Context Based Enforcement of Authorization for Privacy and Security in Identity Management*

Vasu Alagar[1] and Kaiyu Wan[1]

[1] Concordia University
Montreal, Canada
alagar@cs.concordia.ca
[2] East China Normal University
Shanghai, P.R.C
kaiyu.wan@gmail.com

Abstract. Protecting the identity of an individual is a shared responsibility between the individual, the organizations with whom the individual will be transacting during her life time, and the state of which the individual is a legal resident. Identity theft occurs when someone uses an individual's personal information without the knowledge of the individual to commit a crime, such as fraud or theft. Of late identity theft has become one of the fastest growing crimes, not only in western countries but also in developing countries where internet dominates business, financial transactions of big organizations, and social activities of individuals. In this paper we discuss a context based enforcement of authorization to protect the privacy of individuals and secure information about them stored in large identity management systems.

1 Introduction

Privacy and security are complementary issues. From an individual's point of view privacy is paramount. From an organization's point of view security is a quality of service (QoS) requirement. In practice, it is required to protect the identity of an individual (honoring her privacy), while at the same time revealing information of that individual to another individual who is authorized to get such information. In several countries privacy is regarded as a fundamental right and is implemented in state legislation. To assure privacy of an individual and to decide who is authorized to get a specific information about that individual, *identity* is the common factor.

The *identity* of a person is established at birth and it may exist even after the death of a person. At birth, the birth *certificate* records the place and date of birth of the child, the name(s) given to the child, the identities of her legal parents, and any identifiable physical marks of the child. This certificate, issued by a *trusted authority* who is authorized by the state to issue it, establishes the identity of the child. At death, the death certificate is issued which in principle should make a *reference* to the birth certificate of the deceased. The death certificate does not necessarily annul the birth certificate but

*The research is supported by a grant from Natural Sciences and Engineering Research Council, Canada.

only disassociates the identity of the person from the person, because the person no longer exists. In between birth and death, more and more certificates may be issued to a person by various trusted organizations and government agencies of the country where the person lives. Different certificates of the individual are normally used to establish the identity of the person at different *contexts*. We formalize the context notion in Section 2.2. Until then we use the term "context", as defined in Oxford English Dictionary: *context defines the circumstances that form the setting for an event*.

We distinguish between *credential* and certificate: a certificate of a person is a declarative statement which is true at all times, whereas a credential presented by a requester is a document which should *prove* beyond doubt the identity of the presenter to an *authority* who at the context of its presentation is responsible to enforce certain *policies*. Examples of automated systems that use credentials to establish the identity are electronic voting systems, on-line banking systems, and on-line health care systems. A typical scenario in completing a transaction in such systems involves three steps: (1) a client (subject) submits a request, usually supported by a set of *credentials*; (2) the system applies the current policy to the submitted credentials to establish the identity of the user, and validates whether or not the user has the rights to receive the requested resources; (3) the system applies the security and privacy policy in servicing the request, if step 2 is successful. Thus, identity management in a system requires the management of a policy base and an automated procedure for applying policies at different contexts.

A policy is a *rule* that applies to some scenario in the daily life-cycle of the organization's activity. Broadly, the rules can be classified into *business rules* and *security rules*. Business rules describe terms and conditions, service provisions, contracts and their execution. Typically, a work-flow specification in an organization is driven by business rules. On the other hand, security rules set restrictions on access to resources and regulate information flow. Security policies are domain-specific, restricting access to objects in that domain. When an organization sets up a system to automate identity management, the application of a policy defines a certain choice in the behavior of the system. It is important to separate the policy from the system implementation, in order to allow the policy to be modified which in turn can dynamically alter the system behavior, without changing the system implementation. In this paper we explain how this may be achieved using contexts.

The main contributions of this paper are: (1) security contexts for enforcing security; (2) policy base organization as a rule-based system with context condition attached to each rule; (3) a formal approach to automate authorization and secure service provision.

2 Notation and Concepts

In this section we make precise three basic concepts and give their abstract view. By a system we mean a set of policies PB, interacting *subjects (entities)* S, and a set of *objects (resources)* O controlled by subjects. The *status* of a subject $s \in S$ at any time in the system is one of the following: (i) s is an individual subject; (ii) s belongs to one or more groups; or (iii) s plays a set of roles with or without group membership.

