SLA Driven Process Security through Monitored E-contracts

Ritesh Kumar Tiwari
CDE, IIIT-Hyderabad, Hyderabad, India
ritesh@research.iiit.ac.in

Vishal Dwivedi
SETLabs, Infosys Technologies, Bangalore, India
vishal_dwivedi@infosys.com

Kamalakar Karlapalem
CDE, IIIT-Hyderabad, Hyderabad, India
kamal@iiit.ac.in

Abstract

In this work we look into the domain of process security from a service perspective. Most often process security has been enacted through service level agreements (SLA) and business agreements. However, in a multi-party environment such as business process outsourcing (BPO) where processes themselves are offered as a service, the qualitative nature of SLA makes their monitoring quite difficult and their implementation through various restrictions, quite costly. We present our approach wherein we provide security as a process represented using e-Contracts and enacted through workflows. We explore if security too could be offered as a service which could be enacted and monitored by the process participants themselves; thus ensuring more trust. Most of the process based systems employ either Task Based Model (TBAC) or Role Based Model (RBAC) for granting privileges that are needed for executing the individual activities of the workflow. Current approaches are either potentially weak from security perspective, as they grant even those permissions to user which are actually not needed by him for executing the tasks, or they have very high administrative overhead. In this paper, we propose to couple RBAC with TBAC and additionally enforce sequential and temporal constraints over them so that process participants get only 'Need to know information'[*] with less administrative overhead. In this paper, we propose our extended e-contract framework for security (EC^{FS} framework), and the architecture of a system which implements it. In the end we present a brief case study presenting our process security model.

1. Introduction

Process security has been one of the tougher security related problems because of the qualitative nature of the entities involved. Most often it is enacted through SLA's which themselves are more qualitative and are often in practice achieved through appropriate restrictions and guidelines. Even after having strong system level and network level security, the system security could be compromised by insecure processes. In domains such as a BPO where the processes (workflows) are human centric most of the security SLA's are often implemented as rules, the execution of which could not always be monitored with the current approaches.

As of now, we still lack the presence of a model and formalism for provable process security[†]. Workflow security requires context sensitive authorization support at per task granularity for both subjects and objects. Workflow authorization framework must take into account SSOD (static separation of duty), DSOD (dynamic separation of duty), cardinality, temporal and sequential constraints for making authorization decisions. Warner et.al. [1] proposed to use intra-instance and inter-instance workflow constraints taking into account the execution history of completed workflow instances for mitigating collusion attacks. In practical scenarios, users may have to delegate their assigned tasks to other users because of performance and availability constraints. Such delegations are usually short lived, and workflow authorization framework should support such task delegations based on user's capability (i.e. roles) and permission authorization. Atluri et.al [2] proposed a secure conditional delegation framework where delegation decisions are based on time, workload and task attributes.

Taking into consideration, the comprehensive security requirements of underlying workflow, we propose our e-contract framework for security (EC^{FS}). The main contributions of this paper are: 1) EC^{FS} security framework which could model security contract enactment 2) A methodology to ensure TBAC and RBAC allowing only 'need-to-know' information being accessed by process participants 3) Providing support for intrusive and non-intrusive auditing and monitoring

[*] Developing access control model which satisfies "*need to know*" constraint with low administrative overhead is still a challenge.

[†] A model capable of encapsulating controlled access to data objects, secure execution of tasks in an intended manner, with efficient management/administration of security

of security e-contracts through meta workflows 4) Ensuring a quantitative bounded provisioning of security through legal clauses which are not just representative but can also be enacted through workflows.

2. Previous Work

Security, *flexibility* and *performance* are the most important aspects of organizational workflow. The Task Based Authorization Model [3] was introduced to provide the notion of "*just in time*" permissions via processing states and life cycle of authorizations. Joshi et.al. [4] proposed a Generalized Temporal Role Based Access Control (GTRBAC) to support temporal role enabling and disenabling using trigger, and a set of language constructs for specification of temporal constraints on roles. Some good amount of published work focuses on various security aspects of workflows, but none of them provides a comprehensive framework for mapping high level Service Level Agreement between parties to actual security enforcement at workflow (i.e. task) level. The proposed EC^{FS} framework aims to fill this gap by providing a model for mapping high level contract specification between parties to actual security enactment at operational workflow (i.e. task) level.

There has been some recent work on e-contract enactment and execution but many of them have focused on the modeling and enactment issues. ER^{EC} framework [5] defined means to design e-contract processes and to deploy, monitor and execute them. SweetDeal system by Benjamin et al [6] and work by Grefen et al [7] were aimed in solving this problem. We extend the ER^{EC} framework for the security domain and explore the ways to ensure the enactment of a security e-contract, with meta-workflows providing support for both intrusive and non-intrusive monitoring.

