Self-Protecting Documents for Cloud Storage Security

Manuel Munier, Vincent Lalanne, Magali Ricarde

2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications

Self-Protecting Documents for Cloud Storage Security

Manuel Munier, Vincent Lalanne
LIUPPA
Univ Pau & Pays Adour
Mont de Marsan, France
Email: {manuel.munier, vincent.lalanne}@univ-pau.fr

Magali Ricarde
BackPlan Company
Project Communication Control
Pau, France
Email: magali.ricarde@backplan.fr

***Abstract*—Information security is currently one of the most important issues in information systems. This concerns the confidentiality of information but also its integrity and availability. The problem becomes even more difficult when several companies are working together on a project and that the various documents "go out of" their respective information systems. We propose an architecture in which the documents themselves ensure their security and thus can be exchanged over uncontrolled resources such as cloud storage or even USB flash drives. For this we encapsulate within the document itself some security components (e.g. access control, usage control) to achieve an autonomic document architecture for Enterprise DRM (E-DRM). Using such self-protecting documents, a company can ensure security and privacy for its documents when outsourcing storage services (e.g. cloud).**

***Keywords*- self-protecting document; autonomic document; usage control; E-DRM; cloud storage security;**

I. INTRODUCTION

Information security is currently one of the most important issues in information systems. Security criteria most commonly used are confidentiality (assurance that information is shared only among authorized persons or organizations), integrity (assurance that the information is authentic and complete), availability (assurance that the systems responsible for delivering, storing and processing information are accessible when needed, by those who need them) and traceability (ability to chronologically interrelate uniquely identifiable entities in a way that is verifiable). The problem becomes even more difficult if a user wants to export a document from the information system to work offline, for example, or to broadcast it to other persons outside the company. Drawing on the object-oriented approaches, we chose to encapsulate in the document itself some security components to achieve an autonomic data management architecture for Enterprise Digital Rights Management (E-DRM).

New technologies (ADSL, laptops, smartphones, tablets,...) have provided us with new technical means to communicate but have also created new needs. For example, information must be accessible at any time (possibly offline); using a centralized site for the exchange is seen as a constraint (client-server architecture vs. peer-to-peer architecture); data are stored on USB flash drives and used on untrusted computers (available on open access); we carry our data in our smartphones and share them via (possibly unsecured) wireless communications like 3G, wifi or bluetooth; cloud storage provides a platform for connected devices (laptops, smartphones, tablets,...) to access data without the need to store it locally on the device. Obviously, these new practices raise some issues related to information security.

Besides the technical aspects of these exchanges of information, their contents have changed. Data are more and more complex (notions of structured documents, whole archives, or even complete projects). Nowadays, public data are sometimes combined with more confidential data (notion of access restriction). The content provider may wish to control how the user is handling the content (concepts of usage control, DRM). This can cover both the consultation of information (access control, data presentation) and its modifications (access control, management of data consistency, logging and/or creation of metadata, update management).

To ensure information security companies typically deploy an information system responsible for providing a network storage and access (and possibly usage) controls. The implementation of such a platform, however, requires certain resources (human and technical). Companies are sometimes tempted to outsource the storage of their data: cloud, data center,... But one important consideration regarding cloud storage is security. This has also been one of the major obstacles to a larger adoption of cloud storage: clients aren't likely to entrust their data to another company without a guarantee that they'll be able to access their information whenever they want and no one else will be able to get at it. To secure data, most systems use techniques like encryption (they use a complex algorithm to encode information; to decode the encrypted files, a user needs the encryption key), authentication processes (the system requires to create a user name and password), authorization practices (the client lists the people who are authorized to access information stored on the cloud system; many corporations have multiple levels of authorization). Even with these protective measures in place, many people worry that data saved on a remote storage system is vulnerable.

978-0-7695-4745-9/12 $26.00 © 2012 IEEE
DOI 10.1109/TrustCom.2012.261

There's always the possibility that a hacker will find an electronic back door and access data. Works exist about security issues in cloud storage [1], [2], [3], [4], or more generally in cloud computing [5], [6], [7].