2.1 Identity of an Entity

It is commonly understood [2] that the identity of an individual comprises a set of personal information about the person. We try to make this concept as precise as possible. Determining the identity of an entity x arises whenever it enters into a transaction which in turn requires access to a resource o controlled by another entity y. The requester x must provide sufficient credentials to y so that y can establish the identity of x and then determine the type of access that can be granted for x over o. The task of determining the identity is called *authentication*. Knowing the identity of the requester, the act of granting appropriate access to o is based on access control policy. Authorization is authentication *plus* granting access.

In traditional systems, which are closed systems, the requester is identified by the system because the credentials are known only to each other. However, in many internet applications the resource owner, who has to authorize, and the requester are unknown to one another. The requester may want to remain anonymous and present only a minimum amount of information, which may be a partial set of credentials furnished by third parties. Based on the submitted credentials the system has to decide whether or not to authorize the request. Below is a typical scenario arising in on-line banking systems.

Example 1 *Alice has a checking account with ABC bank and is privileged to use the bank's on-line banking facility. She is given a 14 digit account number and has chosen a password to go with it. The personal information of Alice, collected by the bank at the time of opening the account, is saved under her personal profile. She has also recorded with the bank a set of skill-testing questions and her answers for the questions. Assume that whenever she uses her personal computer at home to access the banking facilities, her account number and password combination is accepted by the system as proof of her identity. Once on her travel, let her access the on-line facility using the computer in her hotel room. After her usual log-in session, she may get a message "the system is unable to identify you", and direct her to contact the bank either in person or through telephone. Why her identity is not authenticated? A possible answer is that the bank's online system uses a whole lot of information that it has collected and saved in the system file to authorize her. When she accesses the system from the hotel computer, the system infers non-compliance with her access history and decides not to authorize her. Another possible scenario is when she accesses the system from a hotel room or internet cafe in her later travels: the system, instead of denying her service, may try harder to prove her identity. The system may interrogate her with one or more randomly chosen skill-testing questions saved in her profile. The system will allow her access if her response to the query matches the answer stored in her profile. Hence, the system will require different sets of credentials to be presented at different contexts in order to authenticate its user.*

Based on the above discussions we propose a definition of identity:

Definition 1 *A certificate is a declaration about an entity or an object, issued by a trusted authority, who is not the entity. A credential is a declaration of an entity about an entity (possibly self-referencing). A certificate is a credential. The identity of an entity at any context is a non-empty subset of certificates accumulated by the entity until that*

context. Authentication in a context is the proof of compliance that the credentials that are presented by an entity in the context are sufficient to establish the identity of the entity in that context.

2.2 Context

We use the definition of context given in [7], in particular use their definition and notation for security contexts. To understand the need to formalize context, let us consider the policy *Policy1: A physician on-duty in a hospital can access the medical records of patients either admitted by her or treated by her.* This policy refers to physician name, her current status (on-duty or off-duty), patients admitted by her, and patients under her care. The context for enforcing the policy is suggested by the above information. Notice that the context information is *multidimensional*, and is determined by the four dimensions PN (a finite set of physician names), PS (a finite set of statuses of physicians), WS (a finite collection of work schedules), PA (a finite set of patients admitted) and PC (a finite set of patients cared). Associated with each dimension there exists a finite set of values, called tags, as indicated above. An example of a context c with these dimensions is represented in the syntax $[PN : Bob, PS : on-duty, WS : 1, PA : Alice, PC : Tom]$. This context describes the setting in which "physician Bob is on duty on the first day of the week, admitted Alice and cared for Tom". "Policy1" can be applied to this context to validate Bob for accessing medical records, as well as track the flow of information of the patient records. What is important is to make clear that "Alice" and "Tom" are patients and not hospital personnel. That is, context definition requires a unique dimension name for each entity type, because a hospital patient may also be an employee in the hospital.

The security context types introduced in [7] are useful for representing rules in the policy base and for enforcing policies. Let us briefly comment on how and where they fit in our work. Three security categories proposed in [7] are *Internal Security Category*(ISC), *Boundary Security Category*(BSC), and *External Security Category*(ESC). A context in ISC category specifies the authorization for one user to access data in one category. An example of ISC context is $[UC_1 : u, DC_2 : j, PC : Legal]$, meaning that user u with role UC_1 is allowed to access the data (resource) type j in category DC_2 for legal purposes. This context type is useful for the access policy that specifies "which entity can access a resource and for what purpose". A context in BSC category specifies the configuration of fire wall security for each user category. An example of BSC context is $[NAME : Alice, SP : NULL, IF : vlan100, IF : vlan120, CLASS : gold, CURL : root]$. The BSC contexts are usually configured by the system administrator to optimize resource utilization and system protection. This context configured at the fire wall authenticates remote users and directs their requests through interfaces that can apply the access policies. A context in ESC category specifies the contextual information governing the user's request. An example of ESC context is $[LOC : Berlin, TIME : d_1, WHO : Alice, WHAT : filetransfer, WHERE : Paris, WHEN : d_2, WHY : Auditing]$. This context type is relevant for monitoring "data transfer" policies.

