Security for Workflow Systems

Vijay Atluri, Rutgers University

1. What are Workflow Systems?

Organizations constantly reengineer and optimize their business processes to reduce costs, deliver timely services, and enhance their competitive advantage in the market. Reengineering involves assessment, analysis, and redesign of business processes, including introducing new processes into existing systems, eliminating redundant processes, reallocating sharable resources, optimizing the process execution, among others. Business processes relate to tasks that create, access, process and manage business information throughout organizational information systems that include databases.

As advances in *database management systems* (DBMS) take place to facilitate business transactions, organizations are seeking ways to effectively integrate their business processes, and automating them as much as possible. The advent of database technology has made the change of data more adaptive by successfully separating the access of data from the applications. However, any change and enhancement to the business policies would entail modifying application code, as the business policy is still mainly hard-coded [ED97]. *Workflow systems* are a step in the direction in providing both automation and reengineering functionalities. The fundamental idea of workflow technology is to separate the business policy from the business applications to enhance flexibility and maintainability of business process reengineering. This separation facilitates reengineering at the organization level without delving into the application details. Other advantages include supporting resource allocation and dynamically adapting to workload changes.
As a testament to the recognition of these benefits, workflow systems are today used in numerous business application domains including office automation, finance and banking, healthcare, telecommunications, manufacturing and production, for automating their day-to-day applications, as well as in scientific application domains such as in spatial processes and DNA sequencing.

Workflow management aims at modeling and controlling the execution of business processes involving a combination of manual and automated activities in an organization. A workflow is defined as a set of coordinated activities that achieves a common business objective [WFMC]. Thus, a workflow separates the various activities of a given organizational process into a set of well-defined tasks. An activity, or a *task*, is a logic step or description of a piece of work that contributes toward the accomplishment of a process [RS94,GHS95]. Tasks may be carried out by humans, application programs, or processing entities according to the organizational rules relevant to the process represented by the workflow. Tasks that build up the workflow are usually related and dependent upon one another, which in turn are specified by a set of execution constraints called *task dependencies*. These task dependencies play a key role in supporting various workflow specifications such as concurrency, serialization, exclusion, alternation, compensation and so on. For ensuring the correctness of workflow execution, tasks have to be executed in a coordinated manner based on these dependency requirements. A workflow management system (WFMS) is a system that supports process specification, enactment, monitoring, coordination and administration of workflow process through the execution of software, whose order of execution is based on the workflow logic [WFMC]. In the following, we provide a concrete example of a workflow to provide a better understanding of tasks and dependencies.

Example 1: Consider for example a travel reimbursement processing workflow, shown in Figure 1. This workflow consists of four tasks: preparing a claim (T_1), approving the claim (T_2), issuing a check (T_3), and notifying the employee in case of declination (T_4). Note that, there exist dependencies that represent the coordination constraints among these tasks. For example, the task dependency "bs" between T_1 and T_2 states that T_2 can begin only if T_1 successfully finishes. Similarly, T_3 can begin only if T_2 successfully finishes, and T_4 begins if T_2 fails, (represented as a "bf" dependency). Therefore, a check will be issued if the claim is approved, otherwise a notification is sent to the employee. Also note here that each task is executed by a processing entity, in this case by humans. For example, task T_1 is executed by an employee, T_2 has to be executed by a supervisor, whereas T_3 and T_4 need to be executed by clerks, possibly T_3 by a clerk in the disbursement office, and T_4 either by a clerk or by a computerized process invoked by a clerk in the supervisor's department.