In this paper we propose an innovative approach that aims to make documents self-secure so they can be stored even on untrusted servers: we encapsulate within the document both the data it carries and the security mechanisms to control the use of such data. Thus security constraints can be (partially) relaxed on the server since all information is encrypted and only the security kernel of the document can access them. Using such self-protecting documents, a company can ensure security and privacy for its documents when outsourcing storage services (eg cloud).

The remainder of this paper is organized as follows: Section II presents our motivations for the design of a new E-DRM architecture based on autonomic documents; this section also presents a concrete case study that we developed with our partner company BackPlan™ [1]; in Section III we describe our autonomic document approach and we give some details about the metadata and the usage control mechanisms; Sections IV and V outline our current work about cloud storage and security issues, and then conclude this paper.

II. Motivations

In the area of collaborative work, the partners are required to consult but also to modify the documents. It is therefore necessary to implement security mechanisms that go beyond a simple access control: usage control (how partners can use a document: obligations, workflows, delegation rules,...), information consistency management (e.g. some documents may reference others), traceability (monitoring of actions, metadata attached to information),... This is the role of the Information System which is the core of the company or project.

A. Architecture

However, centralized architectures for information sharing (Fig.1) have two drawbacks compared to the use of traditional documents:

1) Work on protected documents requires the user to connect to a server responsible for enforcing the security policy. This server hosts all the documents and controls all user actions. Users can not directly exchange information: all the exchanges have to go through this server.
2) These security mechanisms require installation, on the user side, of applications and/or proprietary modules to be able to handle such documents. The user becomes dependent on the security provider for his/her applications.

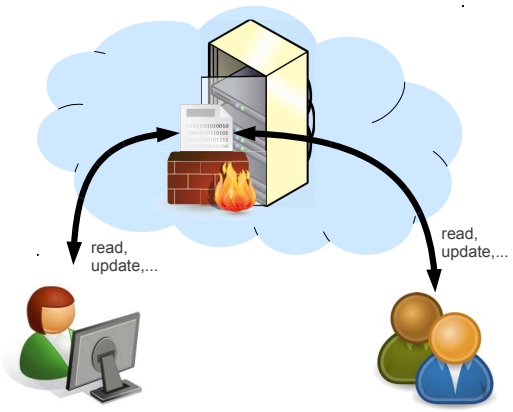

Figure 1. Document security enforced on server side

These are the two findings that led us to propose a new E-DRM architecture inspired by object-oriented concepts where the document becomes an autonomous operating entity able to control by itself how the information it contains can be accessed and used (Fig.2). Such a document is a kind of information system on its own embedding both a data warehouse and various security modules (access control, usage control, metadata,...).

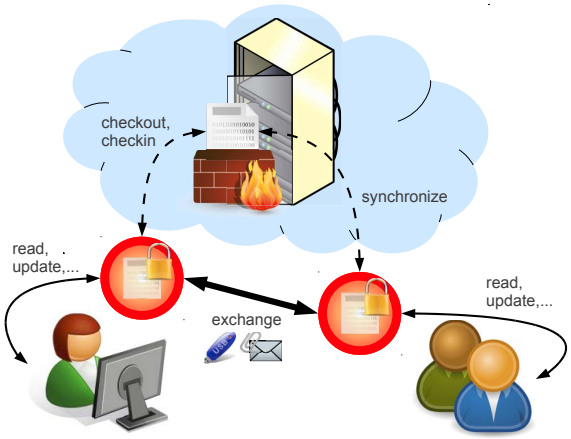

Figure 2. Autonomic document self-manages its security

This allows users to exchange such documents safely, or even portions of documents. Moreover, security constraints are relaxed on the server side since the documents themselves provide their own security. The server will only be used to synchronize versions of documents exchanged by partners.

B. Sample Application

Consider an Oil & Gas project as the construction of a pipeline or an oil installation. The information system

[1]BackPlan™ , Project Communication Control
http://www.backplan.fr

consists of numerous documents, it has a central role, its structure and development evolve along with the progress of the project: the documentation must always precede action (design, work procedures). Documentation is a requirement at the closure of the project along with verifications (records, the minutes,...). The document evolves at the same rate as the project. These documents are specifications, drawings, records of expertise, procedures, records,...