For an efficient representation and retrieval of rules from the policy base we associate with a context of one type a set of contexts of another type. We may call this

association *lifting*. Intuitively, a context c and a context c' that is lifted from it have something in common: a policy that is valid in context c may also be valid in context c'. Usually, contexts c and c' will have some common dimensions or same tag types of some of the dimensions in c are the same as those of some dimensions in c', although the two dimension sets are different. Lifting is used in Section 2.3 for constructing security contexts and in Section 4.1 for linking related policies.

2.3 Access Policy and Grant Policy

We modify the grant policy defined in traditional systems in order to make it context-specific. Abstractly, we define access policies by functions: (1) Function AS assigns to an individual $s \in S$ a set of signed actions, called *access rights*, on an object $o \in O$. If $+a \in AS(s,o)$, then the subject s is allowed to perform action a on object o; however, if $-a \in AS(s,o)$, then the subject s is not allowed to perform action a on the object o. If neither $+a$ nor $-a$ is $\in AS(s,o)$, the access policy is undefined for subject s on object o. (2) Function AG assigns to a group $g \in G$ a set of rights on an object $o \in O$. The function SG gives for a subject s the groups $SG(s)$ to which the subject s belongs. (3) Function SR gives for each individual subject $s \in S$, the set $SR(s)$ of roles assumed by s. The function AR defines for each role $r \in R$, the set $AR(r,o)$ of rights that r has on the object o.

We define the grant policy as a function SP, which for a subject s in context c grants or denies access to object o. We use the notation $PB@c$ to mean "the evaluation of policy base PB at context c". The result of evaluation is either a successful validation (true) or an unsuccessful validation (false). In the former case a non-empty subset of PB, rules that are applicable for context c, is also retrieved from the policy base. Section 5 explains $PB@c$ in detail.

1. [P1:] *s is an individual subject* The subject s is granted to perform the actions explicitly allowed for it on the object o if there exists no policy in context c that overrules that privilege . $SP(s,o,c) = if\ PB\ @\ c\ then\ AS(s,o)\ else\ \emptyset$
2. [P2:] *s has a set of roles but is not a member of a group* The subject s is granted the right to perform an action a on an object o in context c if at least one of the roles in $SR(s) \neq \emptyset$ is authorized to access o and none of them is denied to access o in context c. $SP(s,o,c) = \{+a \mid p_r(a,s,o) \wedge a \in A \wedge r \in SR(s)\}$, where $p_r(a,s,o) \equiv PB\ @\ c \wedge +a \in AR(r,o) \wedge \sim \exists r' \in SR(s) \bullet (-a \in AR(r',o))$.
3. [P3:] *s has no roles and belongs to one or more groups* In context c the subject s belonging to the groups in $SG(s)$ is granted to perform an action a on an object o, if at least one of the groups in $SG(s)$ is authorized to access o in context c and none of the groups in $SG(s)$ is denied to access it in context c. $SP(s,o,c) = \{+a \mid p_g(a,s,o) \wedge a \in A \wedge g \in SG(s)\}$, where $p_g(a,s,o) \equiv PB\ @\ c \wedge +a \in AG(g,o) \wedge \sim \exists g' \in SG(s) \bullet (-a \in AG(g',o))$.
4. [P4:] *s has a set of roles and belongs to one or more groups* Using the predicates defined in the previous two steps we define $SP(s,o,c) = \{+a \mid p_r(a,s,o) \wedge a \in A \wedge r \in SR(s)\} \cup \{+a \mid p_g(a,s,o) \wedge a \in A \wedge g \in SG(s)\}$

The grant policy is applied only if a request is successfully validated. The procedure for validating a request at the security contexts is as follows: (1) *Fire wall-* From the

user request, an ESC context associated with the request is extracted. A BSC context is constructed by "lifting" the ESC context. (2) *Authorization:System Access-* One of the servers in the interface IF of the BSC context should authenticate the user. If the authentication fails service is denied. If the authentication succeeds the security policy SP of the context is applied. The result of this application is either "Allow" or "Deny". If the permission is granted, the request is taken up for internal processing. (3) *Access/Grant-* An ISC context is constructed from the ESC and BSC contexts and the grant policy is applied in this context. The next example illustrates this procedure.

