, this specification, originally designed by ownCloud, is focused on file and directory sharing and therefore does not address the issue of an agnostic model vis-à-vis the types of document.

Annals of R.S.C.B., ISSN:1583-6258, Vol. 25, Issue 3, 2021, Pages. 165 - 179
Received 16 February 2021; Accepted 08 March 2021.

Sandstorm allows applications to run in containers isolated from each other others, while leaving them the possibility of communicating thanks to a low-level communication, Capnproto 39, with *capabilities* authentication. Grace to their container system, it is theoretically possible to run any application in Sandstorm, developed in any language, with a specific configuration. And any app can implement the sharing API provided by Sandstorm. This API is based on the principle of delegation which takes up the key concept of DAC: any subject having rights to an application can delegate these rights, or a sub- together, to other subjects by communicating the associated *capability to them*. This approach works particularly well for collaborative contexts which is the bias of Sandstorm. However, this type of sharing becomes dangerous for more contexts intimate, whether for personal photos or sensitive documents, because it becomes very complicated for the owner to regulate his access and control who can access what.

Finally, Cozy the solution edited by the company Cozy Cloud, is a Cloud platform staff close to Sandstorm, where the data is not just files to synchronize, but can be of multiple natures, such as events, memos, GPS tracks, bank details, etc. But unlike Sandstorm, the applicationsinstalled are essentially**client-side**, i.e., they run in the browser, in a dedicated subdomain.

## Conclusion

Access control administration is a central and complex task, usually provided by security experts. The management of access policies, users, even roles or attributes, is not an easy task and beyond the reach of individuals non-experts, especially when the policy is based on a set of rules expressed in a specific language, and evaluated on the fly when requests arrive. The application logic, the users and their roles are generally identified a priori, that is to say during the design of the information system. Indeed, the policy of access control is often an integral part of the definition of the database schema. data and must be thought of from the start. Even ABAC, considered more flexible, requires having attributes on unified data schemas to evaluate properly access control. However, in the personal cloud, applications can be added dynamically by the owner to meet specific uses and unpredictable. Each of them can have its own structures and its own vision of how data must be represented in order to implement its uses. This makes these models unsuitable for personal Cloud use where it is impossible to predict which applications, data models and users will be defined.

In addition, specific authentication protocols have been proposed in an attempt to manage the identities of the subjects, fragmented on the Web; each identity is then associated with a set of permissions. Unfortunately, these protocols often induce a technical complexity that puts off most users who find it difficult to know exactly what is shared and with whom. This introduces the problem of *enforcement* permissions, i.e., the correct application of the sharing policy. Techniques of encryption used to assure on centralized platforms not controlled by individuals and guarantee the confidentiality of data. But these techniques impact considerably performance and degrade the user experience. In addition, the complexity of these models makes it difficult for the general understanding of the individual vis-à-vis how its policies are applied. Having no tangible guarantee, its *empowerment* is thus broken and the owner has no choice but to place full confidence in the system, without even having the possibility of knowing if anything goes wrong.

## References

[1] Sengupta, Jayasree, Sushmita Ruj, and Sipra Das Bit. "A Comprehensive survey on attacks, security issues and blockchain solutions for IoT and IIoT." *Journal of Network and Computer Applications* 149 (2020): 102481.

[2] Rahman, Sawsan Abdul, Hanine Tout, Chamseddine Talhi, and Azzam Mourad. "Internet of

Things Intrusion Detection: Centralized, On-Device, or Federated Learning?" *IEEE Network* (2020).

[3] Sha, Kewei, Wei Wei, T. Andrew Yang, Zhiwei Wang, and Weisong Shi. "On security challenges and open issues in Internet of Things." *Future Generation Computer Systems* 83 (2018): 326-337.

[4] Zhang, Kuan, Kan Yang, Xiaohui Liang, Zhou Su, Xuemin Shen, and Henry H. Luo. "Security and privacy for mobile healthcare networks: from a quality of protection perspective." *IEEE Wireless Communications* 22, no. 4 (2015): 104-112.

[5] Pereira, Renata IS, Ivonne M. Dupont, Paulo CM Carvalho, and Sandro CS Jucá. "IoT embedded linux system based on Raspberry Pi applied to real-time cloud monitoring of a decentralized photovoltaic plant." *Measurement* 114 (2018): 286-297.

[6] Sandor, Voundi Koe Arthur, Yaping Lin, Xiehua Li, Feng Lin, and Shiwen Zhang. "Efficient decentralized multi-authority attribute-based encryption for mobile cloud data storage." *Journal of Network and Computer Applications* 129 (2019): 25-36.

[7] Ngo, Canh, Yuri Demchenko, and Cees de Laat. "Multi-tenant attribute-based access control for cloud infrastructure services." *Journal of information security and applications* 27 (2016): 65-84.

