Decentralized Access Right Management for Workflow Applications*

N. Nobelis, K. Boudaoud, M. Riveill
Laboratoire I3S
University of Nice Sophia Antipolis
Sophia Antipolis
France
{nobilis, karima, riveill}@essi.fr

P. Dubois
Certimate
1890, chemin saint bernard - p10
06220 vallauris - Sophia Antipolis
France
dubois@certimate.com

Abstract
This paper describes a way to decentralize the access control of an existing protocol for electronic data exchange. As this exchange can be considered as a Workflow, we will analyze the specificity of security in Workflow Systems.

Keywords
Workflow applications, Security policies, Decentralized Access Control, Access Right Management

1. Introduction

Network security in today’s world is based on two main approaches: a preventive one, which consists in protecting data and identity, and a detection one, which is monitoring continuously for abnormal states or behaviors. For instance, running an intrusion detection system belongs to the second family of security mechanisms whereas encrypting data or authenticating users with a password belongs to the preventive one. The main goal of the preventive approach is to ensure authentication, privacy, secrecy (through encryption) and integrity. Many different technologies have been deployed to fulfill these goals such as Certification, Public Key Infrastructure (PKI) or Digital Signature.

These security concerns occur in every distributed application or architecture, especially in Workflow Systems. A Workflow System is an information technology solution aimed at the ‘computerized facilitation or automation of a business process, in whole or part’ (Hollingsworth, 1995). In this paper, we will consider a special Workflow System instance: a Document-centric Workflow. Such a Workflow focuses on the routing of electronic documents along a pre-defined path.

The foundation of our work is an efficient protocol of non-repudiation of receipt (NRR) for electronic data exchange [1]. Although this protocol isn’t a Workflow, the desired security properties are very similar to document-centric Workflows’ security properties. The main features of this protocol are the embedding of access-right in the exchanged electronic document and the fact that access control in the core of the system is provided by a dedicated server acting as a trusted third party (TTP).

This characteristic could be inconvenient, since it presents a single point of failure at

*This work is supported by Certimate and the Region of Provence-Alpes-Côte d’Azur.
the access right management level. In this context, we propose to design a new protocol, where the access control is decentralized by suppressing the TTP.

This paper is organized as follows: First we'll describe the specificity of the security in Workflow Systems, the existing document exchange protocol (NRR) and the uses-cases that drive our work. We'll then give an overview of an existing architecture quite similar to the NRR protocol, the ‘HP’s Sticky Policies paradigm’. After that we’ll introduce our new approach, and discuss over several issues raised by our protocol. Finally, we will conclude with some remarks about future works.

2. State of the art

2.1 Security in Workflow Systems

The ISO 7498-2 standard suggests five services for information security requirements of Workflow Systems: Identification and authentication, confidentiality, non-repudiation, access control and integrity. For R. Botha and J. Eloff [6], identification (and authentication) confidentiality and non-repudiation are implemented in ways similar to non-workflow systems. The authors manage to pinpoint the principal design requirements of the access control service for Workflow Systems. Such a service must guarantee:

- **Strict least privilege**: A user should receive the smallest possible set of permissions for the current task within the business process.
- **Order of event**: Certain permissions can only be granted once other have been exercised.
- **Separation of duty**: This property has for primary objective the prevention of fraud and errors, thus ensuring the semantic integrity of business information.

The authors describe in [5] a model fulfilling all these properties named CSAC (Context-Sensitive Access Control) based on RBAC (Role-Based Access Control) model, using an Agent-based approach.

In the RBAC model, several sets are constructed (e.g. the set of users, the set of all the objects of the system, the set of actions that may be performed on objects in the system, etc...). Thus, the three fundamental properties can be expressed as functions. Numerous other writers have discussed about security in Workflow System. S. Li, D Jia, G. Zhuang and A. Kitell presented in [7] security in Transactional Workflow, where another constraint sums up: the Workflow must maintain ACID properties (atomicity, consistency, isolation and durability). The authors also presented a security model based on HTTPS to provide secure transport across firewall-protected domains.

For the next parts of this paper, we will consider that our protocol, the NRR protocol is a document-centric Workflow. However, as the main goal of our work is to propose a decentralized access control for this protocol, we won’t focus further on Workflow constraints and properties.
2.2 The NRR Protocol

The Non Repudiation of Receipt protocol defines a secure electronic transaction environment, which provides the originator of the document an electronic proof of the transaction. Thus:

1. The sender should be given the possibility to demonstrate that the recipient actually received the message or the document, even in the case where the latter denies having received it.
2. The sender should be able to give that evidence only in the situation where the recipient actually received the message or the document.