2.1 Open Issues

Service Level Agreement between two organizations is often a *qualitative*, *linguistic*, high level specification of service agreements between organizations. Transition from this SLA specification to actual system implementation (i.e. workflows), and then proving the correctness (i.e. compliance) of implementation w.r.t. specification is an open challenge. Traditional approaches mainly focus on SDLC\ (software development life cycle) and following procedures and guidelines but there is no way that those could be directly automated and monitored.

In case of workflows, the task execution is repetitive in nature. So the user executing the task should have only those privileges that are needed for task execution. This notion of "*need to have privileges*"[‡] can be provided by TBAC. TBAC directly associates privileges with the tasks. Hence operational flexibility at user level is compromised to a great extent. So until and unless, the same user has to perform the same task repetitively, using TBAC exclusively for managing security authorization in workflows is not advisable, as it will result in high administrative overheads.

[‡] Through out this paper, we have used the terms "need to know" and "need to have" interchangeably.

RBAC [8] on the other hand associates users to roles (user-role assignment), and permissions are assigned to roles (role-permission assignment). This abstraction provided by role gives administrative flexibility. The notion of roles and role hierarchies quite often leads to assigning even those extra permissions to users which are not needed by them for executing a task. This opens a window of vulnerability, which can be exploited by malice to gain more information and capability to launch attacks.

As evident, TBAC provides Task-permission assignment which is administratively inflexible but secure. On other hand, RBAC provides user-role assignment and role-permission assignment which is administratively flexible but not very secure. So the open question is – can we achieve "user-task-permission" assignment that is secure as well as administratively flexible. In section 4.3, we show how to achieve this requirement by coupling best features of RBAC and TBAC.

3. Problem Statement

In this work we aim to provide a security framework which lists the steps involved in ensuring process security through a security e-contract enactment. Our aim is to achieve administratively less costly high level of security through efficient processes for enacting security SLA. We propose to use RBAC for administrative purposes and TBAC for permission authorization at task level (actual workflows). The synergistic operation of RBAC and TBAC models when coupled with sequential and temporal constraint enforcement mechanism provides users (i.e. actors)

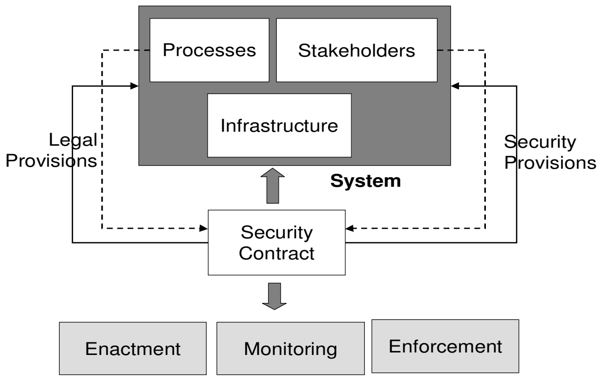

Figure 1: Process Security through e-contracts

with "need to know" information and this minimizes chances of information leakage resulting from unused extra activated privileges. Further the approach should ensure that the clauses and the provisions as agreed to by the participating parties in the contract, could be audited, monitored and controlled through services while the contract is enacted through business processes.

4. Ensuring security through e-contracts

In this work we propose an approach wherein we provide security as a process represented using e-contracts and enacted through workflows. We extend the notion of e-contracts as in [5] to the security domain. We define a security e-contract as an agreement between two or more parties over a set of security measures, tasks and clauses pertaining to actions to enforce security and prevent breaches which could be modeled, specified, monitored and executed by a software system. The set of clauses ensure that processes follow the defined security mechanisms which could be audited by any party. An e-contract thus lays down a set of deterministic tasks and rules which need to be followed during process execution.

Implementing security e-contract at systems level needs significant caution and effort as we need to guarantee that:

1. All security conditions specified in clauses are correctly implemented.
2. No other operation can be performed (or information is extracted) than what has been specified to be done on a task by a specific role (or user).

