Access control in Multi-Organization Environment (MOE) is a critical issue. Several access control models need some improvements to be used in such environment where the behaviors of the users and the organizations may change during the collaboration. In similar environments, the use of trust management with an access control model is recommended. It is worth to mention that trust allow us to increase the security of the system, due to the fact it is dynamic, so it let us represent at runtime changes in the behavior of the players.

In this paper, we aim to define trust of users and organizations in MOE. In these models, the initial valuation of the trust is crucial, in the sense that all the trust framework will depend on these initial values. In particular, in MOE these values should be computed using not only the experience of the users with an organization but also the experience of an organization with another organizations. However, the combination of these values normally is not an easy task. Therefore, in this paper we propose an algorithm that uses fuzzy logic to manage this information. In particular, this methodology allows us to extract the initial set of trust values of a MOE taking into account past experiences.

I. INTRODUCTION

Collaboration between organizations forms a very active research area where industrial and academic companies are interested. In this environment, any participant may obtain several advantages such as: (1) the ability to use extern and professional resources, services and knowledge, (2) the reduction of the timing intervention and (3) the gain of the experience of some experts. However, several problems could be present in this Multi-Organization environment (MOE). For instance, the integration and the definition of a trust model in order to provide a dynamic access control of the resources is one of the critical issues. This concept allows to have a dynamic access control that is close to the human interactions. In fact, this access will be firstly based on some attributes as the role, the age, the company, etc. However, the interactions with the a user will be considered as a good or bad behaviors in order to punish or encourage this trustee. This process should be represented with a trust model.

In our framework we present a trust model that conforms to the MOE requirements. In particular, the trust is defined as a solution to guarantee a certain level of security between the user and the resource provider (truster). This relationship is restricted to a situation and a time. Therefore a user can be trusted to read some files and distrusted for others. Furthermore, our model is dynamic since it depends on time. We believe that a trust is a relationship associated with a trust vector that contains two parameters: the experience evaluation of the user and his organization. This definition permits to introduce a dynamic evaluation of the trust of an organization and its influence regarding the rights of their users.

The experience evaluation depends on a special event noted behavior, that is an interaction between a user and the trustee. This notion is characterized by a situation and an appearance time. In this paper we discuss and define some strategies that can be used with our framework in order to evaluate a behavior. These information will be combined in order to have one value. This process is done with fuzzy logic concept. Finally, we present our algorithms that permits to determine the experience values from a trace file.

The rest of the paper is structured as follows. Section II presents some related work. In Section III some preliminaries concepts are introduced. Next, in Section IV the trust model notions experience, satisfactory strategies and fuzzy logic are introduced. Moreover, this section defines the two basic algorithms that evaluate the experience based on a trace file. Following, in Section V a case study is presented to highlight some advantage of our model. Finally, in Section VI we present the conclusions and some lines of future work.

II. RELATED WORK

Several trust models are used in a similar distributed system.

In [4], [13], authors aim firstly at providing a clear and general definition of trust vector, and then they propose a mathematical evaluation of different parameters of this vector. Moreover, in [3] the relationships between different contexts is formalized using a context graph in order to extrapolate values from related contexts to approximate the trust of an entity even when all the information needed to calculate the trust is not available. These solutions are very interesting. However, they do not evaluate the trust level of an O-grantee and its impact on the trust of the users. Therefore, their definition and evaluation are not compatible with MOE requirements where an organization cannot have the same trust level regarding the other organizations of the MOE. In addition, the trust level of
an organization can be changed during collaboration that is not studied in these propositions.

In [9] the researchers propose a method to evaluate the trust of an organization before the collaboration. They define a method to evaluate the maturity level of the security practices deployed within the organization. The security practices are provided by the ISO/IEC 17799 security standard. This solution aims at establishing a chain of trust. However, this solution is static. It does not take into account the evolution of the trust level of a user or an organization. The study of these works concludes that trust needs more investigation to be adapted to the requirements of this special distributed system.

III. PRELIMINARIES

A Multi Organizational Environment, in short MOE, is a structure that contains at least two organizations. These organizations are O-grantor and O-grantee. The O-grantor is the participant that offers resources. These resources are accessed by users of another organization called the O-grantee. An organization may be an O-grantor and O-grantee at the same time.

The sharing of resources is based on some restriction rules that forms an interoperability security policy in order to control the access of these resources. This policy is completely parameterized by the organization.