The protocol satisfies other standard properties, such as integrity, confidentiality and the sender’s authenticity. When a sender A wants to send an electronic document to a recipient B, he constructs a digital envelope containing:

- An encrypted version of the document.
- The decryption key of the document, encrypted using the public key of a Trust Third Party (TTP).
- A validation certificate proving that he is the legitimate sender of the document.
- (Optional) An embedded security policy, defining constraints for the opening of the document. An example policy may contain: ’this document can’t be opened after a given date’ or ’this document can only be opened by a member of the given domain.’

After receiving the digital envelope, B contacts the TTP. After verifying his credentials (against the policy if it’s present), the Trust Third Party decides to authorize the decryption of the document by sending the decryption key back to B. Thus, the TTP keeps a proof of B access to the document and guaranty this way the property of non repudiation.

Every document exchange protocol can be centralized or decentralized at different levels, as presented in the following table:

<table>
<thead>
<tr>
<th></th>
<th>Centralized approach</th>
<th>Decentralized approach</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rights storage</td>
<td>One or several database(s)</td>
<td>Embedment in the envelop</td>
</tr>
<tr>
<td>Policies storage</td>
<td>Idem</td>
<td>Idem</td>
</tr>
<tr>
<td>Access control</td>
<td>TTP</td>
<td>?</td>
</tr>
</tbody>
</table>

Table 1 Comparison of two approaches for the right management.

In the current state of the NRR protocol, the rights and policies are already stored in the digital envelope. Thus the main goal of our work (described in the next section), is to propose an evolution of NRR by providing decentralized access control. We will not take into account all the properties of NRR (for instance, we are going
to slightly alter the proof of receipt), however we will add new ones through our use cases.

2.3 Use cases

In this section, we would like to present some of the use cases that influenced our model’s design. Starting from the following use case, which is ‘A (Alice) wants to send an electronic document to B (Bob)’ [noted as A sends B], we have identified several variations:

- **A sends X**: A doesn’t know the recipient of the message. Thus the problem is how to define the rights that an unknown person has on the document. A possible solution could be to define access rights that don’t concern a specific user identity such as ‘this document can only be opened after the following date: day/month/year’ or ‘this document can’t be opened by a person belonging to the domain X’. We name this property **undefined document opening**.

- **A sends B sends C**: B is forwarding the message to a tiers, C. So, how are the rights propagated? Must C have the same rights as B? Can A originally define the rights at different levels of forwarding? We name this property **document tracing**.

- **A sends (C inherits from B)**: A wants to send a message to B but B has temporarily delegated its ‘authority’, by enabling C to sign on his behalf. Hence, we have to verify the validity of this delegation: has the delegation be revoked, has it expired, does B have the right to delegate? We will name this property **temporary delegation of the recipient’s rights**.

From these atomic use cases, which seem to us the most pertinent ones, we can compose more complex use cases, for instance A sends X sends B. However, satisfying all these properties in our protocol is illusory. For instance, temporary delegation of the recipient’s rights is a research domain on its own, and a lot of papers have been dedicated to this property. On the other hand, the first property is already present in the NRR protocol, and is easy to define with some policies. Since this is a Work In Progress paper, we won’t describe the implementation of these two properties in our protocol. On the other hand, the tracing of documents is particularly interesting in our situation, and we will focus on this property for now.

2.4 HP’s Sticky Policies

The Sticky Policies (SP) [3] have been developed by Hewlett-Packard (HP) Bristol to manage the disclosure of data on the Internet, during an online purchase. The main issue with online marketing is that the customer has no control over the personal data he sent:

1. Does the seller offer all the security guarantees required for the use of these data?
2. Do the seller’s third parties (suppliers, bank, etc...) offer the same guarantees?
3. Is the client able to track his personal data?
The reader will notice that two latter issues are also present in the secure electronic document exchange: these are *document forwarding* and *tracking* issues.

To answer these problems, the authors have developed the Sticky Policies architecture. In this model, the customer attaches a set of security policies with his sensitive personal data. This policies may contain:

- Conditions to fulfill in order to access the data.
- Actions to run when an access to the data is carried out. For instance notify the customer that his personal information are accessed.
- Lifetime of these policies.