4.1 Overview of the EC^{FS} security model

We extend the ER^{EC} framework [5] to include the security mechanisms which could be instantiated as per a given clause for a set of tasks (refer Figure 3). We present our Extended E-contract Framework for Security (EC^{FS} framework) which allows the modeling, design, and deployment of security e-contracts, and their monitoring through security meta-workflows which could ensure both intrusive and non-intrusive monitoring. (in latter case for auditing purposes)

Our EC^{FS} has the following four main layers:

1. ***Document Layer:*** This layer comprises of XML based e-contract specifications, the examples of which could be any of the following: RuleML, ECML, tpaML, ebXML etc.
2. ***Conceptual Layer:*** This layer comprises of the meta-model of the security e-contract, the UML structure for which is given in Figure 4. It is mainly used for representing the activities, clauses, the actors, their inter-relationships and the security mechanisms which should guide the activities under specific contract clauses.
3. ***Logical Layer:*** This layer comprises of the data model, event condition action (ECA) rules, activity party clause (APC) constructs and is mainly used to map the e-contract to underlying business processes. We introduce the notion of meta-workflows which are representative workflows which map to actual workflow instances. Not only they provide a level of abstraction of actual business processes but they tremendously help in both intrusive and non-intrusive monitoring of workflow instances. They also tremendously help in ensuring that only 'Need-to-Know' information is available to the process participants, by enabling RBAC and TBAC even in a delegation driven environment. As given in Figure 3, the run time granting of privileges to any workflow instance activity is enabled through meta-workflows.
4. ***Implementation Layer:*** This layer provides for the workflow management and execution infrastructure along with the service infrastructure (web-services) for business activity monitoring (BAM) and in this case monitoring the execution of processes which were defined through an e-contract.

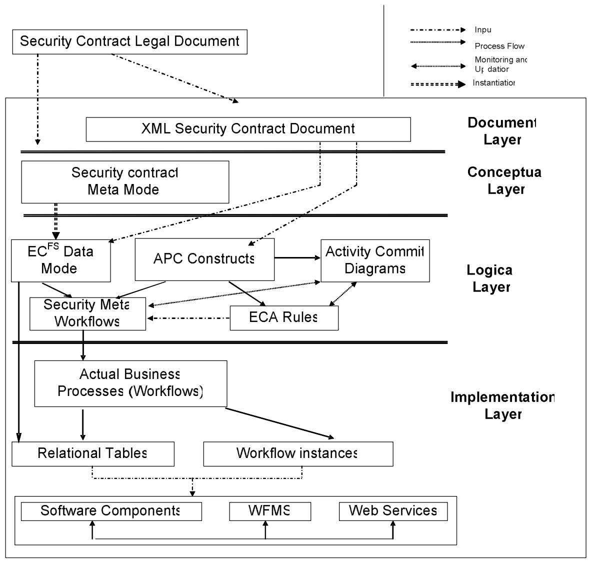

Figure 2: EC^{FS} security framework for e-contracts [extended version of ER^{EC}]

4.2 Implementing the EC^{FS} model

In Figure 3 we represent a sample security contract for a BPO company processing banking information. The contract defined above lays down clauses which define rules. Exceptions from the clauses are provided as rules which can be triggered or may trigger certain