**Definition 1:** An organization is an entity that is organized in different groups of active entities, such as subjects, playing some role or others. The set of all organizations will be denoted by Organizations.

In this paper, the organizations will be ranged over orgA, orgB,... An important notion in several models [3], [1], [7], [11] is the idea to classify entities in concrete and abstract. A model with high level of abstraction offers a more independent security policy. In an informal way, when we refer to concrete we think about single units, or elements, while we refer to abstract then we think about global units, or sets.

In MOE there are three concrete and their respective abstract entities. A subject is abstracted into a role which is a set of users to which the same security rule apply. Similarly, an activity and a view are respectively a group of actions and objects to which the same security rule is applied.

Our solution aims to define trust valuation of the different entities of MOE to provide, dynamically a mapping function between subject and roles.

The concept of rule allows us in MOE to represent the different rights that are given by using abstract notation.

**Definition 2:** A rule is a tuple \( (ac, a, v) \). Where \( ac \in Access \) is the type of access for this rule that (e.g. permission, interdiction), \( a \in Activities \) is the activity to perform, and \( v \in Views \) refers to the shared resource. The set of all rules will be denoted by Rules.

An internal politic of rules is defined by the following function, that assigns to each role a set of rules

\[
    f_p : \text{Roles} \longrightarrow \varphi(\text{Rules})
\]

Finally, the classical MOE request-scheme model, presented in OrBAC [3] and its derivatives as Poly-OrBAC [7] and O2O [2], is based on two parts: the subject and the Orbac Model. On the one hand, a user can send several requests at the same time to an O-grantor in order to access to a shared resource. So, there are several requests, each one identified by a unique id, and these can be collected into logs. On the other hand, for each request, some attributes are collected by the O-grantor in order to offer the required service. In particular, it is identified the subject that makes the request. The different attributes permits to define a possible roles \( \{r_1, \ldots , r_n\} \) where this entity is mapped. After that, it is computed the kind of access \( a \), the activity \( act \), and the view \( v \) that are related to the request.

Finally, it evaluates if there exists a rule for any role \( r \in \{r_1, \ldots , r_n\} \). If this rule permits the access to this view, then the verdict of this request is ACCEPT, otherwise DENY.

In the following, we define our trust model that will permit to define a dynamic mapping function between subject and roles based on the recent interactions.

IV. TRUST WITH FUZZY LOGIC

**Trust** is a relation between two entities a truster and a trustee. The truster that is the O-grantor in MOE scenarios, relies on offering access to a specific resource for the trustee, i.e, the O-grantee and/or the user. This relationship is always restricted to a situation and a time value, that is, the time of the trust evaluation [10], [13], [15].

**Definition 3:** We define a situation as a tuple \( (a, v) \) where \( a \in Activities \) is an activity and \( v \in Views \) is a view. The set of all possible situations will be denoted by Situations.

Regarding time parameter, we say that \( T_i = [t_i, t_{i+1}] \) with \( i \in \mathbb{N} \), is a time interval if \( t_{i+1} \in \mathbb{R}_+ \), \( t_i \in \mathbb{R}_+ \), \( t_i < t_{i+1} \), and \( t_{i+1} - t_i = dt \), where \( dt \in \mathbb{R}_+ \) is a fixed duration. We consider that \( TR \) denotes the set of time intervals. In this paper we consider that the fixed duration is similar in all time invariants.

We will denote by \( T_0 \) the initial time interval, and by \( T_{n+1} \) the last one, also called the evaluation time.

In our framework, any trust relationship between a truster \( e_1 \) and a trustee \( e_2 \) related to situation \( s \) and a time period \( T_i \) will be denoted by

\[
    \text{trust}(e_1 \rightarrow e_2)_{T_i}^s
\]

This relation will be associated to a trust vector that is a couple of values \( (Exp_{user}, Exp_{org}) \) where \( Exp_{user} \) and \( Exp_{org} \) are the experience evaluation of the user and its organization respectively.

**Experience** is a process which aims to establish wisdom on making decision. It is based on the evaluation of the previous interactions between the trustee and the truster related on a specific situation at a period of time.