Example 2 *Let* $[LOC : Berlin, TIME : d_1, WHO : Alice, WHAT : filetransfer, WHERE : Paris, WHEN : d_2, WHY : Auditing]$ *be the ESC context constructed from a request made by* $Alice$*. The dimension* WHO *from ESC is mapped to the dimension* $NAME$ *in* BSC*, partially constructing* $[NAME : Alice]$*. This step is justified because these two dimensions have the same tag set. The dimensions* $LOC, WHERE, WHAT, WHY$ *from ESC taken together are mapped onto the dimension* SP *in* BSC*. That is because the policy for transferring a file is relevant for the context defined by these dimensions. The part of that policy, say* p*, that is relevant to the interface is assigned to* SP*. Thus, the constructed context* $[NAME : Alice]$ *is extended to* $[NAME : Alice, SP : p]$*. From the name* $Alice$ *and the fire wall configuration policy for user categories, the system will determine the rest of the dimensions* IF*,* $CLASS$*, and* $CURL$*, and complete the construction of the* BSC *context corresponding to the* ESC *context. This step may fail. If it fails, the request is not validated at the fire wall. If the validation is successful the ISC context is constructed. The system constructs the context* $c' = [WHO : Alice, WHAT : filetransfer, WHY : Auditing]$ *by first mapping* $NAME$ *to* WHO*, next mapping the dimensions* $CLASS$ *and* $NAME$ *from BSC of context* c' *to the dimension* UC_i *of ISC, and finally maps the dimension* $WHAT$ *and* WHY *from context* c' *respectively to the dimensions* DC_2 *and* PC*. Thus, it constructs the ISC context* $[UC_i : Alice, DC_j : filetransfer, PC : Auditing]$*. The grant policy for* $Alice$ *on the particular object* $file$ *to be transferred is evaluated at this context.*

3 Threats and Safeguards

There are three sources of threat: *personal life style*, *organizational*, and *systemic*. Many institutions and governments have set up web sites to warn individuals about identity theft and how it can prevented through a disciplined life style. See [8–10] for information posted from North American and European government agencies warning individuals about identity theft and what should be done to prevent it. The two primary sources of organizational threats are employees of the organization who manage the database, and outside attackers. The first kind of threat arises when some employees use illegitimate means and violate local policies to access the information which is not legitimately required in their job related tasks. Some of the disclosures that happen inside are "accidental". As an example, the information left on the screen of a computer can be seen by another employee who is not authorized to know it. Some simple remedies include automatic log-outs whenever the system is left idle, and reminding employees about

their behavioral code. Another kind of threat is that employees who have authorized access violate the trust instituted in them. As an example, an employee may be curious to know a particular individual's date of birth or marital status. A remedy is to protect confidential data through encryption, authentication based on public key cryptography, and electronic signature. Another kind of threat is due to collaboration among a group of employees to access data on an individual, which cannot be accessed individually. Rotating employees to work in different groups, and audit trails seem the best way to deter this kind of threat. An individual who has no physical access and not an authorized user in an organization is a threat when the security perimeter of the organization is compromised. As an example, an intruder from one city might gain authorization to a financial institution in another city and obtain the identities of all millionaires in that city. Another example is when an outsider is able to infect the files in the system with virus, which make the system loose its valuable information or crash. A remedy is to strengthen fire walls, use encryption to protect the confidentiality and integrity of vital data, and safe-keeping back up tapes in encrypted form.

Systemic concerns are rooted in the procedures followed for using personal information by various agencies. Information flows across different domains, where the policies for accessing and using the information in different domains are in general different. The fact is that policies in one domain may not be known to other domains. Consequently, policies in different domains do not add up to a comprehensive global policy that will protect the privacy or the identity of the individual. As an example, the personal information of a person who is admitted to a hospital is shared by physicians, health care providers, insurers, and government agencies who provide Medicare. We contend that ESC context can be used effectively to protect personal information.

