Apply Measurable Risk to Strengthen Security of a Role-based Delegation supporting Workflow System

Weili Han[1,2], Qun Ni[2], Hong Chen[2]
[1]*Software School, Fudan University, Shanghai, China*
[2]*Department of Computer Science, Purdue University, West Lafayette, Indiana, USA*
wlhan@fudan.edu.cn, {hanw, ni, chen131}@cs.purdue.edu

Abstract

Workflow systems often use delegation to enhance the flexibility of authorization. However, using delegation also weakens security because users may have difficulties understand and design correct delegation policies. In this paper, we propose the Measurable Risk Adaptive Role-based Delegation (MRARD) framework to address this problem. MRARD employs measurable risk for SSOs (System Security Officers) to provide a complementary protection mechanism in role-based delegation supporting workflow systems. In MRARD, when another enterprise user wants to use a delegated role to execute a task, a fuzzy logic based inference processor will infer the risk_level. Based on simple risk adaptive decision policies, a decision module will determine whether the access should be granted under a certain risk mitigation action.

1. Introduction

The application of measurable risk is recently recognized as an effective approach to access control in dynamic and emergent application scenarios[1][2][3]. In this paper, we propose a novel framework, called Measurable Risk Adaptive Role-based Delegation (*MRARD*), to strengthen the security of a role-based delegation supporting workflow system.

Delegation[4][5] is a flexible mechanism in an enterprise, which ensures that business processes are executable when a key user temporarily leaves his/her position. However, A delegation may assign a security administrative task to an enterprise user who is not well-trained in security administration. Thus, *it is difficult for the enterprise user who is not the SSO to understand and design traditional delegation policies.*

Although SSOs can define high level policies and rules to avoid low-level mistakes committed by a delegator[4][5], the traditional access control model[6] and the delegation model[4] are still too rigid in dealing with risky delegations. In traditional methods of role-based access control and role-based delegation, every one has the same priority when he/she has qualified roles or attributes according to the delegation policies. Finer-grained rules and policies can reduce the probability of risky delegations, but are hard to be well-defined due to their complexity and size. The SSOs are, therefore, falling into a dilemma where they must trade off between keeping the business processes executable and enforcing the *Principle of Least Privilege*. On the one hand, if the SSOs make less strict rules and policies which allow more users to accept delegation, the delegator might maliciously or accidentally assign a role to an unsuitable user when the SSOs are unaware until a serious consequence happens. On the other hand, if the SSOs make finer rules and policies to limit who can receive delegation, the workflow instance might not be executable due to absence of a key user.

Now, a measurable risk adaptive method provides an alternative and flexible way to protect an information system by identifying risk and mitigating risk [1][2][7]. However, current research works about measurable (or quantitative) risk[1][7] can not be applied directly to an enterprise-oriented role based application.

This paper, therefore, proposes a novel access control framework named *MRARD* for a role-based delegation supporting workflow system. *MRARD* explicitly applies measurable risk as an indicator to strengthen security of a role-based delegation supporting workflow system. When a user wants to use a delegated role to execute a task, the system will evaluate the risk according to relevant fuzzy risk evaluation policies which are predefined by the SSOs. After risk evaluation, the system can determine whether the execution is allowed according to risk-adaptive decision policies, the risk and a risk mitigation action.

The contributions of this paper are as follows:

- We propose a novel measurable risk adaptive framework, *MRARD*, which explicitly evaluates the risk_level in a role-based delegation supporting workflow system, and strengthens the security of role-based delegation;
- We leverage fuzzy set and fuzzy logic to deal with the multi-value problem which is a major challenge during evaluating risk_level.

The rest of the paper is organized as follows: section 2 introduces the framework; section 3 introduces the background knowledge; section 4 introduces *MRARD*; section 5

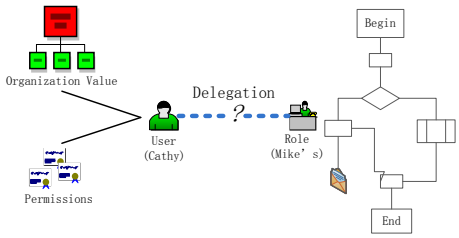

Figure 1. A role delegation supporting workflow system

leverages fuzzy set to define a formal fuzzy risk mitigation language, and introduces how to evaluate risk_level by the theory of fuzzy logic. In section 5, we also introduce a simulated case which explains how the risk evaluation works; section 6 describes how to make a risk adaptive decision; section 7 analyzes the efficiency of *MRARD*; section 8 discusses the related works; the final section summarizes the paper and discusses future work.