In our social life, any person will rely on this parameter to trust another entity and to decide whether to continue or not the collaboration. In our framework, there are considered two types of experiences. On the one hand the experience of the trustee organization that takes into consideration the previous behaviors of all users of this organization, that is, organization...
to organization experience. On the other hand the direct experience where only the previous behaviors between this user and the truster are considered, that is, user to organization experience.

**Definition 4:** Let $\mathcal{U}$ be the set of users in a MOE scenario. We define a behavior as a tuple $(s, T_a)$ where $s \in \text{Situations}$ is the situation and $T_a \in \mathbb{R}_+$ is the time interval of this behavior. For any $u$ in $\mathcal{U}$ we denote by $B_u$ the set of behaviors of $u$ related to the situation $s$ and period $T_a$ and by $B$ the set of all behaviors.

In order to evaluate the experience of the trustee, we have to define a satisfactory evaluation method. This concept is detailed on the following.

A. Satisfactory evaluation strategies

This satisfactory evaluation method aims to assess each behavior $b$. This assessment associates a value between $[-1,1]$ to this behavior, it will be denoted $\text{Sat}(b)$. If $\text{Sat}(b) \in [0,1]$ then this means that the previous behavior does not respect some requirements and it is considered as a bad one that have to decrease the experience evaluation of the user. Otherwise $b$ is considered as a good behavior that will increase the experience evaluation of the user. As it is introduced in the Figure 1, the reception of a request needs the consultation of the interoperability security policy in order to collect the different attributes. This process is more detailed in several architectures [5]. Then, these informations are sent to a decision engine that uses the security policy and a configuration file in order to deny or permit the access to a resource. The configuration file contains the trust vector of the different users. If the request is permitted, the related behavior will be evaluated. This will modify the configuration parameter of the user and his organization. This modification may cause a deny response of the same request at next time.

In MOE, we aim to propose some techniques that can be used to evaluate a behavior.

1) **Recommendation strategy** Access control Based on OrBAC [3], [8], [9], [7] permits to write some recommendation Rule with the following predicate:

\[ \text{Recomx(Any2OrgA, role, activity, view, context)} \]

This rule is defined as an authorization. It includes some desirable or advisable events. If they are not done, they abstract the responsibility of the user. Therefore, we propose to use this predicate in order to define a list of rules that will be named recommendation policy. For each situation some recommendation policies with new parameters (views, activities,..) are defined and their verification is done during the execution of the behavior.

**Example 1:** In an MOE scenario, we may have a situation $\text{Edit\_Source}$. If an employee will be permitted to perform this situation, he will be able to write code and description or copy several files of a projects. A recommendation policy may contain some rules to check, for instance a) some files are edited before others, b) critical bugs are corrected before the normal one, or c) the time needed to execute some actions does not exceed some limits.

2) **Modification Assessment Strategy.** This strategy aims to evaluate the modification of the resource. It is done after the execution of the behavior, i.e, in the context of programming, we have to evaluate the syntax of the source code after each modification. This strategy cannot evaluate the behaviors that does not change any data as validation or reading. As a result, the administrator has to define some resource reference to be compared with the modified one.

**Example 2:** For the same situation $\text{Edit\_Source}$, we suppose that any modification of a file will contain firstly the name of the author, the date and the title of the modification. An evaluation system have to check these information to evaluate a behavior related to this situation.

3) **Hybrid Strategy.** This strategy is based on the two previous solutions. with this strategy, we have three cases related to one situation. These are a) those that can be evaluated with recommendation and modification strategies, b) other that can be associated to only one method c) the third set could not be evaluated.

The Figure 2 illustrates these previous strategies.

B. Satisfactory evaluation with fuzzy logic

We do not assume that the reader is familiar with the concepts of fuzzy logic. Therefore, we present some of its
basic concepts: *fuzzy relations* and *triangular norms*.

Fuzzy relations play an important role in fuzzy modeling, fuzzy diagnosis, and fuzzy control. In ordinary logic, a *set* or a *relation* is determined by its characteristic function, that is, a function that returns *True* if the element is in the set (or if some elements are related) and *False* otherwise. Working with a fuzzy framework, we do not have that clear distinction between truth and falsehood. Instead of this, we have a complete range of values in the interval [0,1]. The larger this value, the more confidence we have in the assessment.

In this paper we only consider relations over the set of real numbers $\mathbb{R}$. Thus, a fuzzy relation is a mapping from the Cartesian product $\mathbb{R}^n$ into the interval $[0,1]$.