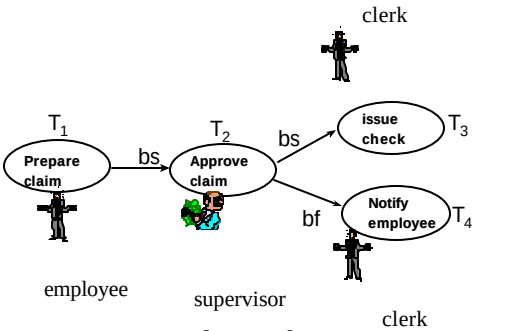

Figure 1: Travel Reimbursement Process Workflow

Workflow technology has been widely promoted by software vendors over the past few years. More than a few dozen products are available on the market including IBM's MQ series, Domino Workflow form Lotus Corporation (IBM), Workflow product from Oracle, COSA Workflow, FlowMark, Staffware, Ensemble, Deskflow, InConcert, OPEN/workflow, Visual Workflo, FlowMake, FlowWork, Microsoft Exchange Server/ Microsoft Outlook Client, TeamWARE Flow, Ultimus, just to name a few. These technologies and products support a range of business applications including banking and financing (loan processing), legal, insurance (claim processing), health care (clinical processes), military, manufacturing, telecommunication, and office automation.

2. Security Requirements in Workflow Systems:

A number of security measures need to be taken into account while building a secure workflow system. These include:

- *Confidentiality*: This refers to unauthorized disclosure of information including the workflow specification, and the workflow instances during its execution
- *Integrity*: Refers to the unauthorized modification of information, again including the workflow specification as well as the data manipulated during the execution of a workflow instance

- *Availability*: Refers to making the data and resources available to the agents responsible for executing the tasks in a workflow
- *Authentication*: Refers to reliably verifying the identity of the task execution agents
- *Authorization*: Refers to enforcing access control to ensure confidentiality and integrity
- *Audit*: Refers to recording information about who has performed which actions at what time within the workflow, which can later be analyzed to detect suspicious behavior and misuse of authority
- *Anonymity*: Refers to keeping the agents anonymous from other agents executing the workflow. This may be needed especially when agents with conflict-of-interest execute different tasks within a workflow
- *Separation of duties*: These are additional constraints associated with the workflow to reduce the risk of fraud

In the following sections, we will address some of the well explored issues listed above, not necessarily in the same order.

3. Authorization

A proper authorization system protects information against both direct and indirect leakage to unauthorized users. These authorization requirements can be broadly classified into two categories: (1) discretionary access control (DAC), in which users, at their discretion, are allowed to grant permissions on the data they own, and (2) mandatory access control (MAC), in which all data are labeled based on their level of sensitivity and all users are given clearances, and users are allowed to access data with a given label only if their level of clearance permits them. While MAC is essential in environments with very stringent security requirements such as the military, DAC is suitable for almost all commercial applications. Hence, we focus our attention only on DAC.

3.1 Access Control Requirements

In order to ensure that the various tasks in a workflow are executed by authorized users or processes (subjects), proper authorization mechanisms must be in place. The traditional authorization model expresses an authorization as a tuple (s,o,p), specifying a subject (s) can gain privilege (p) to an object (o). To ensure that authorized subjects gain access on the required objects only during the execution of the specific task, granting and revoking of privileges need to be synchronized with the progression of the workflow from one task to another. For example, consider once again the ``travel reimbursement processing workflow," that involves ``preparation of the claim" by the employee, ``approval of the claim" by his supervisor, and ``issue of the check" by a clerk in the disbursement control department. An effective authorization model must make sure that it is not possible to change the amount of the claim by an employee after its approval by his supervisor.
This is not feasible unless there exists a security system that authorizes an individual in synchronization with the progression of workflow.

Although access control models exist that allow specification of temporal authorizations associated with a time interval, they are not perfectly suitable for workflow environments. An authorization model to represent temporal privileges has been proposed by Bertino et al. [BBFS95]. Since in this

model authorization is granted during a predefined and fixed time interval, it is not suitable when dealing with workflows because appropriate authorizations should be granted or revoked synchronously with the starting and ending of a task. A predefined specification of the privileges often allows access to objects for periods longer than the required time. As a result, though a subject completes the task or have not yet begun the task, it may still possess privileges to access the objects, resulting in compromising security. It is difficult to predict the actual execution time of each task in many workflow situations and therefore not possible to determine their time interval in advance during the specification phase. Even if one can determine the temporal interval during which a task has to be executed, delays experienced by one task may propagate to subsequent tasks thus delaying their execution. Thus by the time the task actually starts, the required authorization may expire. On the other hand, it is not practical for a security administrator to monitor the workflow and grant (or revoke) authorizations accordingly.