2. A Risk Adaptive Access Control Framework

We follow the three guiding principles in JASON Report[2] to design a measurable risk framework that evaluates risk_level and determines whether the risk_level is acceptable under a certain risk mitigation action. The framework includes four steps:

- **Risk Components Measurement:** SSOs specify which risk components are the main ones, and specify their respective measurement method;
- **Risk_Level Evaluation:** the system will evaluate how much risk there is according to the previous specified risk components. This step is generally complex because it aggregates the risk components which are not *conditionally independent*, but connected with each other. It is hard to use a simple formula to get a precise value. But we use ten different risk levels from risk_level 0 to risk_level 9, instead of the precise value.
- **Define policies of acceptable risk_level:** SSOs define which risk_level can be acceptable when the delegation happens. The policies can either be applied to all delegation in a system, or depend on concrete delegation respectively.
- **Determine whether the risk_level is acceptable under a certain risk mitigation action:** A risk aware request is allowed according to evaluated risk_level and risk mitigation action. A risk mitigation action could reduce the degree of risk_level.

3. Background

3.1. Risk Measurement

Risk is defined as the *expected value of damages*[7] or *the possibility of loss or injury* in the Merriam-Webster dictionary[1]. But due to the complexity of risk, how to measure risk is variable in different domains. For example, Value at Risk (VaR)[8] was defined as a specified probability and a specified time horizon for a specific portfolio of financial assets. VaR is a risk measurement based on historic data. However, in the access control field, there are not enough log data that can be used to measure the risk[8].

Therefore, research in the access control field started to explore risk measurement recently. Cheng et al. [1]proposed a risk measurement method based on *temptation* and *inadvertent disclosure*. And Cheng et al. tried to use both the attributes (security level and category) of subjects and objects to calculate the probability of expected loss. However, in major commercial information systems, the attributes security level and category are usually absent, and the access control policies are more complex than those in a Multi-Level Security system.

Other researchers did some work on risk measurement by using fuzzy logic[9]and Bayesian networks [10]. They studied how to evaluate risk of asserts in an enterprise. But we need further study on risk in a commercial access control system, because risk is variable in different domains and the above method can not work in a role-based delegation supporting workflow system.

This paper leverages fuzzy set and fuzzy logic to evaluate final risk_level rather than calculate a precise value from a formula[1] or a Bayesian network[10].

3.2. Role-Based Access Control and Role-Based Delegation

Role-based access control is a popular access control model in the recent decades[6]. A comprehensive introduction of general Hierarchy Role-Based Access Control model is given in[6]. Figure 2 shows an instance of role hierarchy and a user-role configuration.

Role-based delegation was defined as a process whereby an active entity in a distributed environment authorizes another entity to access resources[4]. Thus, Li et al.[5] defined a trust-management method to deal with the delegation in a distributed environment. And Zhang et al.[4] defined a role-based delegation and revocation model to manage role-based delegation.

3.3. Workflow System

Delegation discussed in this paper is combined with a workflow system[11], because a workflow system is a

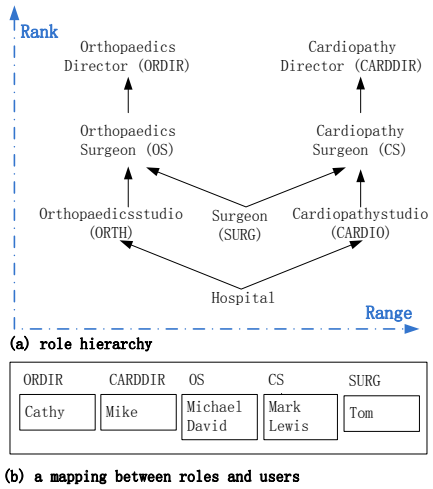

Figure 2. Role Hierarch and User-Role Assignment

popular and basic system in an enterprise. Furthermore, delegation can improve flexibility of authorization in a workflow system when a key user temporarily leaves his/her position.

Generally, a workflow system consists of three parts: workflow designs, workflow instances and a workflow engine. Each workflow design consists of activities, activities' dependency relationship and other properties. Here, we make a simplified workflow definition where a workflow design consists of a graph and one specified property.