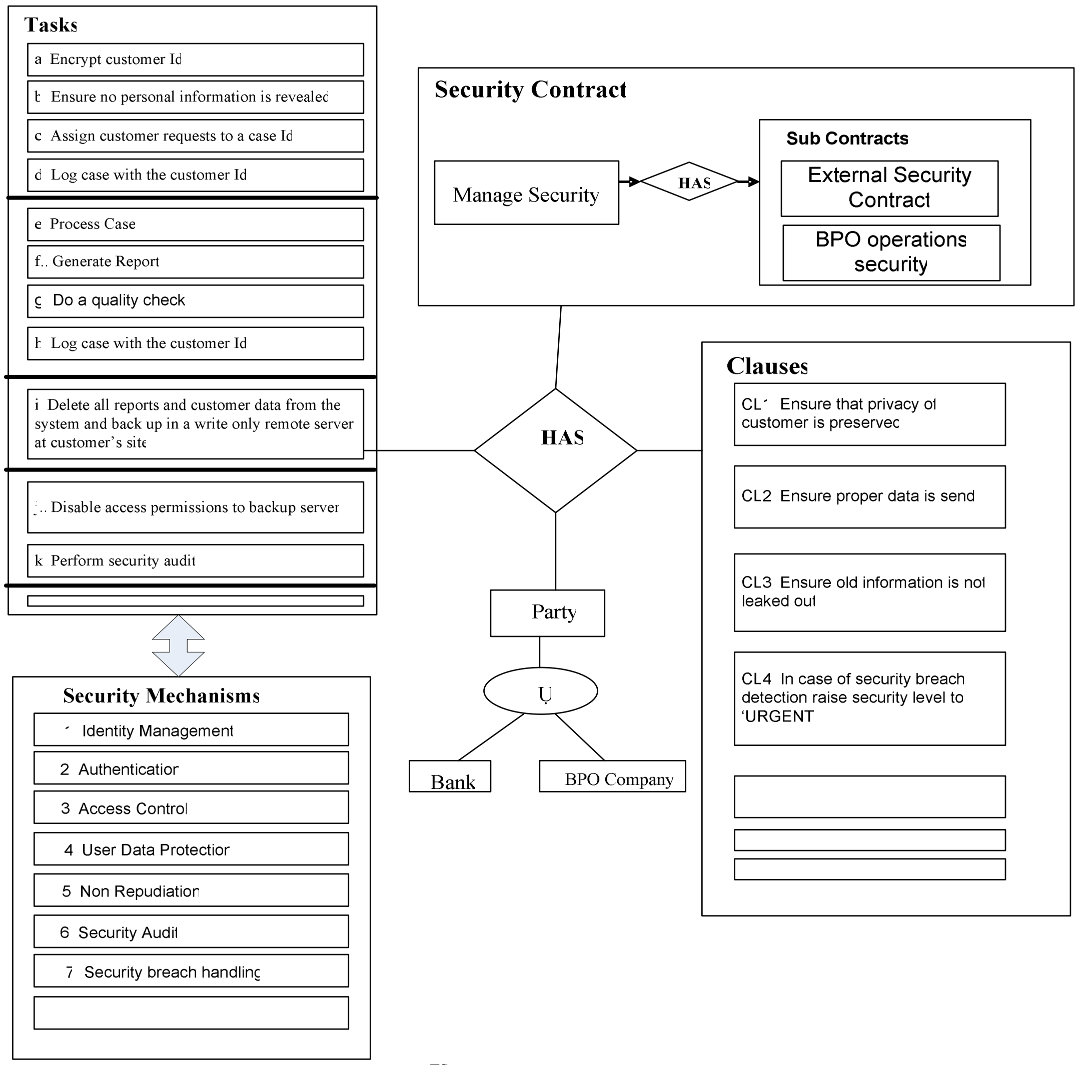
Tasks
a Encrypt customer Id
b Ensure no personal information is revealed
c Assign customer requests to a case Id
d Log case with the customer Id
e Process Case
f.. Generate Report
g Do a quality check
h Log case with the customer Id
i Delete all reports and customer data from the system and back up in a write only remote server at customer's site
j.. Disable access permissions to backup server
k Perform security audit
Security Mechanisms
1 Identity Management
2 Authentication
3 Access Control
4 User Data Protection
5 Non Repudiation
6 Security Audit
7 Security breach handling
Security Contract
Manage Security
HAS
Sub Contracts
External Security Contract
BPO operations security
HAS
Party
U
Bank
BPO Company
Clauses
CL1 Ensure that privacy of customer is preserved
CL2 Ensure proper data is send
CL3 Ensure old information is not leaked out
CL4 In case of security breach detection raise security level to 'URGENT

Figure 3: EC^{FS} diagram of a security e-contract

tasks. A set of tasks need to be executed to fulfill a particular clause by ensuring a particular security mechanism. The given e-contract is then represented in XML consisting of a set of ordered pairs of clauses, activities and actors. Our security meta-model is similar to ER^{EC} meta-schema template but differs in being domain specific and using security mechanisms for realization of the clauses through tasks. We further use the notion of meta-workflows as defined by Chiu et al in [9] which provides support for handling exceptions, business rules and workflow instances. These meta-workflows in practice help us to achieve a level of abstraction over workflow instances and are designed to enable monitoring support in our EC^{FS} framework. In the security perspective, not only they help in auditing but they can also be used as a vigilant entity, observing the accesses of workflow instances in the implementation layer. A snapshot of these meta-workflows could be exposed to outside world as process dashboards or they could be designed to provide data to underlying web-services to provide static data about executing workflow instances and their level of abeyance of the security clauses. Any breach of a laid down clause could raise an exception which could be logged and captured by the all the parties.

These meta-workflows could be efficiently used for both intrusive and non-intrusive monitoring of business processes and provide the audit results to the parties concerned. In the next section we discuss how these provide support for TBAC and RBAC allowing only need-to-know' information being accessed by process participants.

4.3 Approach for enacting EC^{FS} at System level

In Figure 4 we express the interrelationship between various elements of EC^{FS} framework.