1) Business Aspects: Such a project obviously involves many partners and subcontractors. Here is a representative example of a project timeline:

- 1st level design: this step is performed by the land surveyor who makes a topographical survey of the site where the work will be done.
- 2nd level design: is the design phase of the project itself, and is performed by an engineering company that will plan the work and the construction will be launched from this plan. This level involves various partners: civil engineering, pipefitters, instrumentation engneering, utilities,... These documents are usually schemes that can have many levels or layers, each company or sector can thus share the same document. This stage ends with all engineering documents validated by the stamp "Approved for construction".
- Construction phase: it is based on engineering documents and work procedures. It comes with many documents that are intended to demonstrate compliance of the book in terms of quality and regulatory standards (e.g. multifluid standard, water code, capacity under pressure).
- At the end of construction the land surveyor will come again, and verify the topographic survey: this is a control operation of a project to verify the differences from the planned location and update the data (to know where everything is). The engineering documents then pass status "As built". This operation can also update the geographic information system (GIS) of the place.

As we have stated, the partners will handle many documents. There can also occur unforeseen circumstances during the project. Here are some identified difficulties and problems:

1) Several actors in various fields must work on the same document. Take for example the passage of a pipe under a river: the land surveyor made the survey, but it's not him who knows the techniques of drilling, centering techniques and the radii of curvature of the pipe. These characteristics are a function of pipe diameter, its alloy or its profile,... In 2nd level study partners will therefore have to refine the study of a 1st level.
2) The recordings are very numerous during the construction. For example: inspection of welds. The sections of pipe are welded every 12 to 15m. These welds should be checked: radiography, analysis by a certified individual, hydraulic tests. These controls are spread over time and generate many records that are, once made, a legal value.
3) The code requirements to be applied are numerous and specific whatever the project: standards and regulations, good practice guides, internal standards,... As part of the quality approach, journals are provided to verify that these code requirements are taken into account. It is recommended to involve the regulatory bodies in these reviews. Unfortunately, these documents are currently independent of each other (in terms of information system).
4) Consequences for digital documents: whether for office or mobile use (eg tablet or laptop on a site with no network coverage), the dependence on the document registry can be restrictive (even by a checkout of files for offline viewing). It would be more convenient to group all the information in one document, structured and secure, so that different partners can exchange it directly (and synchronize it occasionally with a server to "publish" their work).

2) Security Aspects: We propose to improve the security of information in two directions: metadata management and usage control.

Metadata: It could be used to "bind" reviews, certifications, and other minutes to design documents within the information system (see item 3 above). The aim is to improve traceability, both in the design process (concept of workflow) in case of litigation (concept of proof of conformity, digital forensics). Take for example a phase control such as checking the welding of a pipe (see item 2 above), it would be interesting to use metadata to improve the traceability of the process for purposes of validation and/or evidence: photos geotagged (to certify checkpoints), metadata associated with the plans,... Since several partners are working on such a project, everyone could also attach some metadata to the information: confidence and trustworthiness indicators, impact risk of a change,...). This metadata would permit to calculate various performance indicators for monitoring the partners' tasks or metadata to the information they produce.

Usage control: Here are some examples of security rules that we would implement to control how partners use documents:

- It is possible to write a deliverable of the project only if confidence in the various technical documents exceeds a certain threshold. It is a dynamic access control based on trust (whether in a document or a partner).
- The security rules may prohibit access to parts of the document based on location data. This prevents, for

example, on a site (or train) an unknown person takes sensitive information over the shoulder of someone. The same argument could be used for putting visual filters in place (blur, masking, wrong information,...).

- A responsive access control may require a partner (or subcontractor), via a mechanism of pre-obligation to accept the terms of a contract (non-disclosure agreement, delegation of responsibility, deadlines,...) before accessing a plan and contribute to the design.
- A user control would be to require a partner to complete parts of design documents (eg. inform the radii of curvature of the pipe, write the study of soil before drilling) before a deadline if he wants to stay a project member (notions of punishment and penalty).
- The usage control can also define collective obligations. For example, each partner must have reread each concept study in which they participate at least 7 days before the deadline for validation.

Currently, the various documents of such a project are managed independently of each other. In the best case, a document registry is set up to try to centralize critical information. The use of a true information system would allow the implementation of such a security policy. However, users now want to handle these documents on their laptops, smartphones, or tablets, and using a centralized site for the exchange is seen as an unacceptable constraint. Hence our idea of developing self secure documents that can be stored anywhere on the cloud or exchanged with USB flash drives.