Definition 1: **workflow**

$$workflow = \langle Activities, \rightarrow, Workflow\_Value \rangle$$

Here, *Activities* refers to all activity entities. Definition 2 makes a formal definition of *Activities*; $\rightarrow$ is a partial order relationship among Activities. $\forall act1, act2 : act1 \rightarrow act2$ means the starting of act2 directly depends on the accomplishment of act1; *Workflow_Value* refers to refers to the importance degree of a workflow instance in an enterprise. Workflow_Value is specified as a percentage in the workflow runtime phase. For example, Workflow_Value of a Reimbursement workflow instance is 30%, and Workflow_Value of a Fire workflow instance is 70%. During workflow designing, Workflow_Value is usually variable according to other workflow attributes. For example, Workflow_Value of a Reimbursement workflow instance is variable according to its amount of money.

Definition 2: **Activities**

$$Activities \subseteq Tasks \times 2^R$$

Here, Tasks refers to all tasks and R refers to all roles in the information system.

4. Risk Vector in MRARD

Definition 3: **Measurable Risk**

$$Measurable\_Risk = \langle Rank\_Diff, Range\_Diff, Workflow\_Value \rangle$$

In *MRARD*, we define measurable risk as a vector where each component is measurable. Each measurable risk vector consists of three components: **Rank_Diff**, **Range_Diff**, **Workflow_Value**.

Risk in delegation partly arises when a user wants to use a delegated powerful role to execute a task. And this part of risk includes two components: **Rank_Diff**, **Range_Diff**. Because a powerful role generally has some sub-roles, the user could activate different sub-roles (we note the executor's role as an *active role*) to execute different tasks. As a result, different *active roles* will bring up different risks:

- **Rank_Diff:** **Rank_Diff** is measured by Rank_Difference between an *active role* and a user. The ranks of roles come from SSO's assignment, when the ranks of users are derived from ranks of roles. As is shown in Figure 2, a structure of role hierarchy actually is a two-dimension diagram. In the vertical direction, there is an implicit rank which is similar to the *security level* in a Multi-Level Security system. For example, in Figure 2, the rank of *CARDDIR* is higher than the rank of any other role which is directly below *CARDDIR*, such as *CS*;
- **Range_Diff: Range_Diff** is measured by permission difference between an *active role* and a user. In the horizontal direction of the role hierarchy structure, there is a measurable range according to access control permissions. This can resolve the problem where the measurable **Rank_Diff** is negligible, but the delegatee is not a member of the active role. For example, in Figure 1, a user (*Cathy*) of *ORDIR* wants to activate the delegated *CARDDIR* to execute a task. And the **Rank_Diff** between *Cathy* and *CARDDIR* is negligible. However, because the two roles are defined in two different studios, the delegation obviously has risk due to their difference of permissions between *CARDDIR* and *ORDIR*.

The third component of risk is **Workflow_Value**. It is measured by importance degree of an object (an workflow instance in *MRARD*). Generally, as is shown in Definition 1, **Workflow_Value** requires SSOs' assignment before a workflow instance is started.

4.1. Measure Rank_Diff

This component of measurable risk is calculated by the difference between **Role's Rank** of an *active role* and **User's Rank**. The former describes the basic rank of an *active role*,

and the latter describes the user's rank which is derived from **Role's Rank** by the relationship of *UA* (User-Role Assignment).

The function of **Role's Rank** is defined as follows:

Definition 4: **Role's Rank**

$$\Re(r : R) : R \rightarrow N$$

Here, R is a role set; N is a set of natural numbers. $\Re(r : R)$ returns **Role's Rank** ($\Re(r)$) of an input role.

$\Re(r)$ is defined by a SSO after a role hierarchy is set up. And $\Re(r)$ is a number which meets the following condition:

$$\forall r1, r2 \in R : r1 \succeq r2 \Rightarrow \Re(r1) \succeq \Re(r2)$$

Here, $r1 \succeq r2$ means $r1$ is a directly super role of $r2$.

Definition 5: **User's Rank**

$$\Re(u : U) = \max\{\Re(r : R) | r \in assigned\_roles(u)\}$$

Here, U refers to a user set; $assigned\_roles(u)$ refers to all roles assigned to the user.