[8] Cherdantseva, Yulia, et Jeremy Hilton. «A reference model of information assurance &security.» *Availability, Reliability and Security (ARES).* IEEE, 2013. 546-555.

[9] Bertino, E. & Sandhu, R. «Database security-concepts, approaches, and challenges.» *IEEE Transactions on Dependable and secure computing* 2, n° 1 [2005]: 2-19.

[10] Sandhu, R.S., & Pierangela S. «Access control: principle and practice.» *IEEE communications magazine*, 1994: 40-48.

[11] Bertino, E., Gabriel G., & Ashish K. «Access control for databases: Concepts and systems.» *Foundations and Trends® in Databases*, 2011: 1-148.

[12] Griffiths, Patricia P., et Bradford W. Wade. «An authorization mechanism for a relational database system.» *ACM Transactions on Database Systems (TODS)*, 1976: 242-255.

[13] M. Jemel and A. Serhrouchni, "Decentralized Access Control Mechanism with Temporal Dimension Based on Blockchain," 2017 IEEE 14th International Conference on e-Business Engineering (ICEBE), Shanghai, 2017, pp. 177-182, doi: 10.1109/ICEBE.2017.35.

[14] Sandhu, R.S., Coyne, E.J., Feinstein, H.L., & Youman, C.E. «Role-based access control models.» *Computer*, 1996: 38-47.

[15] Sandhu, R.S., & Munawer, Q. «How to do discretionary access control using roles.» *ACM workshop on Role-based access control.* 1998. 47-54.

[16] Osborn, S., Sandhu, R., & Munawer, Q. «Configuring role-based access control to enforce mandatory and discretionary access control policies.» *ACM Transactions on Information and System Security (TISSEC)*, 2000: 85-106.

[17] Al-Kahtani, M., & Sandhu, R. «A model for attribute-based user-role assignment.» *Computer Security Applications Conference.* IEEE, 2002. 353-362.

[18] Yuan, E., et Tong, J. «Attributed based access control (ABAC) for web services.» *ICWS.* IEEE, 2005.

[19] Kauser S., Rahman A., Khan A.M., Ahmad T. (2019) Attribute-Based Access Control in Web Applications. In: Malik H., Srivastava S., Sood Y., Ahmad A. (eds) Applications of Artificial Intelligence Techniques in Engineering. Advances in Intelligent Systems and Computing, vol 698.

Springer, Singapore. https://doi.org/10.1007/978-981-13-1819-1_36.

[20] Anderson, A. *Core and hierarchical role based access control (RBAC) profile of XACML v2. 0.* OASIS Standard, 2005.

[21] Ashley, P., Hada, S., Karjoth, G., Powers, C., et Schunter, M. *Enterprise privacy authorization language (EPAL).* IBM, 2003.

[22] Seong, S. W., Seo, J., Nasielski, M., Sengupta, D., Hangal, S., Teh, S. K., Chu, R., Dodson, B., et Lam, M.S. «PrPl: a decentralized social networking infrastructure.» *Workshop on Mobile Cloud Computing & Services: Social Networks and Beyond.* ACM, 2010.

[23] Yeung, C. M. A., Kagal, L., Gibbins, N., et Shadbolt, N. «Providing Access Control to Online Photo Albums Based on Tags and Linked Data.» *AAAI Spring Symposium: Social Semantic Web: Where Web 2.0 Meets Web 3.0.* 2009. 9-14.

[24] Recordon, D., et Reed, D. «OpenID 2.0: a platform for user-centric identity management.» *Digital identity management.* ACM, 2006. 11-16.

[25] Shafagh, Hossein, Lukas Burkhalter, Sylvia Ratnasamy, and Anwar Hithnawi. "Droplet: Decentralized Authorization and Access Control for Encrypted Data Streams." *In 29th {USENIX} Security Symposium ({USENIX} Security 20)*, pp. 2469-2486. 2020.

[26] Zimmermann, P. R. «The official PGP user's guide.» *MIT Press*, 1995.

[27] Tootoonchian, A., Saroiu, S., Ganjali, Y., et Wolman, A. «Lockr: better privacy for social networks.» *Emerging networking experiments and technologies.* ACM, 2009. 169-180.

[28] Van Kleek, M., Smith, D. A., et Shadbolt, N. « A decentralized architecture for consolidating personal information ecosystems: The WebBox.» *PIM.* 2012.

[29] Thilakanathan, D., Chen, S., Nepal, S., et Calvo, R. A. «Secure data sharing in the Cloud.» *Security, Privacy and Trust in Cloud Systems*, 2014: 45-72.