C. *BackPlan*™

BackPlan™ is a French company providing document management services and collaboration workflow applications to improve project communication, transparency across the project, ability to manage schedule and risks, reliable indicators and regulatory compliance. From the engineering phases to the construction phases, projects involve different companies. All of them will use the collaboration solution BackPlan™ to ensure consistency of information across the project and a complete audit trail of project communication. BackPlan™ document management services are currently provided on a server hosted in a data center: the document registry.

However, as explained in the case study in section II-B, the solution of a centralized information system is not entirely satisfactory. Whether for office or mobile use (e.g. smartphone, tablet or laptop on a site with no network coverage), the dependence on the document registry can be restrictive (even by a checkout of files for offline viewing). It would be more convenient to group all the information in one document, structured and secure, so that different partners can exchange it directly (and synchronize it occasionally with a server to "publish" their work).

III. Autonomic Documents

Our approach is therefore to encapsulate within the document both the data it carries and the security mechanisms to control the use of such data. The architecture we propose for such autonomic documents which can manage their security autonomously is outlined in Figure 3. Here are the most significant components that will then be detailed in the following subsections:

- the database for storing the contents of the document and metadata; structured contents are represented in terms of nodes and relationships; metadata are properties attached to these nodes and relationships
- the security kernel which is responsible for enforcing the security policy and then monitors all actions on the document; it relies on various security modules dedicated to specific tasks (OrBAC for access control and usage control, metadata management, computation of indicators,...)
- embedded applications to operate on the document with dedicated tools; embedded services and/or mechanisms of export and import to handle the document using legacy applications
- the license for the user (stored outside the document) containing the permissions, prohibitions and obligations assigned to this user; later, such a license may also contain other directives for the kernel (and its modules), such as metadata to be collected or context management rules

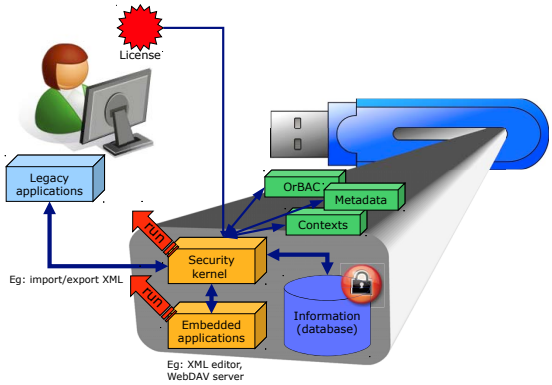

Figure 3. Autonomic document architecture

When the user wants to access the secure document he must first identify himself and provide his license which contains the security policy rules specific to that user. These "rules" may represent permissions, but also prohibitions, obligations, or metadata to be collected during the user's actions.

The document then starts the security kernel of our architecture which will in turn initiate various security modules:

a access control module, a metadata management module, a trustworthiness management module, a module dedicated to collaborative work,... The kernel and the modules are configured with the rules contained in the license provided by the user.

From that moment on, all the accesses to information within the document is made through the security kernel. For any action performed by the user, the kernel requests each security module to validate this action. Some modules will indeed confirm or deny the various actions (e.g. access control module). Others will simply add some metadata to these actions (e.g. who performed this action, from which IP, at what time, with which application, in which context,...) so that other modules can compute their data (e.g. trustworthiness indicator, collaborative work management).

The main difference to "classic" E-DRM schemes is how and where the security mechanisms are implemented. Traditionally, the document (or its subset that is accessible to the user) is encrypted using player keys; decryption takes place in a trusted viewer, which has to run securely on the client side. The trusted viewer is responsible for enforcing a license distributed alongside the document. If the user updates the document, it is generally necessary to publish the amended document on the server (Fig.1) so that another user can access and see the changes (the server must reencrypt the amended document). In the proposed architecture, the document (as an object embedding both data and security modules) is more like a decentralized part of the whole information system. It can manage the updates made by users and store these changes in its embedded encrypted database. The document can be transferred directly to another user without going through a server (Fig.2).

The second difference is that access to the document can be done either through dedicated embedded applications (like "lightweight" trusted viewers), or "heavy" trusted applications (using the API), or by exporting data (e.g. XML format) to some legacy applications.