$\Re(u)$ is derived from the maximum **Role's Rank** of all roles of a user.

Based on Definition 4 and Definition 5, we define IDX_{rank_diff} as follows:

Definition 6: IDX_{rank_diff}

$$IDX_{rank\_diff}(u : U, ar : R) = \begin{cases} \frac{a^{(\Re(ar)-\Re(u))}-1}{upper-\Re(ar)}, \text{if } \Re(ar) \geq \Re(u) \\ 0, \text{if } \Re(ar) < \Re(u) \end{cases}$$

Here, a is a real number that is greater than 1, and $upper$ is a real number that is greater than the maximum allowed value of $\Re(ar)$.

IDX_{rank_diff} refers to the direct difference between the **User's Rank** and the activated **Role's Rank**. The formula satisfies the following properties:

- $IDX_{rank\_diff} \geq 0$; and IDX_{rank_diff}=0, when the **User's Rank** is larger than the activated **Role's Rank**;
- IDX_{rank_diff} increases as the activated delegated **Role's Rank** increases, and is biased toward the higher **Role's Rank**.

Definition 7: **Rank_Diff**

$$Rank\_Diff(u : U, ar : R) =$$

$$\begin{cases} \frac{1}{1+e^{(-k)\times(IDX_{rank\_diff}-\frac{m}{IDX_{rank\_diff}})}}, \text{if } IDX_{rank\_diff} > 0 \\ 0, \text{if } IDX_{rank\_diff} = 0 \end{cases}$$

We choose *sigmoid* function[1] to calculate **Rank_Diff**. In Definition 7, ar refers to an *active role*; k determines the slope of the **Rank_Diff** curve with regard to IDX_{rank_diff}; m is equal to $IDX^2_{rank\_diff}$ when *Rank_Diff(u, ar)*=0.5.

rank_diff is a probability that User's Rank reaches the activated Role's Rank. **rank_diff** monotonically increase with IDX_{rank_diff}. And if the User's Rank is equal to, or larger than the Role's Rank, **rank_diff** should be negligible and represented by zero.

4.2. Measure Range_Diff

Range_Diff measures the risk component when an *active role* is delegated to a user whose assigned roles exclude the *active role*, but the User's Rank is the same as or similar to the Role's Rank.

Definition of **Range_Diff** is based on the following assumption: *If a user has fewer permissions an active role is assigned, risk is higher.* The assumption comes from the following fact: *When the delegator and the delegatee are at the same division or involved in similar jobs, the risk is low because the delegatee is familiar with the delegated role's job function. Therefore, the intersection of delegatee's and delegated role's permissions is big. However, when the delegator and the delegatee are at different divisions, even different companies, the risk is high because the delegatee is not familiar with the delegated role's job function. Therefore, the intersection of delegatee's and the active role's permissions is small.* We formally define the **Rank_Diff** as follows:

Rank_Diff:

Definition 8: IDX_{rank_diff}

$$IDX_{range\_diff}(u : U, ar : R) =$$

$$\frac{a'^{(|assigned\_permissions(ar) \cap assigned\_permissions(u)|)} - 1}{upper' - |assigned\_permissions(ar)|}$$

Here, a' is a real number that is greater than 1, and $upper'$ is a real number that is greater than the maximum allowed value of $|assigned\_permissions(ar)|$; In Definition 8, $authorized\_permissions(ar : R) =$

$$\{p \in PRMS | ar \succeq^+ r' \wedge (p, r') \in PA\}$$

and $assigned\_permissions(u : U) =$

$$\{p : PRMS | \forall r \in assigned\_roles(u) : (r, p) \in PA\}$$

Here, PA is Permission Role Assignment; and |set| refers to the number of elements in the set. $ar \succeq^+ r'$ refers that ar is a direct or indirect super role of r'.

Definition 9: **Range_Diff**

Range_Diff(u: U, ar: R) =

$$\begin{cases} \frac{1}{1+e^{(-k')\times(IDX_{range\_diff}-\frac{m'}{IDX_{range\_diff}})}}, \text{if } IDX_{range\_diff} > 0 \\ 0, \text{if } IDX_{range\_diff} = 0 \end{cases}$$

We also choose *sigmoid* function to calculate **Range_Diff**. In Definition 9, ar refers to an *active role*; k' determines the slope of the **Range_Diff** curve with regard to IDX_{range_diff}; m' is equal to $IDX^2_{range\_diff}$ when *Range_Diff(u, ar)*=0.5.