Suppose a subject s_1 in domain d_1 sends some information to a subject s_2 in domain d_2. An ESC context is constructed for this request at the periphery of the domain d_2. This context must include "the purpose" of the request. Since "purpose" is a domain information, it will be assigned a security level clearance. Consequently, the transmission channel through which the request is sent must have a security clearance higher than or equal to that assigned for the "purpose" category. Moreover, the subject s_1 should have the security clearance for sending the request, and s_2 in domain d_2 must have the security clearance to receive the request. The security level clearances within a domain is confidential to that domain, and consequently it is not possible to compare security level clearances in different domains. However, the medium used to communicate is common for both domains. Therefore it seems appropriate to assume that the security level assigned to the communication channel between the domains is known to both domains. Assume that security levels are modeled by functions slc and olc, where $slc(s)$ gives the *security level clearance* for the subject $s \in S$, and $olc(o)$ gives the *security level classification* for the object $o \in O$. Based on this premise, we impose three constraints for a secure information flow from s_1 to s_2 while sending data o along a channel α.

(1) [secure channel for object o]: $olc(o) \leq olc(\alpha)$
(2) [s_1 can write on α]: $slc(s_1) \leq olc(\alpha)$
(3) [s_2 can read, not write on α]: $slc(s_2) \geq olc(\alpha)$

4 Public Policy Framework - a proposal

A policy mentions subjects, objects (resources), roles, and suggests either directly or indirectly a sequence of actions to be done when the rule is followed. Policies govern the sets S, O, G, and R that exist across different domains. We assume that each group is associated with a distinguished subject, called *leader*, who ensures that policies are followed in all transactions engaged by the group. In a computerized system, this leader can be realized by an *agent*.

4.1 Policy Representation

A policy in every domain is a rule. An example policy is *a physician will have access to medical information on the patients under her care*. A policy, being a declarative statement, does not dictate how it should be represented in organizational databases and how it should be implemented. However we recommend that the policy representation include information on where it is applicable. A policy is represented by a rule $H \Leftarrow B$, where H is called the *head (consequent)* of the rule and B is called the *body (antecedent)* of the rule. In general, the body of a rule is a conjunction of one or more conditions; no disjunction is allowed in the body. The head of a rule, expressed declaratively, is an action specification. We associate a context condition U with each rule to suggest that the rule is applicable in any context that satisfies this condition. This is a major difference between our approach and others [3,5,4] who have developed languages for policy specification and application. By separating the context condition from the rule we achieve rule generality, and flexibility in the application of the rule.

Example 3 *Consider the rule* $U : has\_record(x, y, z) \Leftarrow patient(y, z) \wedge attends(x, y) \wedge staff(x, z)$, *where* $U = (physician(x) \wedge service(x) \geq 5) \vee head\_nurse(x) \vee secretary(x) \wedge admits(x, y, z)$ *is the context condition. The meaning of the rule is "*a staff x attending on a patient y in department z has access to the patient record within the department*", and the context condition provides the different contexts in which this rule can be applied, namely "*a physician with 5 or more years of experience is attending the patient, or a head nurse of that department, or the secretary who admits the patient in that department*".*

In general, a policy database may be large. For an efficient processing of transactions, we propose two methods to organize the rules.

Partitioning Policy Base The set BP of rules is partitioned so that each subset has policies associated with a specific domain. The function DP defines for each domain $d \in D$, the set $DP(d) \subset BP$ of rules that are relevant to that domain. Denoting the domain that is specific to the subject s by s_d, $d \in D$, the set $DP(s_d)$ gives the set of rules to be followed by s. In general, a group $g \in G$ may be responsible to deal with business transactions in more than one domain. The function RD defines for each group $g \in G$, the set $RD(g)$ of domains for business transactions. The set of rules to be followed by individual s belonging to group g, as well the rules that must be enforced by the leader of the group $g \in G$ is $BR(g) = \bigcup_{d \in RD(g)} DP(d)$. An advantage of partitioning is that each partition contains domain specific rules.

Linking Partitions For every pair of domains $d_i, d_j \in D, d_i \neq d_j$, we define a lifting function ϕ_{ij}, that associates with a rule $r \in DP(d_i)$ a subset $\phi_{ij}(r) \subset DP(d_j)$. The interpretation is that for the rule $r \in DP(d_i)$, the set of rules $\phi_{ij}(r)$ belonging to $DP(d_j)$ are *relevant*. This association is important in tracing the information flow from a source in domain d_i to a source in domain d_j. We define relevance in terms of the context condition:

A rule $r :: U : H \Leftarrow B$ is relevant to the rule $r' :: U' : H' \Leftarrow B'$ if $U' \rightarrow U$.