A. Information Storage

Document data are stored in the embedded database accessible only by the security kernel. This store implements the multi-view model for secure versioned repository as defined in our previous work [8]. In this model, information is organized as a graph of which the nodes are connected by relations of various kinds. We can represent XML documents, trees like files/directories structures, project files when certain items are dependent on other elements. Each version of each node is kept with its own relationships. The view of a user is a subset of this graph computed according to his/her privileges. All user actions (accessing, updating, creating, deleting,... nodes and/or relationships) are performed with respect to his/her view. For instance, if he/she deletes a folder that contains "hidden" files (i.e. not available in his/her view), the execution of this action should result in the removal of this directory in the view of the user. This to ensure the confidentiality property for other information. Refusing the removal because "the directory is not empty" violates this property. Accepting the removal and deleting, as a side effect, an "unauthorized" file also presents a problem of confidentiality. The solution we adopted is to translate each action into elementary operations at the level of the data warehouse in terms of updates of versions of those nodes and their relationships. This is to ensure the integrity of the information. For instance, the "hidden" file that belonged to the deleted directory should not become an orphan. Since the data warehouse stores all the versions, a user who has access to all files in the previous directory could still access the file through the tree, but would also be informed that the parent directory has been "deleted" by another user (along with some files it contained).

B. Security Modules

Obviously the core of our architecture is the security kernel. It is the document interface with the outside world: all the actions performed by the user to handle the document need to be done through the security kernel (like data abstraction in object-oriented programming). From a functional point of view it has two aims: (1) translate the actions performed by the user about his/her view into basic operations on the data warehouse; (2) check that the user's actions are in accordance with the security policy.

The first task is therefore to translate user actions into basic operations at the level of versions of nodes and their relationships. We invite you to consult [8] for more details on this section. For example, to update a node, we create in fact a successor to the version contained in the view of the user. Deleting a node is equivalent to an update with the special value `NULL`. Moving a node results in the creation of a new version (link in succession) but with new relationships.

In this article we focus on the implementation of the security policy. As we stated at the beginning of Section III, the security kernel can be configured to use various security modules. For now we identified three categories of modules as they are responsible for:

- accepting or rejecting the user's actions,
- collecting and attaching metadata to the actions,
- calculating new data as actions go along.

When the user requires the execution of an action, the security kernel performs control in two stages. The first is to validate this action. For this the kernel requests each security module to validate the action. Some modules will indeed accept or reject the action (e.g. access control, usage control). Others will add information to this action (e.g. metadata). If the action is validated by all the security modules it then enters the second stage: the processing of the action. Basic operations implementing this action are then

performed on the data warehouse. Then, the security kernel broadcasts this action a second time to each security module so they can achieve the associated processing: logging (e.g. access control, usage control), computation of additional information (e.g. trustworthiness management, collaborative work management).

The first security module that we developed belongs to the first category and relates to the access and usage control to information contained in the document. From the formal point of view we are using the OrBAC model [9] which can express security policies in terms of permissions, prohibitions and obligations between a subject, an object and an action. These are the rules contained in the license provided to the user. OrBAC model also supports the notion of context [10], [11], [12]. Indeed, security rules in these policies are no longer static but dynamic, depending on the context. They must be also self-adaptive with respect to temporal conditions, the user's location, previous behavior of this user, etc... Many works attempt to extend existing models to deal with access control based on user's location context [13] or temporal context [14]. Some models are specifically dedicated to collaborative environments (in [15] users obtain permissions according to their role and the team in which they are involved) or workflow environment (in [16], [17] subjects gain access to required objects only during the execution of the task). As discussed in [11], authors use contexts in OrBAC to express different types of extra conditions that control activation of rules specified in the security policy:

- the *Temporal context* that depends on the time at which the subject is requesting for an access to the system,
- the *Spatial context* that depends on the subject location,
- the *User-declared context* that depends on the subject objective (or purpose),
- the *Prerequisite context* that depends on characteristics that join the subject, the action and the object.
- the *Provisional context* that depends on previous actions the subject has performed in the system.