Currently, this must be implemented as application code and embedded into the various tasks. Such an approach makes it difficult, if at all possible, the specification and management of authorizations, given also the large number of tasks that typically occur in a workflow. Therefore, it is essential to have workflow authorization models in place, as organizations are moving towards a paperless office environment where electronic documents are exchanged among individuals and automated processing agents.

3.2 Workflow Authorization Model

As described in section 1, a workflow deals with coordinated execution of tasks that involve processing of each of the tasks in the workflow by subjects (either humans or programs). To execute a task, relevant privileges on required objects have to be granted to appropriate subjects. Atluri and Huang have proposed a *Workflow Authorization Model* (WAM) [AH96] that is capable of specifying authorizations in such a way that subjects gain access to required objects only during the execution of the task, thus synchronizing the authorization flow with the workflow. To achieve this synchronization, WAM uses the notion of an *Authorization Template* (AT) that can be associated with each task, which allows appropriate authorizations to be granted only when the task starts and to revoke them when the task finishes. WAM *dynamically* assigns authorizations to support workflow activities in a way that the time interval associated with the required authorization to perform a task changes according to the time during which the task actually executes. AT is comprised of the static parameters of the authorization that can be defined during the design of the workflow. ATs are attached to tasks. A task may have more than one authorization template attached to it. Multiple ATs are required in cases where there are more than one type of object to be processed, or more than one subject is required to perform the processing. An authorization template allows us to specify rules such as ``A supervisor is allowed to approve a travel claim." These can actually be stated during the design phase by the workflow designer. When the task starts execution, its AT(s) is used to derive the actual authorization. When the task finishes, the authorization is revoked. However, the authorization to approve a specific claim is granted to a supervisor only when the task of approval actually starts. And this privilege will be revoked when this task is completed. This is accomplished by placing an *object hole* in the authorization template. A new authorization is granted to a subject on the specified object only when the object hole is filled with that object, and if the object's type is same as that specified in the template. We illustrate this with the following example.

Example 2: Consider once again the workflow in example 1. For the sake of simplicity, we have omitted T4. Suppose the associated subjects for performing these processes are John, Mary, and Ken, respectively. Now, instead of granting all the required privileges for every involved staff in advance, they potential authorizations are specified by means of the authorization templates. Appropriate authorizations to perform these tasks are not enforced until the tasks are actually processed. The templates given below can be attached to tasks, as shown in Figure 2.
AT(T1) = (employee, (claim,--), prepare)
AT(T2) = (Supervisor,(claim,--),approve)
AT(T3) = (clerk, (check,--), issue)
Now suppose that John starts preparing the claim. He has to first obtain a claim form. As soon as the object hole in the authorization template is filled with the claim form, John receives the authorization to prepare it. Assume he starts this at time 40. At this point, John is granted the authorization to prepare the claim. Suppose he finishes it and sends it to his supervisor at time 47. The authorization template then generates the authorization (John, claim1, prepare, [40,47]), which means the authorization is revoked as soon as he finishes his task. When the claim (the instance is claim1) arrives to Mary at 47, an authorization to approve is given to Mary. However, John no longer holds the authorization on this instance of the claim any more. When Mary finishes the approval task, say at 82, her authorization is revoked, thus generating (Mary, claim1, approve, (47,82)). In this fashion, WAM synchronizes the authorization flow with the progression of the workflow.

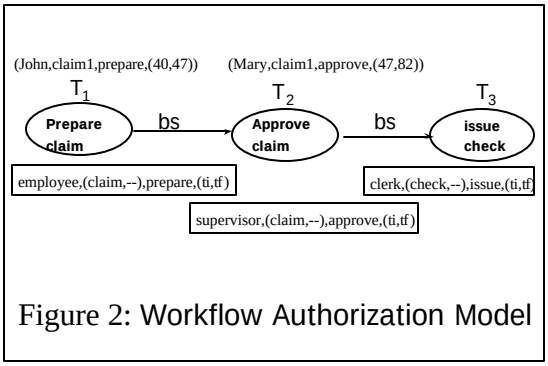

Figure 2: Workflow Authorization Model

4. Separation of Duties Constraints

Often, organizational workflows are very large with users in the range of several thousands and number of process instances in the range of tens of thousands. Quite often, security policies of a given organization are expressed in terms of the roles within the organization rather than individuals. Roles represent organizational agents intended to perform certain job functions within the organization. Users in turn are assigned to appropriate roles based on their qualifications. Specifying authorizations on roles is not only convenient but reduces the complexity of access control because the number of roles in an organization is significantly smaller than that of users. Moreover, the use of roles as authorization subjects, instead of users, avoids having to revoke and re-grant authorizations whenever users change their positions and/or duties within the organization. Furthermore, role-based authorization is particularly beneficial in workflow environments to