[30] Wang, F., Mickens, J., Zeldovich, N., et Vaikuntanathan, V. «Sieve: Cryptographically Enforced Access Control for User Data in Untrusted Clouds.» *NSDI*, 2016: 611-626.

[31] Sahai, A., et Waters, B. «Fuzzy identity-based encryption.» *Eurocrypt*, 2005: 457-473.

[32] S.J. De and S. Ruj, "Efficient Decentralized Attribute Based Access Control for Mobile Clouds," in IEEE Transactions on Cloud Computing, vol. 8, no. 1, pp. 124-137, 2020, doi: 10.1109/TCC.2017.2754255.

[33] Ali, M., Dhamotharan, R., Khan, E., Khan, S. U., Vasilakos, A. V., Li, K., et Zomaya, A. Y. «SeDaSC: secure data sharing in clouds.» *IEEE Systems Journal* 11, n° 2 [2017]: 395-404.

[34] Yuan, L., Korshunov, P., et Ebrahimi, T. «Privacy-preserving photo sharing based on a secure JPEG.» *Computer Communications Workshops (INFOCOM WKSHPS).* IEEE, 2015. 185-190.

[35] Guha, S., Tang, K., et Francis, P. «NOYB: Privacy in online social networks.» *Online social networks.* ACM, 2008. 49-54.

[36] Liu, Y., Gummadi, K. P., Krishnamurthy, B., & Mislove, A. «Analyzing facebook privacy settings: user expectations vs. reality.» *SIGCOMM conference on Internet measurement.* ACM, 2011. 61-70.

[37] Mazurek, M. L., Arsenault, J. P., Bresee, J., Gupta, N., Ion, I., Johns, C., Lee, D., Liang Y., Olsen J., Salmon B., Shay R., Vaniea K., Bauer L., Faith Cranor L., Ganger, G.R., et Reiter, M.K. «Access control for home data sharing: Attitudes, needs and practices.» *SIGCHI Conference on Human Factors in Computing Systems.*ACM, 2010. 645- 654.

[38] Geambasu, R., Balazinska, M., Gribble, S.D., et Levy, H. M. «Homeviews: peer-to-peer middleware for personal data sharing applications.» *SIGMOD international conference on Management of data.* ACM, 2007. 235-246.

[39] Seltzer, M. I., et Murphy, N. «Hierarchical File Systems Are Dead.» *HotOS*, 2009.

[40] Klemperer, P., Liang, Y., Mazurek, M., Sleeper, M., Ur, B., Bauer, L., et al. «Tag, you can see it!: Using tags for access control in photo sharing.» *SIGCHI Conference on Human Factors in Computing Systems.* ACM, 2012. 377-386.

[41] Wobber, E., Abadi, M., Burrows, M., et Lampson, B. «Authentication in the Taos operating system.» *ACM Transactions on Computer Systems (TOCS)* 12, n° 1 [1994]: 3-32.

[42] Riva, O., Yin, Q., Juric, D., Ucan, E., et Roscoe, T. «Policy expressivity in the Anzere personal cloud.» *Symposium on Cloud Computing.* ACM, 2011.

[43] Fang, L., et LeFevre, K. «Privacy wizards for social networking sites.» *World wide web.*ACM, 2010. 351-360.

[44] Squicciarini, A. C., Sundareswaran, S., Lin, D., et Wede, J. «A3p: adaptive policy prediction for shared images over popular content sharing sites.» *Hypertext and hypermedia.* ACM, 2011. 261-270.

[45] Bocchi, E., Drago, I., et Mellia, M. «Personal cloud storage: Usage, performance and impact of terminals.» *Cloud Networking (CloudNet).* IEEE, 2015. 106-111.

[46] D. C. Nguyen, P. N. Pathirana, M. Ding and A. Seneviratne, "Blockchain for Secure EHRs Sharing of Mobile Cloud Based E-Health Systems," in IEEE Access, vol. 7, pp. 66792-66806, 2019, doi: 10.1109/ACCESS.2019.2917555..

[47] Hermann, V. «Lycos annonce la fermeture de Caramail pour le 15 février.» *nextinpact.com.* 14 Janvier 2009. https://www.nextinpact.com/archive/48412-lycos-caramailfermeture. htm.

[48] Stallman, Richard. 2016. https://www.gnu.org/philosophy/open-source-misses-the-point.html.

[49] X. Zheng, R. R. Mukkamala, R. Vatrapu and J. Ordieres-Mere, "Blockchain-based Personal Health Data Sharing System Using Cloud Storage," 2018 IEEE 20th International Conference on e-Health Networking, Applications and Services (Healthcom), Ostrava, 2018, pp. 1-6, doi: 10.1109/HealthCom.2018.8531125.

[50] Karlitschek, F. 2016. http://karlitschek.de/2016/04/big-changes-i-am-leaving-owncloud-inctoday/