Deploying security policy in intra and inter Workflow Management Systems

Samiha Ayed, Nora Cuppens-Boulahia and Frederic Cuppens

TELECOM Bretagne
2 rue de la chataigneraie, Cesson Sévigné 35576
Email: {samiha.ayed, nora.cuppens and frederic.cuppens}@telecom-bretagne.eu

Abstract—Workflows are operational business processes. Workflow Management Systems (WFMS) are concerned with the control and coordination of these workflows. Workflow technology has typically been deployed in domains with processes that have obvious coordinating requirements. As such, the flow of control and data offered by this technology is easily mapped to process effectiveness. With the diversity of resources, subjects and activities in the system, ensuring a secure execution environment of the workflow becomes a critical issue. In this paper, we are interested on deploying a WFMS security policy. We study, once the workflow specification and its security policy are ready, approaches to deploy this security policy on an information system. This is can be done either within an intra-organizational or an inter-organizational workflow. The later case is more complex and requires more control since it includes communications between different organizations which must be controlled.

Keywords: WFMS, OrBAC, Intra-organizational, Inter-organizational, Security Policy.

I. INTRODUCTION

A workflow is a process consisting of several tasks to be executed by respecting specific constraints. WFMS are based on representing processes as workflows. A workflow representation implies that tasks composing it are interdependent and are communicating control information and data to each other. For example, let us consider a workflow composed of tasks $T_1$, $T_2$ and $T_3$ which must be executed in a sequential order. If we suppose that these three tasks act on the same documents, the access to these documents must be controlled according to the execution order of tasks. In other words, this access control must be synchronized with progression of the workflow execution. In addition, the execution of a task is related to the execution of precedent tasks. So an access control model is needed. On the other hand, workflow tasks can be executed within the same organization or within different ones. In the first case, the workflow security is controlled by the same organization which has the authority to manage the security of its different roles and their interactions and to control the access to its different objects. In the second case, a more rigorous treatment must be done to ensure a secure execution of the workflow. So, security requirements are not limited to access control which must be applied but also it concerns the need to apply a flow control to manage interactions between different components of the WFMS. If a piece of information Info flows from a role $R_1$ to another role $R_2$ of the same organization or a different one, we must be sure that $R_2$ has privilege allowing it to get an access to Info. If $R_2$ must not access to Info and $R_1$ transmits the information Info to $R_2$, a leakage of information occurs. The problem is known as the "confinement problem" [10], [9] and requires a flow control model to solve it. So, a workflow specification must be correlatively defined with a security policy which takes into account access control and information flow control simultaneously. Since a workflow execution implies interaction between different entities, how to deploy a security policy becomes a critical issue. Either with a centralized or a distributed execution of the workflow, we have to define different responsibilities of different workflow entities to enforce the whole security of the workflow execution.

Many research works have been interested in studying the security in WFMS but no study has addressed the deployment part. Several proposals [12], [14], [15], [13], [11] suggest different access control models to manage security in WFMS without defining how to implement such a policy. They have only defined a centralized procedure that controls the execution of the workflow. [14], [12] propose a conceptual and a logic model WAM (Workflow Authorization Model) to enforce the authorization flow based on the inter-dependencies between tasks using coloured and temporized petri nets. Hung and Karlapalem [13] have presented an authorization model for WFMS. This model addresses three security properties: integrity, authorization and availability. [11] studies authorizations in WFMS considering different constraints. In this work, Bertino describes the specification and enforcement of authorization constraints in workflow management systems. This work enumerates and defines all possible configurations that satisfy the workflow accomplishment and all constraints. But it does not show how a manager must act to deploy these assignments within its workflow without violating its constraints. Also, it does not consider the inter-organizational case where many organizations can interoperate to achieve a common objective. These works suppose the existence of a central entity responsible for specifying and managing the workflow security policy without showing how such a policy will be used. They do not deal with the communications required to apply the security policy with the workflow. Also, they do not show how this security policy can be dynamically supervised during the workflow execution.