The conditions to access the data could be constraints (such as ’the sellers agree not to transmit the data to a third party’) but could also define restricted access to the sensitive information. For instance, if the personal data are a credit card and a cell phone number, the customer is allowed to define a policy specifying ’If the receiver is French, he will be able to access the cell phone number; otherwise he won’t be granted access to the phone number’. Thus, the personal data are fragmented in order to provide only the necessary information: this is called ’self profiling’.

Similarly with the NRR protocol, the Sticky policies are embedded along the personal data inside a digital envelop. When the seller receives the envelop, he presents his credentials to one or several Trusted Third Party (TTP). The TTP will validate the credentials against the embedded policies, but he also has the possibility to perform several others checks, for instance ’verifying if the seller platform (software and hardware) honors all the criteria required for a guarantee of safety for online marketing’. This control is accomplished via the Trusted Computing Platform Alliance (TCPA) specification.

Thus, the customer has the possibility to track his personal data: through the history of the TTP access, he will be able to determine if an untrustworthy seller transmits the sensitive information after having received the decryption key from the TTP.

After this presentation of both the NRR protocol and the Sticky Policies architecture, we can see several similarities between the two models:

- Both attach a security policy to the sensitive piece of data (independently of its type, an electronic document or a set of personal informations).
- The same properties are guaranteed (confidentiality, integrity, and proof of receipt).
- The access control of both models are performed by Trusted Third Party.

However, the models have several differences: The Sticky Policies architecture has been motivated by different use cases, in particular self profiling and forwarding. On the other hand, NRR focus more on the legal aspect of the exchange, i.e. ’how to construct a legal proof of receipt?’.

We are now going to present our model: the decentralization of the NRR protocol.
3. Our model

3.1 Access rights management

Since our objective is to decentralize access control for the NRR protocol, we need to integrate the access rights within the document. In order to manage the rights, many systems use Lampson’s access matrix [2] (see Table 2): the column represents the different objects of the system whereas the row represents the different users.

At the intersection of a user U and an object O (a cell) is stored the access rights of U on the resource O. This matrix may be stored in the system by rows, named capability, or by columns which represent the access control list (ACL). A user holds its capability, i.e. the lists of resources with their respective rights while an ACL is stored with the associated resource.

<table>
<thead>
<tr>
<th>Object 1 (a text file)</th>
<th>Object 2 (a hardware resource)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alice</td>
<td>READ/WRITE/DELETE READ EXCLUSIVELY/HALT</td>
</tr>
<tr>
<td>Bob</td>
<td>READ/WRITE INCREMENTAL Ø</td>
</tr>
</tbody>
</table>

Table 2 An example of the Lampson’s access matrix.

In this Lampson’s access matrix, the first column is the ACL associated with the Object 1, whereas the first row represents Alice’s capabilities.

Such a matrix is similar to the previously mentionned Role-Based Access Control (RBAC) model where several sets are constructed (e.g. the set of users, the set of all the objects of the system, the set of actions that may be performed on objects in the system, etc...).

To fulfill the need of embedding access rights into an electronic document, we have first thought about using ACLs. However, ACLs don’t allow the definition of complex access rights as defined in our use cases. Thus, instead of using ACLs, we propose to use access control policies (ACP), which are already in use in the NRR protocol implementation.

3.2 Decentralized access Control

In this section, we are going to propose a secure document exchange protocol with decentralized access control. We design our protocol with two properties in mind:

- The first one is the proof of receipt. Since the NRR protocol is the base of our work, this property is easily obtained.
- The second one is issued from our use cases. With the ‘document tracing’ property, the sender of the document should be able to track his document, through the various forwarding.

As in the NRR protocol, the access rights are defined through an Access Control Policies (ACP) and are embedded, with the document, in a digital envelope.
Let’s take an example by considering the following case: Alice (A) wants to send Bob (B) a document (D). Alice constructs an envelope, which contains the document and an ACP. Considering R as ACP and K as the encryption key, the first version of the envelope is:

\[ R \quad [D]_K \]

**Figure 1:** The first version of the envelope.

In this version, the security of the document is ensured, since it’s encrypted. However, several problems appear: the ACP isn’t encrypted and the key K isn’t available for the receiver B. Thus, to resolve these problems we propose a second version of the envelope, where we introduce another encryption key K’ that will be used to encrypt the ACP and the decryption key of the document D:

\[ [R]_{K'} \quad [K']_{K'} \quad [D]_K \]