facilitate dynamic load balancing when a task can be performed by several individuals. As a matter of fact, many commercial WFMSs support role-based authorizations. With traditional role-based access control (RBAC), roles are assigned to users based on their qualifications, and tasks in turn are assigned to roles, thereby assigning permissions to users. However such a simple model of RBAC is not adequate in WFMS.

A common drawback of role-based authorization models used in current WFMSs (or even DBMSs) is that they are not capable of modeling authorization constraints on roles. A typical authorization constraint, which is very relevant and well-known in the security area, is separation of duties [S91]. An example of a business rule reflecting such a constraint would be, ``an employee preparing a claim cannot be the same as the one issuing a check." As evident from this example, the aim of such constraints is to reduce the risk of frauds by not allowing any individual to have sufficient authority within the system to perpetrate a fraud on his own. Separation of duties is a principle often applied in everyday life; e.g., opening a safe requires two keys, held by different individuals, approval of a business trip requires approval of the department manager as well as an accountant, and a paper submitted to a conference requires to be reviewed by three different referees who must be different from the author(s).

Under the principle of separation of duties, a complex action is decomposed into several smaller steps, which are executed by different individuals. Therefore, perpetrating a fraud would require collusion of several individuals, thus making it more difficult. To directly represent such organizational security policies, an access control mechanism must therefore be capable of supporting roles as well as the constraints.

Separation of duties constraints can be categorized into the following three types [BFA99]:

1. *Static constraints*: These constraints can be evaluated without executing the workflow. Examples of such constraints include: (i) At least three roles must be involved in executing the workflow. (ii) The same role must execute tasks T1 and T2.
2. *Dynamic constraints*: These constraints can be evaluated only during the execution of a workflow, because they express restrictions based on the execution history of the workflow. If John belongs to role R1 and has performed task T1, then he cannot perform T2. The constraint mentioned above in the context of claim processing is a dynamic constraint. This is because the employee for each instance of the claim processing, one cannot enforce the constraint only during its execution.
3. *Hybrid constraints*: These are constraints whose satisfiability can be partially verified without executing the workflow. An example of such a constraint would be, task T2 must be executed by a role dominating the role, which executes task T3.

Even though separation of duties has been applied to computerized information systems, current WFMSs provide no support for it. If no proper support is provided, constraints like separation of duties must be implemented as application code and embedded into the various tasks. Such an approach makes it difficult, if at all possible, the specification and management of authorization constraints, given also the large number of tasks that typically occur in a workflow. In [BFA99], Bertino et al., present a formalism to specify and enforce these authorization constraints.

5. Authentication

With the explosion of the Internet technologies, the demand on business process management has raisen to a new level. Web and workflow management systems together serve as an ideal combination to integrate the distributed processes that are across or within enterprise boundaries. This is due to the fact that, Web is already globally distributed, robust and reliable communication mechanisms are already in place, web browsers are commonplace, and web servers can interact with the databases via CGI programs to store, retrieve, and route data. The Internet-based workflow systems have drawn a lot of attention recently [DCY97, HA99] as they offer a number of advantages. First, they support better mobility by allowing users to access workflow systems from virtually any computer connected to the Internet with a standard web browser. In addition, by using Java applets, the user can acquire the program that performs the task on demand without prior installation on user machine. Second, with the global naming scheme for resources (e.g. URLs), protocols for accessing named resources (e.g., HTTP), HTML, Javascript and some other emerging Web technologies such as eXtensible Markup Language (XML) and Java, all together provide an ideal combination for the development of client-server collaborative workspace. Third, they are extremely scalable.