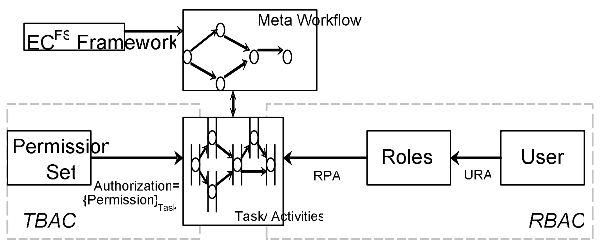

Figure 4: Enforcing EC^{FS} at System Level

A fundamental question that comes is – given the e-contract, how to *generate security policies* and how to *implement them* at system level for fulfilling the terms of contract. A possible solution to this challenging problem is shown in Figure 5.

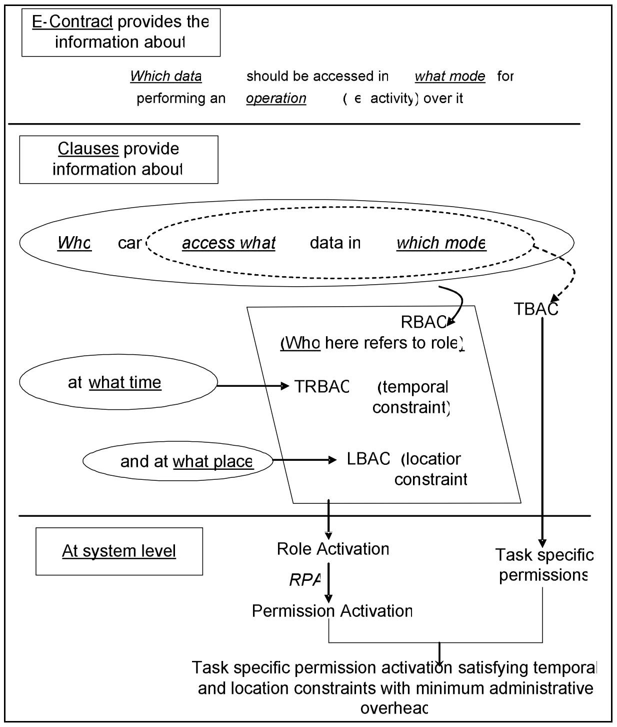
E-Contract provides the information about
Which data should be accessed in what mode for performing an operation (e activity) over it
Clauses provide information about
Who can access what data in which mode
RBAC (Who here refers to role)
TBAC
at what time
TRBAC (temporal constraint)
and at what place
LBAC (location constraint
At system level
Role Activation
RPA
Permission Activation
Task specific permissions
Task specific permission activation satisfying temporal and location constraints with minimum administrative overhead

Figure 5: Conceptual mapping of E-contract to runtime permissions

Let set of tasks to be executed on a workflow be represented as $\{T_1, T_2, \cdots, T_n\}$ and corresponding authorization (permissions) needed for executing them be $\{\{P\}_{T_1}, \{P\}_{T_2}, \cdots, \{P\}_{T_n}\}$. In TBAC model, the necessary permissions are exclusively associated with task $(\exists T_K, \{P\}_{T_K} \rightarrow T_K)$. The advantage of using TBAC at task level is that it provides notion of *protection states,* which represent *active* permissions that are maintained for each authorization step. The protection state of each authorization step is *unique* and *disjoint* from protection state of other steps. So even if two tasks have overlapping permissions set, then also the intersection of their protection set is a null set. TBAC can enforce *strict usage*, *validity*, and *expiration* characteristics on permissions (i.e. can dynamically manage permissions) as authorization progresses to completion. For operational flexibility at task level, TBAC also supports notion of *composite authorization* by associating a *critical set* of component authorization-steps with every composite authorization.

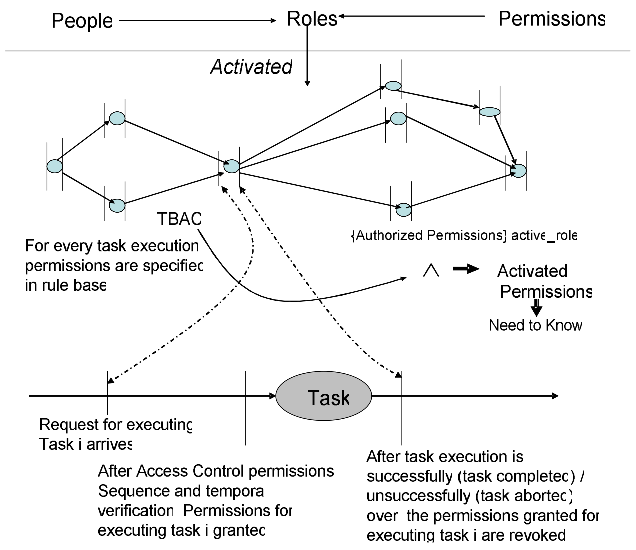
People
Roles
Permissions
Activated
TBAC
For every task execution permissions are specified in rule base
{Authorized Permissions} active_role
∧
Activated Permissions
Need to Know
Task
Request for executing Task i arrives
After Access Control permissions Sequence and temporal verification Permissions for executing task i granted
After task execution is successfully (task completed) / unsuccessfully (task aborted) over the permissions granted for executing task i are revoked