4.3. Measure Workflow_Value

WorkflowImport has been defined in Definition 1. When a user activates his delegated role or one of its sub-roles to

execute a workflow instance, there is more risk when the workflow instance has more Workflow_Value.

5. Risk_Level Evaluation

5.1. Fuzzy Set and Fuzzy Logic

A fuzzy subset FS of a set S can be defined as a set of ordered pairs, each with the first element from S (name of subset), and the second element from the interval [0,1] (membership), with exactly one ordered pair present for each element of S. A membership function of the fuzzy set describes a mapping between FS and membership[12]. Fuzzy logic was introduced by Dr. Lotfi Zadeh[13]. It is a superset of Boolean logic. Fuzzy logic is extended to handle the concept of partial truth.

The theories of fuzzy set and fuzzy logic can deal with very large scale and complex scenarios in the real world. Therefore, they are applied in many decision-based control applications. A fuzzy expert system is one of the typical applications. And its general inference process includes four steps: FUZZIFICATION; INFERENCE; COMPOSITION; DEFUZZIFICATION.

5.2. Fuzzy Risk Evaluation Policy

The process of **Fuzzy Risk Evaluation** evaluates risk_level according to input measurable risk which is defined in Definition 3.

Definition 10: **Fuzzy Risk Evaluation Policy**

$$FREPolicy = \langle FS_{Rank\_Diff}, FS_{Range\_Diff}, FS_{Workflow\_Value}, FS_{Risk\_Level} \rangle$$

Here, FS is a prefix notation for fuzzy subsets, which include, for example, *high, middle* and *low*. Each fuzzy subset is generally defined by a membership function.

The membership functions of fuzzy subsets are valuable to further study, but they are not the main work of this paper. We simply define a fuzzy subset as a range attribute, such as (lowbound, highbound). The range refers to the fuzzy areas of *low* and *high*, and non-zero area such as *middle*. We also define the membership functions as:
low(x)=

$$\begin{cases} 100\%, \text{if } x \leq lowbound \\ \frac{highbound - x}{highbound - lowbound}, \text{if } lowbound \leq x \leq highbound \\ 0\%, \text{if } x \geq highbound \end{cases}$$

middle(x)=

$$\begin{cases} 0, \text{if } x \leq lowbound \vee x \geq highbound \\ \frac{(x - lowbound) \times 5}{highbound - lowbound}, \\ \text{if } lowbound \leq x \leq \frac{highbound - lowbound}{5} + lowbound \\ \frac{(highbound - x) \times 5}{highbound - lowbound}, \\ \text{if } highbound - \frac{highbound - lowbound}{5} \leq x \leq highbound \\ 100\%, \text{else} \end{cases}$$

high(x)=

$$\begin{cases} 0, \text{if } x \leq lowbound \\ \frac{x - lowbound}{highbound - lowbound}, \text{if } lowbound \leq x \leq highbound \\ 100\%, \text{if } x \geq highbound \end{cases}$$

For example, we define the membership functions of fuzzy subsets for Rank_Diff: *high, middle, low*. And we define the fuzzy area of *high* is (40%, 50%); the range of *middle* is (0%, 50%); and the fuzzy area of *low* is (0%, 10%).

Here is an example policy of Definition 10: <high, high, high, high>. It means when a user executes a task, if the Rank_Diff is high, the Range_Diff is high and the Workflow_Value is high , the risk_level is high.

It is not difficult to design **Fuzzy Risk Evaluation Policies**, because the number of **Fuzzy Risk Evaluation Policies** is usually limited, the policy is intuitionistic and there is a case-based method to assist the policy checking. First, the number of the policies is determined by the number of the risk's components and the number of their individual fuzzy subsets in each **Fuzzy Risk Evaluation Policy**. Because every scenario with measurable risk should belong to one fuzzy subset of risk_level, there is at most $3^3 = 27$ policies that SSOs should make. In addition, a **Fuzzy Risk Evaluation Policy** has an important feature: *Monotone Non-decreasing* (which can help SSOs when they are making policies). That is,

$$\frac{\partial FS_{Risk\_Level}}{\partial component} \geq 0$$

Here, the *component* refers to each component except for FS_{Risk_Level} in the **Fuzzy Risk Evaluation Policy** definition. Because of *Monotone Non-decreasing*, it is impossible for some potential policies to co-exist based on the feature. For example, it is impossible for the policy of <high, *middle*, high, high> to coexist with the policy of <high, *high*, high, middle>.