In this paper, using an access and flow control based on OrBAC model [1], we present an approach to deploy a security
policy either in intra-organizational or in inter-organizational workflows. This works is more focusing on the manner which must be used to manage the workflow security policy. It response to questions: Who defines the workflow security policy? Who is responsible for maintaining a secure execution environment? Who has permission to manage the workflow security policy? The paper introduces the PEP and the PDP entities required to deploy the security policy within the workflow. We show how these entities have to communicate with workflow components in order to synchronize and to manage the security policy. In [6] a workflow security policy is modeled and specified to be used in association with the workflow specification. Once such specification is ready to be implemented, we show in this paper how to deploy it and how it must be applied to different components of the workflow. This deployment depends necessarily on the execution mode of the workflow and especially on the type of the workflow (intra or inter organizational workflow).

The paper is organized as follows. Section 2 introduces the workflow management systems in their execution modes and their types (intra or inter organizational workflows). Section 3 presents approaches (provisioning and outsourcing) to manage security policy within a workflow. Section 4 explains how to define the security policy associated with WFMS and how to use the OrBAC model for this purpose. Section 5 specifies our approach to deploy a security policy either in intra-organizational or in an inter-organizational workflow. Finally, section 6 concludes the paper and outlines future work.

II. WORKFLOW MANAGEMENT SYSTEMS

A workflow is a representation of a process. This process is divided into many tasks having an inter-dependent execution. The execution of these different tasks may include temporal constraints or conditions. Actually, we can suppose different manners to constrain tasks execution: two tasks may be executed in sequential, parallel or concurrent mode. A workflow is composed of two phases: a building phase and an execution phase. The first phase defines the formal representation of the process. The second one is an instantiation of the workflow which is based on the specification defined in the first phase.

A WFMS is a system which supports the specification, control, coordination and administration of processes using workflows. This is done through the execution of a software which is based on the workflow structure. These systems are used in different domains: research, industry, commerce, etc. These systems may include many subjects or roles and many resources when managing and executing different workflows. This introduces the need to care about the security in these systems.

Tasks of a workflow may be executed by the same subject or by different ones having different roles and must access to different resources of the system. So, to ensure a secure execution, tasks must be executed by authorized persons or subjects and according to their execution order specification. In addition, subjects must have access only to objects available and authorized. This access must be limited to the period of task execution. Thus, granting or revoking privileges must be synchronized with the workflow execution progress. All these requirements lead us to combine the conception of the workflow and the security policy associated with it. Since security is essential and represents an integral part of a workflow, this security policy must ensure many properties: integrity, authorization, availability, confidentiality, authentication and separation of duty.

A. Centralized and distributed workflows

A WFMS can be executed either in a centralized or a distributed mode. With a centralized control, there exists a single WFMS which is a central entity responsible for (1) distributing the tasks to the appropriate agents, and (2) ensuring the specified task dependencies by sending the tasks to their respective agents only when all requisite conditions are satisfied. This entity has a whole view of the workflow. It is the initiator of the workflow and it evaluates execution results to be able to choose the next agent to whom it will send the next task. In this mode, workflow agents do not know the different dependencies between each others.

Contrary to the centralized mode, in the distributed execution mode the whole workflow is sent by the initiator to the first agent $A_1$. After the execution of the first task associated to the agent $A_1$, $A_1$ evaluates the dependencies with its successors. The rest of the workflow is then sent to the valid successor depending on the result of the execution of the first task. Finally the last agent sends the final result to the central entity which initiated the workflow. In this mode, workflow agents know different dependencies between others. Especially, each agent knows its successors and conditions that are needed to transit from a task to another. Also different constraints defined with the workflows specification are ensured by different agents which have to check if they are respecting workflow constraints when executing different tasks. These two modes are represented in figure 1.

B. Intra and inter organizational workflows

Workflow tasks can be executed within the same organization or within different ones. In the first case we deal with a classical case of an intra-organizational workflow and in the

![Fig. 1. Centralized and distributed workflow](image)
second case we deal with an inter-organizational workflow. Intra-organizational workflows are simpler to manage since all process requirements (resources, subjects...) belong to the same organization. In the inter-organizational case, different requirements and new requirements are needed. Figure 2 represents an inter-organizational workflow. Tasks of this workflow belong to three different organizations (ORGA, ORGB and ORGC). These tasks can belong or not at the same time to other local workflows in their own organizations. On the one hand and as a functional perspective, the execution of an inter-organizational workflow needs a synchronization between the execution of different sub-workflows that participate in the collaboration. On the other hand and as a security perspective, in addition to the interactions within different organizations, we have to deal with interactions between these different organizations. The execution of an intra-organizational workflow can be either centralized or distributed as it is explained in section II-A. The execution of inter-organizational workflows can be considered distributed since each sub-workflow is executed within an independent organization. But the coordination between these different organizations can be done either by these organizations themselves (distributed mode) or through a central entity which deals with the synchronization and the orchestration of different sub-workflows (centralized mode).