Due to this, companies are anticipating to use the public Internet as a vehicle to conduct business-to-business transactions such as electronic commerce. Such web-enabled workflow systems can serve effectively for more dynamic internet-based business processes over heterogeneous computing platforms. Some examples of such processes include global supply chain management, universal telecommunication service management, and mobile patient care service management.

However, a web-based workflow system calls for additional and more stringent requirements on security. The most compelling one is the authentication. Web-based systems should additionally provide secure communication. They may also need to employ security services such as the public key infrastructure to provide confidentiality, integrity and authentication [ASKP00].

6. Anonymity

Execution of inter-organizational workflows may raise a number of security issues including those related to *conflict-of-interest* among competing organizations, especially when they are executed by mobile software agents without using a centralized control the flow. In such an approach, the entire workflow is sent to the first task execution agent, which executes its task, and then sends the remaining workflow to the next task execution agent. That is, the workflow percolates from agent to the other as its execution progresses. If the task execution agents belong to the same conflict-of-interest group, it may lead to unfair, and in some cases, undesirable results, akin to being the wrong side of the *Chinese wall.* The Chinese wall policy, proposed by Brewer and Nash [BN89] for information flow in a commercial sector, states that information flows from one company (e.g. Bank A) to another (e.g. Bank B) that cause conflict of interest for individual consultants should be prevented. The policy enforced is that people are allowed access to information, which is not in conflict with any other information that they already possess. The company information is categorized into mutually disjoint conflict of interest classes. We take a concrete example to illustrate the problem.

Example 3: Consider a business travel planning process that makes reservations for a flight seat, a hotel room and a rental car. The workflow that depicts the process at a travel agent (shown in Figure 3) consists of the following tasks: T1 Input travel information, T2: Reserve a ticket with Continental Airlines, T3: if T2 fails or if the ticket costs more than $400, reserve a ticket with Delta Airlines, T4: if the ticket at Continental costs less than $400, or if the reservation at Delta fails, purchase the ticket at Continental, T5: if Delta has a ticket, then purchase it at Delta. T6: Reserve a room at Sheraton, if there is flight reservation, and T7: Rent a car at Hertz.

Assume that each task is executed at the appropriate agent, for example, T2 by Continental, T3 by Delta, etc. Consider the dependencies between T2-T3 and T2 -T4, which state that T3 should begin only if T2 is not successful or the outcome of T2 is more than $400, and T4 starts when T2 's outcome is $400 or less. Examples such as this are not unusual (consider priceline.com) where a customer sets a maximum he is willing to pay, but not necessarily looking for the best price. At the same time, he may have preferences for the merchants whom he wants to do business with, for example preferences for a specific set of airlines in a certain order to accrue frequent flyer miles. If a mobile agent is used to execute such a workflow, after the execution of T2, Continental must send the remaining workflow to Delta, based on the outcome of T2. That is, Continental now has the knowledge that if it fails (that is, no ticket is available) or if the ticket costs more than $400, the task needs to be sent to Delta airlines, which is a competing company. Due to this fact, Continental can manipulate the price of the ticket and may reduce it to $399, which may result in a loss of business to Delta or may prevent the customer from getting a better price than $399, which may potentially be offered by Delta.

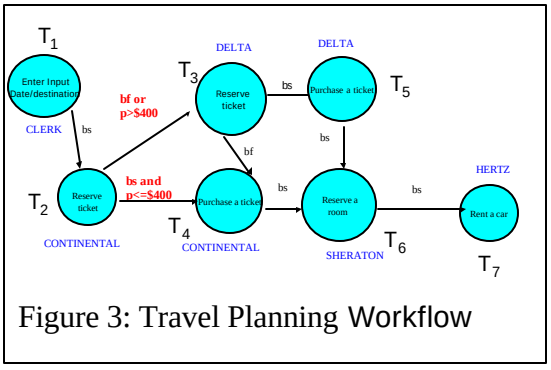

Figure 3: Travel Planning Workflow

It is important to note that hiding everything from Continental except T2 by appropriately encrypting the workflow will not work. This is because Continental has to evaluate the dependency conditions in order to forward the remaining workflow, and therefore it should know both the dependency and the following agent. In [ACM01], Atluri et al. propose a solution to resolve such problems by partitioning the workflow in a restrictive manner.