Our next step was therefore to develop a second security module to manage contexts in our architecture. By adding context awareness to our intelligent document we can express contextual usage control rules in security policies. To control context activation, the embedded information system must provide the information required to check that conditions associated with the context definition are satisfied or not. To do this, either contexts have direct access to the host system (eg a global clock to check the temporal context) or they use metadata carried by the actions.

A third security module has been developed and relates precisely to metadata collection (which performs the action, at what moment, from which IP,...) and thus ensures the traceability of actions performed on the document by the different users. It is not yet configurable because we first need to define, at a license level, a language to describe the metadata logging policy. However, collecting some predefined metadata, we were able to test and validate context management and, thus, the implementation of security rules to do usage control. One of our current work is to implement a security module using these metadata to compute new information (e.g. trustworthiness evaluation for information updates as in [18], [19], [20]). Such metadata may also be useful for providing collaborative work support as discussed in [21], [22]. To do this we could either write a security module either define a (provisional) context. This remains an open question for now.

C. Opening & Using a Document

As we have previously mentioned, access to information contained in the document can then be done in three ways:

- Export and import mechanisms (XML for example) to manipulate information through existing applications. This requires setting up filters at the security kernel level to format information when exporting (check-out) and to interpret them when importing (check-in). In this case, the level of granularity is the whole file.
- Plugins developed for existing applications to have a more granular level. The plugin can then talk directly with the security kernel to interact at the nodes and relationships level, provided of course that the user has the appropriate privileges.
- Use of services and/or dedicated application embedded in the secure document. During its initialization, after starting the different security components, the document can automatically start running a WebDAV server (for example) in charge of presenting the information in the document as a tree of files/directories. Access to information can then be made from traditional applications via the WebDAV server.

D. License Contents

The last component of our architecture deals with licenses for users. As shown in Figure 4, a license is an XML document containing information of the server that issued the license (the licensor), information relating to the licensee and, of course, various security rules. Having used the OrBAC model to implement our security module in charge of access control and usage control, it is in this example OrBAC rules. Compared with policy language standards like XrML or ODRL, we also need in the future to express how metadata should be collected and what triggers must be deployed to manage contexts. Thus the license does not consist only of security rules but also contains the configuration of the various security modules required for handling the document.

To protect the contents of this license, when created, the rules are encrypted with the user's public key (being the only

one who can decrypt with his/her private key and thus the only person to know, for example, the permissions which are granted to him/her). The license is then signed by the issuer to prevent any further modification.

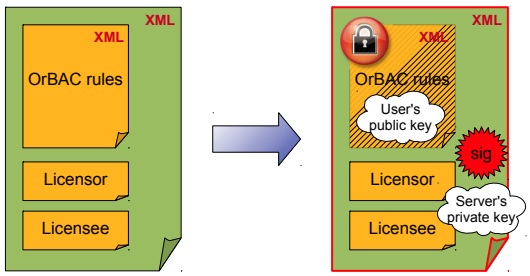

Figure 4. License contents

As we mentioned since the beginning of this article, our approach relies on the concept of metadata for traceability, context activation (dynamic security rules) and, eventually, computation of indicators (collaborative work management). Some of this metadata can however raise privacy issues: user's location, trustworthiness indicator,... One of our perspectives is to define in the license metadata to be collected and for what purposes it will be used. From a legal standpoint, the license will be a user agreement to work on the document.

IV. Cloud Storage

To build our first prototype of autonomic document based E-DRM platform we used standard USB flash drives with an auto-run configuration to launch Java programs. The flash drive represents the document and can be exchanged (physically) between different users. It brings together on the same "support" a database and several executables (security kernel, security modules, embedded services and applications). Such an architecture can run on various operating systems (Microsoft Windows, Linux,...). The only assumptions about the host computer are the availability of a Java virtual machine and enabling the auto-run for removable media. As the user can directly access files on the USB flash drive, and thus in particular to storage files of the database, we encrypted the content of the latter. Thus "raw" information of our document, which might be reached in bypassing the security kernel, is unusable.

This prototype is primarily a proof of concept. The details of "internal" mechanisms such as secret credentials embedded in the security kernel, correct execution of the kernel, ciphering used (embedded database, license contents, communications between the document and external applications,...) are beyond the scope of this paper. We obviously have implemented these mechanisms to experiment our approach, but this part of our work requires further study (e.g. using ISO/IEC 27005:2011 information security risk management approach [23]).