III. APPROACHES OF WFMS SECURITY POLICY MANAGEMENT

Functionally, the central entity of a workflow can be either a coordinator entity which controls the whole execution of the workflow in a centralized workflow or just an initiator entity in a distributed workflow. This entity will be also responsible for the security of the workflow. For this aspect, a choice must be done by the workflow administrator. Indeed, the security requirements have to be managed either using an outsourcing approach or a provisioning approach.

A. Outsourcing approach

With the outsourcing approach, the workflow administrator chooses that the central entity of the workflow is also responsible for the security of the workflow. Figure 3 represents this approach within a centralized and distributed workflow. Arrows in this figure show different communications during the workflow execution. We notice that access to the security policy is only done by the WFC. Using this approach, workflow agents have to ask for permissions to execute workflow tasks from the central entity. This central entity have to check and verify if the agent is authorized to execute the task depending on the workflow security policy and the workflow specification which includes workflow constraints. Even if the execution of the workflow is distributed, the central entity will need all information required about the execution progression when it is asked for a permission from workflow agents. The security policy is managed on the central entity. The different workflow agents do not have any knowledge about it. They have just the obligation to ask for a permission to execute requested workflow tasks. The central entity synchronizes this security policy with the execution progression of the workflow.

B. Provisioning approach

In the provisioning approach, by contrast to the outsourcing approach, the central entity delegates the security requirements of the workflow to different agents. So, if an agent needs to execute a workflow task it will not ask permission to the central entity but it will search it from a security server. This check is done by all workflow agents. This approach reduces the bulk on the central entity but it increases the execution time on the workflow agents. Figure 4 represents this approach with a centralized and a distributed workflow. Arrows in this figure show different communications during the workflow execution. We notice that access to the security policy is done by workflow agents. With this approach the central entity has just functional functionalities. It does not deal with the security part which is managed by agents.
IV. Definition of the WFMS Security Policy

A. Workflow constraints

A workflow specification is correlatively related to a set of constraints which complete the workflow definition. These constraints can concern either workflow tasks or subjects or roles that execute these tasks. We classify workflow constraints into two classes: static constraints and dynamic constraints.

a) Static constraints: This type of constraints can be evaluated during the user role assignment time. This type includes the separation and the binding of duties principles.

Definition 1: SC is the set of static constraints of a workflow. SC is composed of atoms and rules. If Ti, Tj and Tk are workflow tasks and Si a subject belonging to the workflow and ci is a workflow constraint then:

\[ c_i \in SC \Rightarrow c_i \in \{ \text{same_subject}(T_i, T_j), \text{different_subject}(T_i, T_j), \text{must_execute}(S_i, T_j), \text{execute}(S_i, T_j) \rightarrow \neg \text{execute}(S_i, T_k) \} \]

An example of a set of these constraints is given below:
- Task T must be done by subject Sj
- Tasks Ti and Tj must be executed by the same subject
- Tasks Ti and Tj must be executed by different subjects
- If Si executes the task Ti then it must not execute the task Tj

b) Dynamic constraints: Dynamic constraints have to be evaluated during the execution time. Once the workflow execution starts, the workflow manager have to survey it in order to control the consistency of dynamic constraints. This type of constraints includes essentially temporal constraints.

Definition 2: TC is the set of temporal constraints of a workflow. TC is composed of atoms and rules. If Ti, Tj and Tk are workflow tasks and ci is a workflow constraint then:

\[ c_i \in TC \Rightarrow c_i \in \{ \text{Start_Before}(T_i, T_j), \text{Start_With}(T_i, T_j), \text{End_Before}(T_i, T_j), \text{End_With}(T_i, T_j) \} \]

An example of these constraints is:
- Task Ti must start before the task Tj
- Tasks Ti and Tj must end at the same time
- Tasks Ti must start and end before task Tj
- Task Ti must start before Tj and must end at the same time with Tk

Another type of these constraints results from the workflow specification itself. A conditional execution in the workflow specification is interpreted as a dynamic constraint. An example of these constraints is: If Ti have been executed then execute Tj otherwise, execute Tk.