Figure 6: Ensuring RBAC and TBAC through meta-workflows

Let $\left(\overset{n}{\underset{i=1}{\forall}} U_i\right)$ denote set of users, $\left(\overset{m}{\underset{j=1}{\forall}} R_j\right)$ denote set of roles, and $(P)_{R_i}$ denote set of permissions corresponding to role R_i. Let *AS* is set of *activity states* defined as-
AS={Dormant,Invoked,Valid,Hold,Revoked,Complete}

An activity *A* is then defined as 5-tuple- $\langle AS, \{P\}_A, \{R\}_A, \{constraint\}_A, A-type \rangle$ where, AS is activity state, $\{P\}_A$ is set of permissions which can be performed over activity *A*, $\{R\}_A$ is set of roles who are permitted to perform this activity, $\{constraint\}_A$ represents *sequential* and *temporal* constraints that should be satisfied for transiting *A* from dormant to invoked state, and $A-type$ represents type of activity (i.e. sequential or repeated). The first four parameters of tuple <*A*> are of main consideration from security perspective.

4.3.1 Enforcing EC^{FS} clauses through RBAC and TBAC policies

Given any user U_i, based on his capabilities, he will be assigned a subset of roles $\left(\forall_k R_k\right)_{U_i}$ from role set $\left(\overset{m}{\underset{j=1}{\forall}} R_j\right)$ where $\left(\left(\forall_k R_k\right)_{U_i} \subseteq \left(\overset{m}{\underset{j=1}{\forall}} R_j\right)\right)$. Let the set of *direct* and *indirect*[4] (i.e. inherited) permissions corresponding to assigned role set be $\left(\forall_j P_j\right)_{\left(\forall_k R_k\right)_{U_i}}$.

When the user U_i is assigned to perform a specific task T_i, the user invokes those roles R_k whose permissions constitute *superset* of permissions actually needed for executing task T_i i.e. $\{P\}_{T_i} \subseteq \left(\forall_j P_j\right)_{\left(\forall_k R_k\right)_{U_i}}$. Given the fact that residual permission set after consumption of permissions (i.e. completion of task) is not null (and hence can be cause of security problems), so it is always advisable to revoke all unused permissions after the task executing is complete. Another problem with extra privileges is that malicious employee can misuse them during ongoing execution of task to compromise process security. So a mechanism to neutralize (i.e. nullify) the capability of unused extra privileges allocated by role-permission-assignment (RPA) mechanism is needed. We approach this problem by splitting access control enforcement mechanism in EC^{FS} in two layers- At higher layer, we used abstraction provided by RBAC for "*user-role-permission*" assignment and at lower system layer, we used TBAC for "*job-permission*" assignment. During runtime, when user executes certain task, the *common privileges* from RBAC (Role Permission Assignment - RPA) set and TBAC (task permission assignment -TPA) set are taken to enable the actual permissions which a user can execute at runtime on the task.

The advantage of this two tier access model is low administrative cost in case of dynamic *user-task assignment*, and the enactment of "need-to-have" privileges by TBAC. The disadvantage of proposed model is that it is restrictive in allowing permissions that can be executed by user (i.e. restricts the capability of user), and hence should be used in access control enforcement for those tasks whose security is critical (i.e. whose compromise severely limits the functioning of entire process chain).

[4] *Indirect permissions* are the permissions which are actually assigned to a child node and inherited by parent node in node hierarchy.

Another challenge is how to incorporate *Temporal* (*TC*), *Location* (*LC*), and *Sequence* (*SC*) (i.e. precedence) constraints for role activation and task execution. So the clauses in *EC^{FS}* framework will be a mapping into the set $\left\{\{SC\}^* \times \{TC\}^* \times \{LC\}^*\right\}^*$ where the *cardinality* of constraint set at minimal can be null. Every workflow can be represented as partially ordered set of tasks coordinated by set of events, where the order of task execution is governed by constraint satisfaction conditions as described in clauses of EC^{FS} framework.