Informally, if the context condition U' *implies* the context condition U, then a policy enforced in a context defined by U is quite relevant in a context defined by U' and should be enforced there. Thus, every rule $r \in PB$ that is relevant to the rule $r' \in PB$ has a link in it that is directed from r' to r. When a rule changes, the policy base must be updated. However, when a rule does not change but the context of its application changes then only context condition should be changed. In addition, a change in context condition will change the links to its related rules.

4.2 Transaction Representation

An activity flows across several domains when a subject from one domain requests information from another domain. An example is shown in Figure 1. In the figure, LON, PERS, and INVS denote respectively the three domains "loan department", "personal banking department", and "investment department". The other two names EMP and CRTB in the figure denote respectively "the institution where the applicant is employed" and "the credit bureau". The bank policy will require the verification of the credentials of the client before approving the loan. The methods are invoked in a domain in response to requests (shown by $m_1, \ldots, m_8$) arriving from other domains. The message m_1, for example, triggers an action at the personal department. The policy for processing such requests will enforce an ordered flow of information, where a unit of information is computed by performing an *atomic* action. Hence, a transaction is composed of several atomic transactions. An atomic transaction does not involve any sub-activity and can be represented as (s, a), where a is an action performed by a subject $s \in S$.

The diagram in Figure 1, called a *work-flow* diagram, is modeled like a UML sequence diagram [6]. This differs from the notation used in work flow management systems [1]. Assuming that every action in a work-flow diagram is atomic we construct an expression equivalent to the work-flow diagram, and enforce policies for the execution of each action. The construction involves the composition constructs $\gg$ (sequential), $\|$ (parallel), $\circ$ (conjunction with no order), and $\diamond$ (priority). The expression $a \gg b$ defines the sequential composition of atomic actions a and b. That is, action a is executed, and using the result action b is executed. The parallel composition $a \parallel b$ is executed by simultaneous execution of actions a and b. The expression $a \circ b$ defines that action a and b should be executed by the receiver, however the order of execution is not important. The expression $a \diamond b$ defines that action a should be executed first, and if it succeeds, the action b is to be executed; otherwise, action b should be ignored and the entire action is aborted. All the constructs have the same precedence, and hence an expression

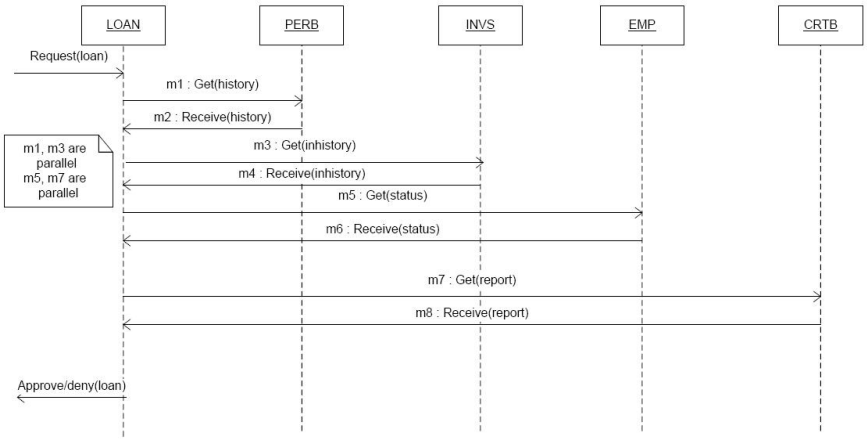

Fig. 1. Transaction - Work Flow

is evaluated from left to right. To enforce a particular order of evaluation, parenthesis may be used. Assume that each m_i in Figure 1 triggers an atomic action a_i for subject s_i. The expression constructed from the work-flow diagram is

$((s_1, a_1) \gg (s_2, a_2)) \parallel ((s_3, a_3) \gg (s_4, a_4)) \diamond ((s_5, a_5) \gg (s_6, a_6)) \diamond ((s_7, a_7) \gg (s_8, a_8))$

An activity requiring the collaboration of several individuals, or groups, necessarily involves a non-atomic action. We can use the same notation shown in Figure 1 for showing the work-flow arising from a non-atomic activity, except that actions in some domains at the work-flow diagram will be non-atomic. A non-atomic action at a domain gives rise to a new work-flow diagram. If this work-flow diagram includes a non-atomic action at a domain, we develop a new work-flow diagram for that action. We continue to develop work-flow diagrams recursively for each non-atomic action until all actions in a work-flow diagram are atomic. From each work-flow diagrams in which all actions are atomic we construct an expression. The expressions are composed in the reverse order of the decomposition of work-flow diagrams. From this discussion it is clear that it is sufficient to consider the secure transaction of an expression in which each action is atomic. Based on this discussion the work-flow expression, denoted wfe, corresponding to a work-flow diagram wf is defined

$$wfe ::= (s, a) \mid (wfe) \mid wfe \gg wfe \mid wfe \parallel wfe \mid wfe \circ wfe \mid wfe \diamond wfe$$