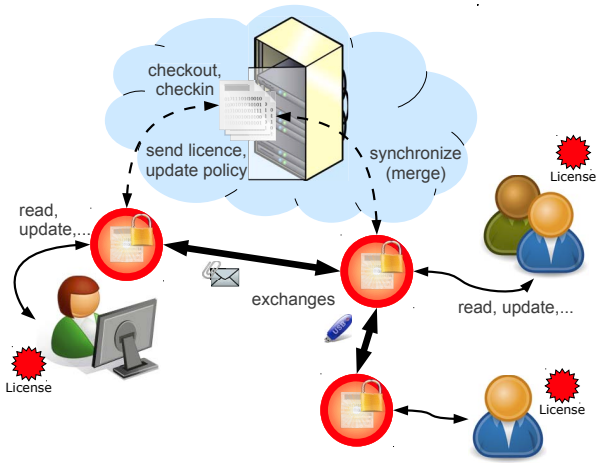

Figure 5. Self-protecting documents in cloud storage

The next step is to develop a storage server on the cloud (Figure 5). This means specifying primitives read (export of a full project document or portion of a document) and write (merging changes made by the partners). This server will also be responsible for distributing licenses and pushing any updates to the security policy. In fact, as with USB flash drives, security constraints can be (partially) relaxed on the server since all information is encrypted and only the security kernel can access them to merge the updates. Using such self-protecting documents, a company can ensure security and privacy for its documents even when outsourcing storage services (e.g. cloud).

V. Conclusions

The architecture of self-protecting documents presented in this paper is currently in the prototype stage. We were able to represent various types of information: tree of files and directories, structured documents in XML (detailed node by node), all files in a project with their dependency relationships. This allowed us to validate, at first, our data warehouse model based on a multi-view approach.

We integrated the OrBAC engine and using metadata and contexts we are currently implementing new security modules specifically dedicated to usage control (e.g. trustworthiness evaluation for information updates, traceability of changes, collaborative work management, schedule and risks management,...). For this we use existing works found in [19], [10], [24], [25], [21].

As explained in section II-B many projects involves several companies, and these companies are increasingly faced with the following dilemma: they want to strengthen controls over the use of their documents, but these documents must also be "dispersed" among users and therefore outside

the information system of the company. We are confident that the self-protecting document architecture presented in this paper is a solution to these new needs. It allows a user to forward the document to another user who handles the document according to his own license. It is no longer necessary to have a centralized server to handle static documents. We only need a drop point (e.g. outsourced cloud storage service) to synchronize the various autonomic documents.

REFERENCES

[1] X. Zhang, H.-t. Du, J.-q. Chen, Y. Lin, and L.-j. Zeng, "Ensure data security in cloud storage," *2011 International Conference on Network Computing and Information Security*, pp. 284–287, 2011.

[2] J.-J. Hwang, H.-K. Chuang, Y.-C. Hsu, and C.-H. Wu, "A business model for cloud computing based on a separate encryption and decryption service," *Information Science and Applications, International Conference on*, vol. 0, pp. 1–7, 2011.

[3] T. Mather, S. Kumaraswamy, and S. Latif, *Cloud Security and Privacy*, M. Loukides, Ed. O'Reilly, 2009, vol. 1, no. 11.

[4] J. Wu, L. Ping, X. Ge, Y. Wang, and J. Fu, "Cloud storage as the infrastructure of cloud computing," *2010 International Conference on Intelligent Computing and Cognitive Informatics*, pp. 380–383, 2010.

[5] B. Grobauer, T. Walloschek, and E. Stocker, "Understanding cloud computing vulnerabilities," *IEEE Security Privacy Magazine*, vol. 9, no. 2, pp. 50–57, 2011.

[6] K. Dahbur, B. Mohammad, and A. B. Tarakji, "A survey of risks , threats and vulnerabilities in cloud computing," *Computing*, pp. 1–6, 2011.

[7] F. Sabahi, "Cloud computing security threats and responses," *Computer Engineering*, pp. 245–249, 2011.

[8] M. Munier, "A multi-view approach for embedded information system security," in *CRiSIS*. IEEE, 2010, pp. 65–72.