B. OrBAC in brief

To define a WFMS policy and its secure execution environment, we suggest using the OrBAC model and its concepts. Thus, before presenting how to manage the security policy associated with the workflow execution, we first briefly recall basic concepts suggested in the OrBAC model.

In order to specify a security policy, the OrBAC model [1], [4] defines several entities and relationships. It first introduces the concept of organization which is central in OrBAC. An organization is any active entity that is responsible for managing a security policy. Each organization can define its proper policy using OrBAC. Then, instead of modeling the policy by using the concrete implementation-related concepts of subject, action and object, the OrBAC model suggests reasoning with the roles that subjects, actions or objects are assigned to in an organization. The role of a subject is simply called a role as in the RBAC model. The role of an action is called activity and the role of an object is called view. Each organization can then define security rules which specify that some roles are permitted or prohibited to carry out some activities on some views. Moreover, an organization can be composed of several sub organizations, each one having its own policy. It is also possible to define a generic security policy in the root organization. Its sub organizations will inherit from its security policy. Also, they can add or delete some rules and so, define their proper policy. The definition of an organization and the hierarchy of its sub organizations facilitate the administration [3]. The security rules do not apply statically but their activation may depend on contextual conditions [2]. For this purpose, the concept of context is explicitly included in OrBAC. Contexts are used to express different types of extra conditions or constraints that control activation of rules expressed in the access control policy. So, using a formalism based on first order logic, security rules are modeled using a 6-places predicate.

Definition 3: an OrBAC security rule is defined as: security_rule (type, organization, role, activity, view, context) where type \( \in \{ \text{permission, prohibition, obligation} \} \). An example of this security rule can be: security_rule (permission, a_hosp, nurse, consult, medical_record, urgency) meaning that, in organization a_hosp, a nurse is permitted to consult a medical record in the context of urgency.

From an abstract policy, we can then derive a concrete policy that applies to subject, action and object using the following OrBAC predicates:
• empower(org, subject, role): means that in organization org, subject is assigned to role.
• consider(org, action, activity): means that in organization org, action is considered as an implementation of activity.
• use(org, object, view): means that in organization org, object is used in view.

C. WFMS policy specification

To specify the workflow security policy we are using the API OrBAC. It specifies the workflow security policy using the OrBAC model. This policy is saved as XML RDF files. Using constraints defined with the workflow specification, the security server defines the different contexts of security rules. These contexts make the security policy dynamic and synchronized with the workflow execution. Workflow constraints are either static or dynamic as shown in section IV-A. In OrBAC model there are built-in contexts that can take into account the definition of such constraints. Prerequisite context aims to restrict or extend privileges granted to a role depending on some conditions. So, this context category is useful in WFMS to specify constraints associated with the workflow process execution. The provisional context depends on previous actions the subject has performed in the system. In other words, it corresponds to a history of execution. Provisional contexts are very interesting in the domain of WFMS since the execution of a task depends on the history of execution of precedent tasks. Also, it permits the definition of a dynamic security policy according to contexts, a very useful requirement in WFMS. Using the workflow specification the policy server specifies an abstract policy of the workflow. Such a policy is based on roles, activities, views and contexts. Until now, the policy do not deal with the instantiation of the workflow. This policy is defined within the organization where the workflow process will be executed.

V. Deploying WFMS security policy in distributed workflows

With the diversity of resources, roles and tasks in a workflow, the deployment of a WFMS security policy becomes a critical issue. In this section we show how to manage such a security policy either in an intra-organizational workflow or in an inter-organizational workflow. The second case is more complex since it deals with different communications between organizations cooperating to execute the global workflow. To specify our security policy, we use the OrBAC model. As shown above, this model is able to represent confinement aspect using organization notion.