5 Secure Transaction

A work-flow wf is *secure* if in the specification wfe if the subject s in (s, a) is authorized to perform the action a, and the transaction is consistent with information

flow policy. From the definition of wfe it is clear that it is sufficient to discuss security conditions for the five expressions (s, a), $(s_1, a_1) \gg (s_2, a_2)$, $(s_1, a_1) \parallel (s_2, a_2)$, $(s_1, a_1) \circ (s_2, a_2)$, and $(s_1, a_1) \diamond (s_2, a_2)$.

Security Conditions for Atomic Activity (SCA) Let (s, a) be an atomic activity. Because the activity is atomic there is no loss of generality in assuming that the subject s belongs to the domain in which the activity a arises. The subject s can be an individual, or play a role, or a member of a group.
Requester Authorization The identity of the requester who triggers action a must be proved by subject s and authorized to receive the requested service. The context c in which the request is initiated includes the credentials of the requester, in addition to those mentioned in the definition of ESC.

1. [Rules for domain d:] The set PB_s of rules in PB that s is allowed to follow in domain d is:
$$PB_s = \begin{cases} DP(s_d) & \text{if s is an individual} \\ \bigcup_{s \in g} DP(s) & \text{if s is in group g} \\ \bigcup_{x \in R} DP(x) & \text{s plays the roles R} \end{cases}$$
2. [Select the rules that are relevant for context c:] For each rule $r \in PB_s, r :: (U : H \Leftarrow H)$, evaluate the context condition U at context c. This is done by substituting the values in c that are appropriate for the predicates in U. If U evaluates to true at c then the rule r is relevant for context c. The set $PB'_s = \{r \in PB_s \mid r\ is\ relevant\}$ is the set of relevant rules that s can use in context c.
3. [Remove rules not useful for the credentials:]: It is sufficient to use those rules in each of which the body of the rule is satisfied by the credential, which is part of context c. The set $PB''_s = \{r \in PB'_s \mid credential\ satisfies\ B\}$. If the set $PB''_s \neq \emptyset$, the requester is validated. However, if $PB''_s = \emptyset$ the credential cannot satisfy the body of any rule and hence the requester's identity is not established. The expression $PB@c$ (see section 2.3) should be interpreted in this manner.
4. [Validate the requested service a:] If there exists a rule $r, r \in PB''_s$, and the head of the rule H *implies* the action a, then the request of the requester is valid; otherwise, the request is not valid.

Secure Execution If the validation is successful, the grant policy discussed in Section 2.3 is enforced.

Security Condition for Non-atomic Executions We discuss security conditions for the execution of expressions $(s_1, a_1) \gg (s_2, a_2), (s_1, a_1) \parallel (s_2, a_2), (s_1, a_1) \circ (s_2, a_2)$, and $(s_1, a_1) \diamond (s_2, a_2)$. These expressions respectively represent *explicit data flow*, *absence of data flow*, *implicit data flow*, and *conditional data flow*.

1. [explicit data flow] $(s_1, a_1) \gg (s_2, a_2)$: A request that triggers this compound transaction, will trigger (s_1, a_1). This being an atomic action, the security condition SCA will be applied. If action a_1 is successfully fulfilled, then subject s_1 must obey *obligations* attached to this action. An example obligation is "inform the user

if her personal information is modified". An obligation rule is independent of rules that enforce security within the system. The result of executing action a_1 must be communicated along a secure channel to subject s_2, provided the information flow policy discussed in Section 3 allows such a communication. The action (s_2, a_2) being atomic, the security condition SCA will be applied. If action a_2 is successfully fulfilled then subject s_2 must obey obligations attached to action a_2.