	Temporal Constraint	*Location Constraint*	*Sequence Constraint*
RBAC	Yes (by GTRBAC [4])	Yes (by LBAC [12])	---
TBAC	Yes (by enforcing them on rule base)	---	Yes (by enforcing them on rule base)

Table 1: Feasibility of Constraint representation in RBAC and TBAC

As shown in table 1, temporal constraints can be enforced using both RBAC and TBAC. Temporal constraints enforced using RBAC govern which user can invoke which role at what time. Temporal constraints enforced using TBAC govern time when a task can be initiated, or the time duration within which some task has to be completed. The later case when coupled with exceptional handling procedures can help in figuring out possibly malicious intent of an employee (for e.g., a bank clerk is supposed to enter transaction amount into customer database, and on average he takes 10 seconds to make an entry, but if he takes much longer time in all the transactions that he is updating, then the security system can become suspicious that the clerk might be trying to leak sensitive information about customer).

Location constraints are important when a critical task has to be executed from a specific location. Rather than binding location sensitivity to task, we bind the role sensitivity to location, and allow only the privileged role which is authorized to execute at specific desired location and satisfies the "*common privilege*" criteria with the task be able to operate on task. If the task movement is static, then location constraint can be associated at task level, but in generic case, from operational flexibility perspective, it is recommended to relate location constraint to a role.

Sequential constraints are task dependent and basically comprises of pre-conditional events that must be fulfilled before a task can be executed. For example, in order to pass the travel requisition form, administration assistant should validate travel form, after validation it is forwarded to accountant who verifies user identity and clears the requisite finance payment, which is forwarded to manager, who signs up the form and authorize sanction of travel grants to the employee. Such sequences (i.e. precedence constraints) can be effectively captured at task level (i.e. by TBAC) rather than at abstract role level (by RBAC)[5].

4.4 The security SLA monitoring services and infrastructure

The meta-workflows ensure that there is a proper mapping between EC^{FS} data model and active workflow instance. Some of the key security SLA's could be gauged and monitored through the execution of meta-workflows. These also form the basis for SLA monitoring services which could be exposed to the participating parties. The key idea is to provide a process dashboard on top of meta-workflows to the participants so that they can monitor the clauses through task execution in the workflow instances in the implementation layer. It is similar to a BAM (business activity monitoring) console over these meta-workflows which is a regular feature of most current workflow management systems.

Some of the minimal services could be the following:

- GetActiveWorkflowInstance Service
- GetExceptionReport Service
- Get UnderlyingWorkflowInstanceStatus Service
- PauseUnderlyingWorkflowInstance Service
- GetPrivilegeForActivityInstance Service

Note that these services are apart from the regular orchestration, deployment and management services for the actual workflow instances executing in the implementation layer. The aim of these is i) to directly monitor workflow instances based on e-contract security clauses ii) provide a dashboard to contracting parties showing the status and exceptions of SLAs on the underlying processes captured through e-contract clauses.

[5] A role is assigned to perform a set of tasks that constitutes an activity.

5. Case Study: Process security in a BPO

A leading US financial institution outsourced some of the non core functions to an Indian Business Process Outsourcing (BPO) company. The process entailed capturing of relatively unstructured data and sending back structured output files over a secure channel; involving human and machine participation for completing the business processes across multiple geographical locations. The data processing operations involved processing sensitive financial information, accessing multiple reports and manipulating figures which although classify as backend operations but are quite sensitive for the financial institution.

The financial institution in its project contract established various security-SLA's containing strict security guidelines including restricted entry, no external network connectivity, dual firewall and other security restrictions. The institution also arranged for frequent audits of the BPO location. In spite of all the strict measures, some questions still remained unanswered:

1. How to ensure that all security SLA's are followed? Is there a better way to monitor?
2. How to ensure that in spite of all security arrangements security is not compromised by an employee itself.
3. Security in itself is unbounded; where to draw a line between service and operating margins.

For the BPO things are further complex. Not only it has to ensure that security is not breached, it has to strike a balance between performance, profits and security. A security contract (as in Figure 4) would help in quantifying the level of efforts and possible tasks, but its enactment remains the key solution.

Our EC^{FS} framework could be effectively used in such a scenario. A security e-contract negotiated between the parties would put in place quantitative tasks and clauses to be acted on by the participants. Our methodology ensures coupled RBAC and TBAC by providing permissions to roles based on tasks. For example even if a manager has all privileges, we ensure that he is provided with as much permissions as required to complete a given task. We feel this has a significant impact on the considered domain and lessens the chances of an insider attack.