**Definition 5:** A fuzzy relation $A : \mathbb{R}^n \to [0,1]$ is a function that provides the confidence of an element in $\mathbb{R}^n$.

Let $x \in \mathbb{R}^n$. We say that $x$ is *not included* in $A$ if $A(x) = 0$. Consequently, we say that $x$ is *fully included* in $A$ if $A(x) = 1$. The kernel of $A$ is the set of elements that are fully included in $A$.

A triangular norm (abbreviated $t$-norm) is a binary operation used in fuzzy logic to generalize the conjunction in propositional logic. In order to generalize conjunction, we have to analyze its basic properties: commutativity $p \land q = q \land p$, associativity $(p \land q) \land r = p \land (q \land r)$, identity $\text{True} \land p = p$, and nilpotency $\text{True} \land p = \text{False}$.

Therefore, we require a $t$-norm to satisfy similar properties.

We also require an extra property: monotonicity. Intuitively, the resulting truth value does not decrease if the truth values of the arguments increase.

**Definition 6:** A $t$-norm is a function $T : [0,1] \times [0,1] \to [0,1]$ which satisfies the following properties:

1. Commutativity. $T(x,y) = T(y,x)$.
2. Monotonicity. $T(x,y) \leq T(z,u)$ if we have both $x \leq z$ and $y \leq u$.
3. Associativity. $T(T(x,y),z) = T(x,T(y,z))$.
4. Neutrality of 1. $T(x,1) = x$.
5. Nilpotence of 0. $T(x,0) = 0$.

Let us remark that the property $T(x,0) = 0$ can be easily deduced from the others. Since $t$-norms have the associativity property, we can generalize them to lists of numbers. Therefore, $T(x_1, x_2, \ldots, x_{n-1}, x_n)$ is a shorthand for $T(x_1, T(x_2, \ldots, T(x_{n-1}, x_n)) \ldots)$.

**Example 3:** We continue with the same situation.

---

**Edit Source** with hybrid strategy. The goal of the use of the fuzzy logic is to combine the different constraints formed by the proposed strategies in order to provide only one value between [0,1]. Since the use of the binary decision is inadequate with our system. A simple example that presents some properties related to the situation **Edit Source** is introduced in the Figure 4. The two first rules are based on recommendation policy, they are verified during the execution, some alert may be established. For the rest, they have to be executed after the execution of event. In this example, the $t$-norm function can be a multiplication operation ($\ast$). The definition of the questionnaire and the $t$-norm function have to be realized by an expert.

---

**C. Trace file**

As it is described in the Figure 5 this satisfactory evaluation will be combined with the related request and some basic information to define a trace file that can be used to determine the experience evaluation of the user and the organization. Indeed, this trace will contain all the information that we need to evaluate the experience. In our model, a line in a trace is formed by the request and the subject identities (Req_id,subject), the organization, the related situation, the timestamp and the satisfactory evaluation of the behavior. The analysis of this trace will permits to define the new trust vector of the different users.

Next we present how to evaluate the experience parameter. We denote by $Exp(u)_{T_n}$ the experience evaluated by a trusting **OrgA** for a user $u$, related to a specific set of behaviors $B$, a situation $s$ at time interval $T_n$. Its evaluation depends on the satisfactory evaluation of all the behaviors of the user $u$ related to the same situation that previously happened, that is in $T_i$
The first one, \texttt{ComputeUser}, computes the experience value of the users with respect to a situation. The second one, \texttt{ComputeOrganization}, computes the experience of the organization. The following two algorithms, presented in Figures 6 and 7, detail these processes.

The first one computes the experience value of the users, while the second one computes the experience of the organization. The first algorithm receives the trace, an user \( u \), a situation \( s \), the \( \text{t-norme} \) to apply and a level of confidence. It transverse all the elements of the trace collecting all that the fuzzy values for the user \( u \) and the situation \( s \) in the variable \( \delta \). Then, the algorithm will return the \( \text{t-norme} \) result associated with these collected values and the input degree of confidence. The second one computes the experience of the organization. This second algorithm firstly computes the trust value associated with the employees of the organization by performing the first algorithm. Then, it returns the average of the previous values.