7. Conclusions

Most commercial workflow systems provide minimal security features such as user authentication. Although some commercial WFMSs such as FlowMark, Lotus Notes and Changengine can support role-based access control, they do not provide support to specify and enforce separation of duties constraints. They have to be implemented in an ad-hoc manner through a script type language. Such ad-hoc implementation makes specification, analysis and maintenance of security policies

more difficult. In addition, many efforts in WFMS implementation have been placed in protecting data transmitted over the network, but little emphasis has been given on providing access control for workflow activities.

In this chapter, we highlight the security requirements of workflow systems and discuss authorization, separation of duties, authentication and anonymity at length. Although our treatment of authorization emphasizes the need for synchronization of authorization flow with the workflow, we recognize that a full-fledged authorization system should consider the following additional requirements: assigning different roles to tasks based on the outcome of the prior task, granting different permissions to roles based on the outcome of the task, capability to specify different authorizations for different instances of the same workflow, ability to specify authorizations based on the context and based on the responsibilities to be performed by individuals, and delegating the responsibility to other users and roles. Currently, security in workflow systems is an active area of research, and many researchers around the world are investigating the above issues.

References:

[ACM01] V. Atluri, S. Chun, and P. Mazzoleni, “A Chinese Wall Security Model for Decentralized Workflow Systems,” submitted for publication, 2001.

[AH96] V. Atluri and W-K. Huang, ``An Authorization Model for Workflows," Proceedings of the Fifth European Symposium on Research in Computer Security, Rome, Italy, and Lecture Notes in Computer Science, No.1146, Springer-Verlag, September, 96, pages 44-64.

[ASKP00] G. Ahn, R. Sandhu, M. Kang, J. Park, “Injecting RBAC to Secure a Web-based Workflow system,” 5th ACM Workshop on Role-based Access Control, July, 2000.

[BBFS95] Elisa Bertino, Claudio Bettini, Elena Ferrari, and Pierangela Samarati, “A temporal access control mechanism for database systems.” IEEE Transactions on Knowledge and Data Engineering, 8(1):67--80, 1996.

[BFA99] E. Bertino, E. Ferrari and V. Atluri, ``An Approach for the Specification and Enforcement of Authorization Constraints in Workflow Management Systems," ACM Transactions on Information Systems Security, to appear in February 1999, Vol. 2, No. 1.

[BN89] D.F.C. Brewer and M.J. Nash. “The chinese wall security policy.” In Proceedings of the IEEE Symposium on Security and Privacy, pages 206 --214, 1989.

[DCY97] U. Dayal, Q. Chen, and T. Yan., “Workflow technologies meet the internet.” In Asuman Dogac, Leonid Kalinichenko, M.Tamer Ozsu, and Amit Sheth, editors, Advances in Workflow Management Systems and interoperability, pages 343--358. NATO Advanced Study Institute, 1997.

[ED97] Ahmed K. Elmagarmid and Weimin Du. “Workflow Management: State of the Art vs. State of the Market.” In Advances in Workflow Management Systems and Interoperability, pages 1--17, August 1997.

[GHS95] Dimitrios Georgakopoulos, Mark Hornick, and Amit Sheth. “An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure.” Distributed and Parallel Databases, pages 119--153, 1995.

[HA99] W-K. Huang and V. Atluri, ``SecureFlow: A Secure Web-enabled Workflow Management System," 4th ACM Workshop on Role-based Access Control, October, 1999.

[RS94] Marek Rusinkiewicz and Amit Sheth. “Specification and Execution of Transactional Workflows.” In W.Kim, editor, Modern Database Systems: The Object Model, Interoperability, and Beyond. Addison-Wesley, 1994.

[S91] Ravi S. Sandhu, “Separation of Duties in Computerized Information Systems. In Sushil Jajodia and Carl Landwehr, editors, Database Security, IV: Status and Prospects, pages 179--189. North Holland, 1991.

[WFMC] Workflow reference model. Technical report, Workflow Management Coalition, Brussels, Brussels, 1994.