A. Intra-organizational case

The deployment of the security policy depends on the execution mode of the workflow and on the approach that we are using to manage the policy. For the intra-organizational case, we suppose that we deal with a distributed workflow. In a centralized workflow, all communications in the workflow are done through the workflow coordinator (WFC). Figure 5 represents a general scheme of deploying a WFMS security policy using an outsourcing approach. The explanation in this section follows the numbers on the arrows in the figure.

1) Specification environment: As shown in 5, the general scheme is divided into a specification environment and an enactment environment.

(1) The definition of the workflow specification takes place. The process designer provides a written or a graphic specification of the different tasks of the workflow and of the roles that have to execute these tasks. Also, he defines different constraints associated with the workflow definition.

(2) Different specification components are transmitted to a modeling tool. Petri nets are an efficient tool to model workflows. Indeed, aside representing the workflow evolution, Petri nets are able to take into account temporal constraints that can be defined with the workflow. With Petri nets, it is possible to specify that the execution of activities must be done in sequence, or in parallel, or that a choice has to be made between activities (alternative activities), or that certain activities need to be executed more than once (iterative activities). An approach using petri nets to represent workflows is explained in [6].

(3) Once the workflow is modeled, the process modeling tool communicates with a process verification engine to check if the workflow is well formed. If the workflow is modeled using Petri nets, there are Petri nets properties (correctness, liveness, well-structuredness, soundness, ...) that must hold to conclude that the workflow is well defined and specified.

(4) Once the workflow is well formed, workflow constraints are checked. Constraints validation engine checks the consistency of constraints defined with the workflow specification.

(5) The policy server has to get information about the workflow specification, so that the security policy can be specified. Thus, the process modeling tool transmits workflow information (roles, tasks, objects, constraints) to the server policy. Based on these elements, the policy server specifies the workflow security policy to be used to ensure a secured execution of the workflow.

(6) The API OrBAC creates the workflow security policy and saves it into a policy repository (PR). This policy is an abstract policy which is based on abstract entities defined in the OrBAC model. From this abstract policy saved in the policy repository, a concrete policy will be derived when the workflow is instantiated. This policy is defined with non active contexts. These contexts contain initial conditions and constraints defined with the workflow specification. With the progression of the workflow execution, these contexts are updated and are activated in order to make the security rules applicable. The process modeling tool provides the workflow specification to the process enactment engine. Then we move to the enactment environment.

2) Enactment environment: The enactment environment presents the instantiation of the workflow. In this environment the workflow specification passes to the run time phase. The user assignment is done at this phase depending on the workflow specification. The steps below define the next phase to be done once the workflow specification is ready. In the
case that we present, we are using a decentralized approach to execute the workflow. Regarding the security part, an outsourcing approach is used. So, the WFC is responsible for the security of the workflow execution. To manage the security policy, the WFC needs a PEP (Policy Enforcement Point) in order to deal with different policy instances. This logical entity is responsible for controlling access to different system resources. Before starting the workflow execution, the WFC has to know how to manage the workflow security policy. So, communication with the policy server takes place. In the side of the policy server, there is the PDP (Policy Decision Point) who is responsible for communicating with the security client and for replying to its different requests. PDP is a component of policy-based management. When a user tries to access a file or other resource on a computer network or server that uses policy-based access management, the (PEP) will describe the user’s attributes to other entities on the system. The PEP will give the PDP the job of deciding whether or not to authorize the user based on the description of the user’s attributes.

(7) In the process enactment engine, the WFC (Workflow Coordinator) entity deals firstly with the assignment of users to different roles and tasks. This assignment is based on a communication with different users and external applications that can participate to accomplish the workflow execution.

(8) The assignment made by the WFC must be checked with the workflow constraints. Such an assignment must not break the consistency of the constraint base. So the process enactment engine has to communicate with the constraints validation engine to validate its assignment and to apply it.

(9) The PEP sends a request to the PDP. The request must contain information about the assignment of users to different tasks and the different contexts of the workflow execution. Then, the PDP receives the request. This logical entity or place on a server makes admission control and policy decisions in response to a request from a user wanting to access system resources.

(10) After receiving concrete entities and information about the workflow execution, the PDP communicates with the PIE (Policy Instantiation Engine) in order to ask for the concrete policy.

(11) The PIE is responsible for generating a concrete security policy based on information received from the PDP and on the abstract policy saved in the policy repository. The PIE updates different contexts and transfers the new security policy to the PDP.