As it is described in the Figure 7, these experience evaluations will be used as a new configuration related to the user and the organization. In our solution, each user will be mapped to a role based on the collected attributes and his trust vector. This means that a user can be assigned to a role based on the collected attributes and his trust vector. The same user may regain this access by performing the first algorithm. Then, it returns the average of these values. The following two algorithms, presented in Figures 6 and 7, detail these processes.

V. CASE STUDY

Let us consider three organizations \( \text{org}_A, \text{org}_B \) and \( \text{org}_C \). The first one, \( \text{org}_A \) plays the role \text{O-grantor}. We have defined a \text{role engineer} that is permitted to read, edit and comment any source project. An extract of the interoperability security policy of the \( \text{org}_A \) related to this role engineer is following presented:

\begin{verbatim}
ComputeUser(tr, u, s, T, \alpha) :
Data: Trace tr, user u, situation s, t-norme T and confidence \alpha
Result: experience of the user u related to the situation s at the period T
(\text{** Parameters initialization **})
line = last_evaluation_period(trace);
(\text{** Main loop **})
foreach line \in trace do
    if line.situation == s \&\& line.user == u then
        collectFuzzyNumber(line.period, line.Sat(b), \delta);
    end
end
return T(\delta, \alpha);
\end{verbatim}

\begin{verbatim}
ComputeOrganization(trace, org_B, T, \alpha, \hat{T}_n) :
Data: Trace trace, organization org_B, t-norme T, confidence \alpha, and time of the evaluation \hat{T}_n
Result: Experience of the organization org_B related to the situation s
(\text{** Main loop **})
forall the For any user u of org_B that have interacted with the truster in the situation s do
    if the experience of u related to s is not available then
        collect(ComputeUser(tr, u, s, T, \alpha), Exp_Users);
    end
end
return average(Exp_Users, org_B, s, \hat{T}_n);
\end{verbatim}

\begin{table}[h]
\centering
\begin{tabular}{|c|c|c|c|c|}
\hline
Resp_id & subject & organization & situation & period & Sat(b) \\
\hline
2117 & sub1C & org_C & s1 & T_1 & -0.6 \\
7725 & sub1C & org_C & s1 & T_1 & -0.6 \\
7869 & sub1C & org_C & s1 & T_1 & -0.6 \\
78_ & sub1C & org_C & s1 & T_1 & -0.6 \\
78_ & sub1C & org_C & s2 & T_1 & 0.8 \\
78_ & sub1C & org_C & s2 & T_1 & 0.8 \\
78_ & sub2C & org_C & s1 & T_1 & 0.5 \\
78_ & sub2C & org_C & s1 & T_1 & 0.4 \\
78_ & sub3C & org_C & s1 & T_1 & 0.45 \\
78_ & sub1B & org_B & s2 & T_1 & 0.8 \\
78_ & sub1B & org_B & s2 & T_1 & 0.65 \\
78_ & sub1B & org_B & s1 & T_1 & 0.9 \\
78_ & sub2B & org_B & s1 & T_1 & 0.8 \\
78_ & sub2B & org_B & s1 & T_1 & 0.8 \\
\hline
\end{tabular}
\caption{Log file.}
\end{table}

- Permission(Engineer, read, Source)
- Permission(Engineer, edit, Source)
- Permission(Engineer, comment, Source)

Let us note that in 6, 5 any request sent by a user have to be accompanied with some attributes to be assigned to this role. In our proposal, in addition to this information, this assignment depends on the situation and the experience of the user and his organization. Therefore a trust policy like the following one is needed:

- An Engineer can edit any resource data, only if:
  1) \( \text{exp}(u) \geq -0.3 \) and \( \text{exp}(org) \geq 0.2 \) or
  2) \( \text{exp}(u) \geq -0.7 \) and \( \text{exp}(org) \geq 0.6 \)
- An Engineer can commit any resource data, only if:
  1) \( \text{exp}(u) \geq -0.3 \) and \( \text{exp}(org) > 0 \) or

The Figure 8 presents an extract of the trace after being filtered and parsed by our method. Each line contains the user, his organization, the situation, the period and the satisfactory evaluation of the behavior.
use the fuzzy logic in order to combine the different values resulting from the satisfactory evaluation. This work is also enhanced with a simple example to highlight the advantages of our work. For the best of our knowledge, this model is the first where the control access of external users takes into account the dynamic behaviors of their organization.

As future work, we are planning to integrate this mechanism with probabilistic information regarding the input parameters of the behaviors.

**References**