On a level higher, EC^{FS} provides mapping between contract and meta-workflows and between latter and workflow instances. Apart from granting privileges, these meta-workflows could be tremendous aid in intrusive and non-intrusive monitoring. A process dashboard on these workflows, or a BAM (business activity monitoring) console based on web-services, could ensure that contract parties can remotely access and monitor the process execution and the clauses which are governed by the meta-workflows. Any exception could then be noticed and remedial action can be immediately taken. Intrusive meta-workflows could even be designed to enact temporal constraints and allow static separation of duty (SSOD) and dynamic separation of duty (DSOD) and even allowing support for delegation which is a frequent in a service environment, without losing on security.

6. Conclusions and Future Work

In this paper we proposed our methodology to ensure security of processes through e-contracts which could be enacted, monitored and executed by the contract parties to ensure that clauses are implemented properly. We customized the ER^{EC} framework and proposed EC^{FS} security framework which could model security contract enactment and monitoring. We introduced the notion of coupled RBAC and TBAC which could ensure the provision of 'Need to Know' information, as privileges are given to roles specific to a particular task. In scenarios such as deferred tasks this could be of some real importance. We also provide for a mapping between e-contract representation and their corresponding processes. Thus we ensure that instead of currently prevalent norm of representing security through SLA requirements it could even be enacted through contracts and implemented through workflows, which could be monitored and controlled to provide information about clause violations.

This paper can be extended for enacting "need to have privileges" in an Inter organizational workflow. However, enforcing DSOD constraints during runtime in an Inter-organizational workflow is a challenge. The key idea behind this paper was to provide contracted security which could be enhanced and extended using multiple ways. Another approach on which we are working right now is to specify contracts from a conceptual model to standard XML SLA language, like WSLA [14], which would make enactment easier. We believe some of the problems we try to solve in this work are open problems and our approach provides a small but novel step towards their solution.

References

[1] Janice Warner, Vijayalakshmi Atluri, "Inter Instance Authorization Constraints for Secure Workflow Management", Symposium on Access Control Models and Techniques, 2006. pp. 190-199

[2] Vijayalakshmi Atluri, Janice Warner, "Supporting Conditional Delegation in Secure Workflow Management Systems", Symposium on Access Control Models and Techniques, 2005. pp. 49-58

[3] Roshan K. Thomas, Ravi S. Sandhu: Task-Based Authorization Controls (TBAC): A Family of Models for Active and Enterprise-Oriented Authorization Management. DBSec 1997: 166-181

[4] James B. D. Joshi, Elisa Bertino, Arif Ghafoor, "Analysis of Expressiveness and Design Issues for a Temporal Role Based Access Control Model," IEEE Transactions on Dependable and Secure Computing, April-June 2005

[5] P. Radha Krishna, Kamalakar Karlapalem, Dickson K. W. Chiu: "An EREC framework for e-contract modeling, enactment and monitoring", Data Knowl. Eng. 51(1): 31-58 (2004)

[6] Benjamin N. Grosof and Terrence C. Poon, "SweetDeal: Representing Agent Contracts with Exceptions using XML Rules, Ontologies, and Process Descriptions", In Proceedings of the 12th International Conference on the World Wide Web, 2003.

[7] S. Angelov, P. Grefen; "The 4W framework for B2B e-contracting", Int. J. Networking and Virtual Organisation, Vol. 2, No.1, 2003; pp.78-97

[8] R. Sandhu et al, "Role-Based Access Control Models", IEEE Computer, 29(2), Feb. 1996. pp. 38-47

[9] Dickson K. W. Chiu, Qing Li, Kamalakar Karlapalem: "A Meta Modeling Approach to Workflow Management Systems Supporting Exception Handling", Inf. Syst. 24(2): 159-184 (1999)

[10] Kamalakar Karlapalem, Ajay R. Dani, P. Radha Krishna: "A Frame Work for Modeling Electronic Contracts," ER 2001: 193-207

[11] Patrick C. K. Hung, Kamalakar Karlapalem: A Secure Workflow Model. ACSW Frontiers 2003: 33-41

[12] Maria Luisa Damiani, Elisa Bertino, Barbara Catania, Paolo Perlasca, "Geo-RBAC: Spatially Aware RBAC", ACM Transactions on Information and System Security, vol. 10(1), Feb 2007.

[13] Christian Skalka, X. Sean Wang, "Trust but Verify: Authorization for Web Services", ACM Workshop on Secure Web Services, 2004. pp. 47-55.

[14] Keller, A., Ludwig, H., The WSLA Framework: Specifying and Monitoring Service Level Agreements for Web Services, Journal of Network and Systems Management, Special Issue on E-Business Management, Volume 11, Number 1, Plenum Publishing Corporation, March, 2003