(12) The PDP takes the policy decision. Whenever it receives a new policy instance, that is a concrete rule (permission or prohibition), a PDP has to map this information onto concrete actions to push the new policy into the PEPs. A PDP thus has to be aware of its PEPs capabilities, so that it can translate the security rules into generic configurations, considering the kind of PEP, and then the generic configurations into specific configurations, considering the implementation of the PEP. So, the PDP translates the concrete policy into a translatable language and it stores it into a PIB (Policy Information Base). Applicable policies are stored on the system and are analyzed by the PDP.

(13) The PDP makes and returns the decision. The PEP will let the user know whether or not he has been authorized to access the requested resource.

(14) The PEP sends an acknowledgment to the PDP to inform about its acceptance of the PDP decision and how it will apply it.

(15) Once the workflow specification and a consistent policy has been defined, the WFC initiates the workflow. It sends the whole specification to the first agent that will execute the first task of the workflow. Since this workflow execution is distributed, each agent responsible for executing a task, sends a request to the WFC in order to grant him the authorization to accomplish the task. These requests are represented by blue arrows in figure 5. When the WFC receives such a request, it takes the decision to accept the accesses. Then, it sends a response to the workflow agent including the permission or the prohibition to execute the workflow task. After the first task, agents have to send the execution context to the WFC. This context provides information about the progression of the workflow execution. The WFC needs such information to be able to update its policy. In fact, when a request is received from a workflow agent, the WFC has to communicate with the PDP and steps 9-14 are executed for each communication between the WFC and workflow agents. The execution context especially indicates about previous tasks that have been accomplished. If we consider for example a workflow where a task T_4 can be executed only if task T_1 has been executed successfully and then tasks T_2 and T_3 have been executed in parallel and successfully, then the authorization is granted to agent executing T_4 only if the execution context holds and the WFC is informed about this.

Steps (16) and (17) show the functional communication between workflow agents. Blue arrows show the requests for the authorization rules from the WFC.

(18) The last agent sends the workflow result to the WFC, and so the workflow execution completes.

If we consider that the execution of the workflow is centralized, the communication between WFC and workflow agents to ask for authorizations will not be present. In fact, the WFC will send the task and the authorization associated with the task execution to the workflow agent and the agent will respond with the execution result. By receiving this result the WFC will update its security policy and select the next agent depending on the result of the task execution that he has received.

If we suppose that the security policy is managed using an provisioning approach, then the WFC will have no idea about the security policy to be deployed during the workflow execution. Also, workflow agents will communicate directly with the policy server to ask for an authorization to execute their tasks. Thus, each workflow agent will have a Local PEP (LPEP) and these LPEPs will communicate with the PDP present on the policy server. In addition to information about their capabilities, LPEP have to communicate to the PDP...
inter-organizational communications. To create a VO, the inter-
workflow constraints. The VO is responsible for the secure
enforce constraints.

T task that task

IV-A are defined within an inter-organizational workflow and
suppose for example that constraints of the examples of section
violates the base of constraints or not. Otherwise, if we
the WFC can enforce and check if its user role assignment
is provided by an organization OrgB, enforcing workflow
security policy must be consistent with the different local
different subjects from an organization to another. The VO
be controlled by this policy. Most workflow constraints are
be done in order to ensure these different communications and flows that
can communicate between each other, workflow organizations have to
be controlled by this policy. Most workflow constraints are
based on tasks of the workflow and subjects executing these
tasks. So, the VO has to mainly manage the transition of
organizational policy of the global workflow. To com-
different organizations continue until obtaining a consistent
inter-organizational constraints base.

2) VO security policy: Based on the different constraints
that the IWFC gathers, it has to define a coordination security
policy and to administrate it in order to ensure the non
violation of the constraints during the workflow execution. To
define its policy, the IWFC must transform the constraints base
into a set of security rules. These security rules constitute the
inter-organizational policy of the global workflow. To com-
municate between each other, workflow organizations have to
be controlled by this policy. Most workflow constraints are
based on tasks of the workflow and subjects executing these
tasks. So, the VO has to mainly manage the transition of
different subjects from an organization to another. The VO
security policy must be consistent with the different local
security policies of different organizations. Also, it must have a
higher priority than local policies. For example, let us consider
the following constraint: "Task $T_2$ of organization OrgA and
task $T_3$ of organization OrgB must be executed by the same
subject."