**Figure 2:** The second version of the envelope.

Having encrypted the ACP, the next questions that still not resolved are: How to access to the ACP? Who has the right to access it? To answer these questions, we add a third party, called Access Control Manager, to manage the rights (see Figure 3):

**Figure 3:** First version of the proposed protocol.

This third party (named Access Control Manager on the schema) is located on B’s computer: it is a manager responsible for controlling access rights of B to received documents. This manager decrypts first the ACP using its private key K’. Then, it asks the credentials of B (certificate, key, password). Finally, if B is authorized to access the document, the manager notifies A from the reception of the document and delivers the
key $K$, allowing $B$ to decrypt the document $D$.

The presence of this third party introduces new issues: How can $A$ access the public key of the manager (needed to encrypt the ACP)? A possible solution would be to assume that all managers of the system have the same private/public key couple $K'$. This solution is not appropriate for two reasons. Firstly, if we need to update the managers’ keys, this should be done at the same time on all managers. Secondly, if the keys are broken, by brute force of by collision, this will compromise the security of the whole system.

There is also a slight flaw in the current envelope: since the ACP $R$ is encrypted using the public key of the manager, it’s possible for $B$ to intercept the envelope, forge a fake ACP and insert it in the envelope. To prevent this, we include a digital signature in the envelope. This signature, made by $A$, uses a secure hash function $h$ to assert the integrity of the envelope:

\[
[R]_{K'} \quad [K]_{K'} \quad [D]_{K} \quad \text{Sign}_A(h(R)_{K'}, [K]_{K'}, h([D]_{K}))
\]

**Figure 4:** The third version of the envelope.

Another design flaw lies in the fact that the manager is located on $B$’s computer, it is possible for $B$ to dump the memory of his system, to gain access to the decrypted ACP or the manager private key. This issue could be prevented by using protected memory for the manager, but not every operating system implements this feature.

To address all these issues, we have designed a second version of this protocol. In this version, the Access Control Manager is located on $A$’s computer, as defined in Figure 5:

**Figure 5:** Second version of the proposed protocol.

In this second version, $A$ still sends the envelope (containing the ACP and the encrypted document) to $B$. Then $B$ sends its credentials to the ACM (he doesn’t need to resend the ACPs since the ACM is locate on $A$ computer). After the reception of $B$’s credentials the ACM takes its decision: if $B$ is authorized to access the document, the manager notifies $A$ from the reception of the document and delivers the decryption key $K$ to $B$. 

8
This model solves all the previous version’s problems, although it raises two new issues:

• A’s computer must be running and available or B won’t be able to contact the ACM.
• A is protected since the ACM can provide a proof that B has received the envelope and asked for the description key. However, B isn’t protected from A. For instance if the ACM doesn’t send B the decryption key, B won’t be able to prove its legitimacy. Therefore, we are losing an important property of NRR, i.e. the proof that the sender and the recipient were truthful. This property can’t be obtained with the localization of the ACM either on A, or on B.

Besides the specific issues resulting from the introduction of the access control manager, the main characteristics of our protocol are: 1) integrity and authenticity of the envelope and 2) confidentiality of ACP and document. However, the ‘document tracing’ property is quite difficult to obtain. Let’s take an example: if A sends a document to B who forwards it to C. C must notify A, so A could continue to track the document. Unfortunately, C doesn’t know A. He must, in a first time, send his notification to B, who would transmit it to A. Thus, the last recipient of the document must send his notification through the chain of all the recipients of the document. This lead to a dramatic increase of the failure points...

After analysis, we conclude that this property is also impossible to obtain if the Access Control Manager model is located on the sender or the recipient of the document. Another solution would be to host our Access Control Manager on multiple nodes of a peer-to-peer network. Each peer of such network could be an Access Control Server for the communication of other peers. This architecture will be the subject of future works.

4. Conclusion
In this work in progress paper, we have introduced our protocol, which aims to ensure a decentralized access control for electronic date exchange.

The solution we proposed for access control isn’t satisfying because we can’t ensure the tracing of documents and the proof of receipt property. For future works, we plan to work over new use cases, especially the temporary access rights delegation, but we are also looking forward new technologies to perform the decentralized access control, in particular the peer-to-peer ones. Although our rights are defined as Access Control Policies, we are also thinking about using a trust-management based approach [4].

Acknowledgment
A special thanks to Florian Doyon for his support during the writing of this paper.
References