[9] A. A. E. Kalam, S. Benferhat, A. Miège, R. E. Baida, F. Cuppens, C. Saurel, P. Balbiani, Y. Deswarte, and G. Trouessin, "Organization based access control," in *POLICY*. IEEE Computer Society, 2003, pp. 120–131.

[10] Y. Elrakaiby, F. Cuppens, and N. Cuppens-Boulahia, "From contextual permission to dynamic pre-obligation: An integrated approach," in *ARES*. IEEE Computer Society, 2010, pp. 70–78.

[11] F. Cuppens and N. Cuppens-Boulahia, "Modeling contextual security policies," *Int. J. Inf. Sec.*, vol. 7, no. 4, pp. 285–305, 2008.

[12] F. Cuppens and A. Miège, "Modelling contexts in the or-bac model," in *ACSAC*. IEEE Computer Society, 2003, pp. 416–427.

[13] M. L. Damiani, E. Bertino, B. Catania, and P. Perlasca, "Geo-rbac: A spatially aware rbac," *ACM Trans. Inf. Syst. Secur.*, vol. 10, no. 1, 2007.

[14] E. Bertino, P. A. Bonatti, and E. Ferrari, "Trbac: A temporal role-based access control model," *ACM Trans. Inf. Syst. Secur.*, vol. 4, no. 3, pp. 191–233, 2001.

[15] C. K. Georgiadis, I. Mavridis, G. Pangalos, and R. K. Thomas, "Flexible team-based access control using contexts," in *SACMAT*, 2001, pp. 21–27.

[16] D. G. Cholewka, R. A. Botha, and J. H. P. Eloff, "A context-sensitive access control model and prototype implementation," in *SEC*, ser. IFIP Conference Proceedings, S. Qing and J. H. P. Eloff, Eds., vol. 175. Kluwer, 2000, pp. 341–350.

[17] E. Bertino, E. Ferrari, and V. Atluri, "The specification and enforcement of authorization constraints in workflow management systems," *ACM Trans. Inf. Syst. Secur.*, vol. 2, no. 1, pp. 65–104, 1999.

[18] E. Bertino and H.-S. Lim, "Assuring data trustworthiness - concepts and research challenges," in *Secure Data Management*, ser. Lecture Notes in Computer Science, W. Jonker and M. Petkovic, Eds., vol. 6358. Springer, 2010, pp. 1–12.

[19] Xingyu Zheng and Patrick Maillé and Cam Tu Phan Le and Stephane Morucci, "Trust mechanisms for efficiency improvement in collaborative working environments," in *MASCOTS*. IEEE, 2010, pp. 465–467.

[20] C. T. P. Le, F. Cuppens, N. Cuppens-Boulahia, and P. Maillé, "Evaluating the trustworthiness of contributors in a collaborative environment," in *CollaborateCom*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, E. Bertino and J. B. D. Joshi, Eds., vol. 10. Springer, 2008, pp. 451–460.

[21] M. Munier, K. Benali, and C. Godart, "Discoo, a really distributed system for cooperation," *Networking and Information Systems*, vol. 2, no. 5-6, pp. 605–637, 1999.

[22] M. Munier, K. Benali, and C.Godart, "A transactional approach for cross-organizational cooperation," in *GlobeComm99 (Enterprise Applications and Services Symposium)*, december 1999.

[23] ISO/IEC, "ISO/IEC 27005:2011: Information technology — security techniques — information security risk management," International Organization for Standardization, Geneva, Switzerland, Published, 2011.

[24] M. Ben-Ghorbel-Talbi, F. Cuppens, N. Cuppens-Boulahia, and A. Bouhoula, "A delegation model for extended rbac," *Int. J. Inf. Sec.*, vol. 9, no. 3, pp. 209–236, 2010.

[25] J. Thomas, F. Cuppens, and N. Cuppens, "Environmental constraints management in digital right licences," in *SARSSI 2008 : 3ème Conférence sur la Sécurité des Architectures Réseaux et des Systèmes d'Information, 13-17 octobre, Loctudy, France*, Publibook, Ed. 75015 Paris: RSM - Dépt. Réseaux, Sécurité et Multimédia (Institut TELECOM ; TELECOM Bretagne), 2008, pp. 85–99.