The IWFC can receive this constraint either from the OrgA
or from the OrgB. These organizations know about the inter-
organizational constraints that they must satisfy but they do not
know how to ensure such a type of constraint. Indeed, each
organization does not have the security policy of the other
organization. So, a third part, which is the VO, will take into
its responsibility this communication. The subject who will
execute the task $T_2$ into the organization OrgA can have a
different role when executing the task $T_3$. We suppose that

Figure 5. Security policy deployment using a provisioning approach

B. inter-organizational case

For inter-organizational workflows, several workflows are
involved in the achievement of a common mission. Interactions
between different organizations are necessary in this case. Such communications have to be well controlled since each
organization has its private information, its private policy and
its own characteristics. A communication protocol is needed in
order to ensure these different communications and flows that
can transit from an organization to another. This requirement
is critical especially to enforce the constraints specification
defined with the workflow specification (see section IV-A).

If these constraints are defined within the same organization,
the WFC can enforce and check if its user role assignment
violates the base of constraints or not. Otherwise, if we
suppose for example that constraints of the examples of section
IV-A are defined within an inter-organizational workflow and
that task $T_i$ is provided by an organization OrgA and the
task $T_j$ provided by an organization OrgB, enforcing workflow
constraints becomes more complicated. Indeed, synchronization
between different organizations must be done in order to
enforce constraints.

1) Creation of the VO: For both types of constraints, we
need to create a virtual organization (VO) in order to synchro-
nize between different organizations and to enforce different
workflow constraints. The VO is responsible for the secure
inter-organizational communications. To create a VO, the inter-
Alice is the subject who will execute these tasks. Thus, for these two tasks, OrgA and OrgB can define respectively these rules and role assignments in their local security policies:

1) security_rule(permission, OrgA, clerk, sign, check, nominal)  
   empower (OrgA, Alice, clerk)
2) security_rule(permission, OrgB, manager, validate, mission, nominal)  
   empower (OrgB, Alice, manager)

To be sure that the subject Alice is executing the task $T_2$ of OrgA, the $WFC_2$ has to inform the IWFC about the execution progress of its workflow. Such a communication allows the IWFC to update its security policy and its different contexts. Indeed, if we suppose that when the task $T_2$ will be executed, Alice will not be ready to perform the task, the $WFC_2$ will be obliged to change the subject executing the task and must inform the IWFC about this change.

So, to enforce the constraint above, the subject Alice has to switch from the role clerk within the organization OrgA to the role manager within the organization OrgB. This transition may imply a leakage of information. The problem is more dangerous if Alice transfers a confidential or sensitive data. This confinement problem has to be managed using an information flow control policy managed by the VO. To define this policy, we are based on the information flow control model DTE (Domain and Type Enforcement model) [8], [7]. An integrated model based on OrBAC model and using a DTE approach is presented in [5]. Let us define the entities that this model introduces.

**Definition 4:** (domain) $S$ is a set of all system subjects (active entities). $S$ is divided into equivalence classes. Each class represents a domain $D$ including a set of subjects having the same role in the system.

**Definition 5:** (type) $O$ is a set of all system objects (passive entities). $O$ is divided into equivalence classes. Each class represents a type $T$ including a set of objects having the same integrity properties in the system.

**Definition 6:** (Entry Point) An entry point is a program or an activity which must be executed to pass from a domain $D_1$ to a domain $D_2$, denoted $EP(D_1, D_2)$ or $EP\_1\_2$. An entry point implies two rules: (1) subjects passing from $D_1$ to $D_2$ obtain a set of privileges depending on the entry point they execute, (2) subjects passing from $D_1$ to $D_2$ lose all their $D_1$ privileges.

The first rule means that the execution of an entry point defines the set of privileges that subjects will obtain when moving from a domain $D_1$ to a domain $D_2$. These privileges are included into or equal to the set of privileges that $D_2$ subjects have. Each domain can have more than one entry point. The execution of these different entry points implies different privilege sets. The second rule means that if a subject leaves a domain it can not return to it only by executing one of its entry points.