2. [absence of data flow] $(s_1, a_1) \parallel (s_2, a_2)$: A request that triggers this compound transaction, will trigger (s_1, a_1) and (s_2, a_2) simultaneously and independently. That is, one subject is not aware of the other subject involved in the transaction. Hence there is no information flow between s_1 and s_2. For each atomic action, the security condition SCA will be applied. Notice that the requestor who triggers this compound action will have to be authorized independently by s_1 and s_2. If action a_i is successfully fulfilled, then subject s_i must obey obligations attached to this action. It is possible that one of the actions fails while the other succeeds.
3. [implicit data flow] $(s_1, a_1) \circ (s_2, a_2)$: If $s_1 = s_2$, then this compound action is necessarily sequential. The authorization procedure is as in "explicit data flow". If $s_1 \neq s_2$, then the requester will trigger both subjects, and will have to be identified independently by both subjects. Different scenarios arise: (1) subject s_1 completes action a_1 and communicates the result to s_2; (2) subject s_1 determines that some information from s_2 is required to complete action a_1. The two other scenarios are mirror images of the above two scenarios. These scenarios are mutually exclusive, and hence only one will occur. The one that occurs, is a sequential composition, with or without data sharing. Hence, the information flow is implicit, not visible to the user. Security conditions discussed in "explicit data flow" are sufficient to be applied here.
4. [conditional data flow] $(s_1, a_1) \diamond (s_2, a_2)$: A request that triggers this compound transaction, will trigger (s_1, a_1). This being an atomic action, the security condition SCA will be applied. If action a_1 is successfully fulfilled, then subject s_1 must obey obligations attached to this action and rest of the procedure is as in sequential data flow. If (s_1, a_1) fails either because of authorization failure or the inability of internal security policy to fulfill the request, the entire transaction is abandoned.

6 Conclusion

Protecting the identity of an individual and at the same time validating the credentials submitted by an individual for services at organizational levels are important issues. An individual should exercise caution in every day life to protect the certificates that establish her identity. At the organizational level, there is a dual responsibility - ability to authenticate clients for services and protect the confidentiality of data concerning individuals. Without authentication of clients, information get stolen and benefits may reach wrong persons. Without confidentiality, personal information of individuals may be revealed violating the fundamental right of an individual to privacy. At the same time, neither of these can be administered in an extremely harsh manner: genuine people get offended when a routine identity check is strictly enforced and certain sectors may not function without getting useful information from other sectors. Balancing these apparently conflicting needs is a challenge. In this paper we have thoroughly analyzed the

sources of identity threats, and investigated some solutions to thwart it. The solutions that we have put forth need to be evaluated in a practical setting, at least by simulating different scenarios. It is our intent to do that in future.

References

1. E. Bertino, E. Ferrari, V. Atluri, "A Flexible Model for the Specification and Enforcement of Role-Based Authorizations in Workflow Management Systems," In *Proceedings of the 2nd ACM Workshop on Role-Based Access Control* (RBAC-97), ACM Press, New York, 1997, pp. 6-7.
2. S. Claußand M. Köhntopp. *Identity management and its support of multilateral security*. Computer Networks, **37** (2001), 205–219.
3. N. Damianou, N. Dulay, E. Lupu, and M. Solomon. *The Ponder Policy Specification Language*. Proceedings Policy 2001: Workshop on Policies for Distributed Systems and Networks, Bristol, UK, 29–31, Jan. 2001.
4. J. DeTreville. *Binder, a logic-based security language*. Proceedings of the 2002 IEEE Symposium on Security and Privacy, IEEE Computer Society Press, May 2002, 105-113.
5. R. Ortalo. *A Flexible Method for Information System Security Policy Specification*. Proceedings of 5th European Symposium on Research in Computer Security, 1998. Louvain-la-Neuve, Belgium, Springer-Verlag.
6. J. Rumbaugh, et al: The Unified Modeling Language Reference Manual, Addison-Wesley.
7. Kaiyu Wan, Vasu Alagar. *Security Contexts in Autonomic Computing Systems*. In *Proceedings of Proceedings of 2006 International Conference on Computational Intelligence and Security (CIS2006)*, November 03-06, 2006, Guangzhou, PRC, page 1523-1527. (also to appear in Lecture Notes in Artificial Intelligence)
8. Fighting Back Against Identity Theft - U.S. Federal Trade Commission. www.ftc.gov/bcp/edu/microsites/idtheft/
9. Identity Theft: What is it and What you can do about it?, Office of the Privacy Commissioner of Canada, www.privcom.gc.ca/fs-fi/02_05_d_10_e.asp/
10. European Conference "Maintaining the integrity of identities and payments: Two challenges for fraud prevention". ec.europa.eu/justice_home/news/information_dossiers/conference_integrity/index_en.htm/