To be coherent with the access control model that we use, namely the OrBAC model, a domain corresponds, within an organization, to a role. The entry points to a domain correspond to different entry points needed to be assigned to a role temporarily, if a subject does not belong to the organization where this role is defined. Thus, to define its security policy, the IWFC sends a request to different organizations to get their entry points to their different roles (i.e. domains). Then the IWFC security policy is a set of rules defined as:

security_rule(permission, VO, $D_i$, Enter, $D_j$, through($E_{i,j}$))

Based on an OrBAC model IWFC rules are specified as specific OrBAC rules. The rule above expresses the transition between the domain $D_i$ belonging to an organization $Org_i$ and the domain $D_j$ belonging to an organization $Org_j$ and uses the entry point concept to define a new context in order to preserve secure information flows and to keep DTE aspects. Therefore, to consider this particular rule we suppose the following hypotheses: (1) the source domain is considered as a role in the OrBAC rule, (2) the destination domain is considered a view in the OrBAC rule, (3) the transition between two domains can be expressed as an OrBAC activity since the basic meaning of this specific rule is to handle interactions between domains. For this purpose, we define the OrBAC Enter activity, (4) the entry point defines the manner to enter into a domain. Thus, an entry point can be included into a specific context in an OrBAC rule denoted through($E_{i,j}$). This context specifies that the rule is valid only through the $E_{i,j}$ execution.

a) Through($E_{i,j}$) context: it is a composed context defined by an organization $Org_j$ as Through($E_{i,j}$) = $\{E_{i,j}, Pr_j, coming\_from(Org_j)\}$. The first argument represents the activity to execute to enter a domain. This activity can be implemented as an action on the system or a program to execute or a procedure to be released. The second argument gives the set of privileges that will be granted to the subject who intends to enter a domain $D_j$ when coming from a domain $D_i$. This set of privileges depends on the activity executed to enter the domain. So, to each entry point corresponds a different set of granted privileges. Since the entry points can also depend on different organizations from where subjects are coming, we define the last argument coming_from($Org_i$). For example, let us suppose that we have two organizations $Org_j$ and $Org_k$ which define a role secretary. If two subjects having this role and coming respectively from $Org_j$ and $Org_k$ need to move to a role clerk in the organization $Org_i$, they will not get the same set of privileges even if they execute the same entry point. This results from different sensitivity degrees of different organizations and also from the relationship between the organizations. For instance a special privilege can be granted to subjects coming from an organization and moving to an organization which has already cooperation and agreement. This context can be composed with other OrBAC contexts. Then it may include extra conditions and circumstances to
supervise the flow control.

The communication between different WFCs and the IWFC are considered secure and all organizations are considered belonging to the same trust alliance. If it was not the case, the policy of the IWFC would be much more complicated and more dynamic. All communications have to transit through a trust server to communicate information about actual status and about the local workflows progress.

VI. CONCLUSION

Several works have investigated security in WFMS since these systems present a special requirement to access and information flow control models. They are characterized by a diversity of tasks, roles, subjects and objects. But once the workflow security policy is defined and specified, the workflow manager has to know how to deploy such a policy. In this paper, we address the new issue of deploying a workflow security policy. We present different approaches and modes to administrate the functional and the security part of a workflow.

Then, we introduce an approach to deploy the workflow security policy, once it is defined and specified. This management is studied for two different cases: intra and inter organizational workflows. For the intra organizational case we have defined functionalities that different entities (WFC, PEP and PDP) must manage. The implementation of this case implemented through calls to the API OrBAC which is responsible for defining the workflow security policy. For the second case a more rigorous control is needed since communications between different and independent organizations is required. Thus, a virtual organization must be introduced to make possible the synchronization and the communication between different organizations. The VO is responsible for ensuring the non violation of workflow constraints and managing the transition between domains of different workflow subjects.

For both cases, the security policy is based on the OrBAC model to specify the workflow security policy. In this work we assume that organizations involved in the workflow belong to some trust alliance that provides some degree of confidence between all parts of an inter-organizational workflow. If a trust alliance does not exist we have to resort to a trust server who will be responsible for leading the whole workflow execution. In a forthcoming paper, we intend to go in details of the implementations of the intra and inter organizational case.

REFERENCES