Next, the **Fuzzy Risk Evaluation Policy** is intuitionistic for SSOs, because the **Fuzzy Risk Evaluation Policy** uses intuitionistic words like *high*, *low*, etc. to express the degree of the components.

Last but not least, we can design a cased-based method to check inconsistency of **Fuzzy Risk Evaluation Policies**. During the case-based checking, the SSOs use a typical scenario which should belong to a certain risk_level. For

Table 1. Parameters in membership functions

	Rank_Diff	Range_Diff	Workflow_Value	Risk_Level
high	(40%, 50%)	(40%, 50%)	(60%, 80%)	(6, 9)
middle	(0%, 50%)	(0%, 50%)	(0%, 80%)	(2, 7)
low	(0%, 10%)	(0%, 10%)	(0%, 20%)	(0, 3)

example, there is a case in which *Mike* delegates his role to *Tom*, and the risk is high when *Tom* uses *Mike's* role to execute a certain task (as is shown in Figure 2). The system can calculate the *Rank_Diff, Range_Diff, and Workflow_Value* in the above case, then calculate the risk_level according to the measurable risk that consists of the three components; finally, the system checks whether the predefined policy is correct according to the feature of *Monotone Non-decreasing*. If the evaluated *Risk_Level* is very low rather than high, then the SSOs should justify the **Fuzzy Risk Evaluation Policies**.

5.3. Evaluation Process based on Fuzzy Logic

Based on the policy definition in the above section, we use defined policies to determine risk_level for a measurable risk vector.

As is shown in Figure 3, the process of evaluation for a measurable risk vector includes four standard steps: Fuzzification; Inference; Composition; and Defuzzification.

1) Fuzzification will use the membership functions of fuzzy subsets for each component of measurable risk to assign a degree to every fuzzy subset;
2) Inference will evaluate a measurable risk vector by each **Fuzzy Risk Evaluation Policy**. Because the policy uses the AND operator in the **Fuzzy Risk Evaluation Policy** definition, we use MIN inferencing to get the evaluation result of each policy;
3) Composition will compose all the results from the Inference phase. Due to MIN inferencing, we use MAX composing to compose the results;
4) Defuzzification will use the membership function to defuzzificate the result from the Composition phase. We use CENTROID rather than MAXIMUM to get the final risk_level.

5.4. An Simulated Case Study

As is shown in Table 1, we describe the parameters of membership functions of fuzzy subsets for the four components. And as is shown in Figure 3, we describe three **Fuzzy Risk Evaluation Policies**. Then we will demonstrate the calculation process when we assume we get a vector of measure_risk as: $\langle 45\%, 45\%, 75\% \rangle$.

We use MIN-MAX to do inference. According to the first policy in Figure 3, the result of inference is 19% of the high fuzzy subset of *Risk_Level*. The inference result

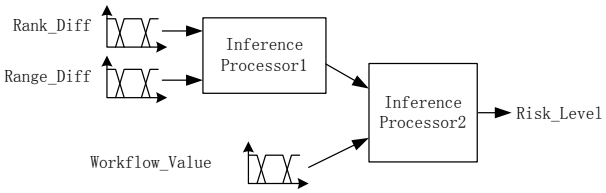

Figure 4. Multi-level structure of fuzzy inference

of the second policy is 8% of the middle fuzzy subset of *Risk_Level*. And the result of the third policy is 0% of the low fuzzy subset of *Risk_Level*.

Through composition and defuzzification, we calculate that the *Risk_Level* is 6.

But, if we are given the risk of $\langle 5\%, 45\%, 85\% \rangle$, the risk_level is zero after inference, composition and defuzzification according to the membership functions in Table 1 and the policies in Figure 3. That is unreasonable. The reason is the policy set is incomplete or not well designed. Thus, we should improve the policy set by designing new **Fuzzy Risk Evaluation Policies**.

5.5. Problems in Fuzzy Risk Evaluation

5.5.1. Other Components of Risk and Multi-Level Inference. An information system is so complex that there are a few other components of risk. For example, credit of a user, which is often used in a financial system, also affects the risk in role-based delegation. A user with poor credit could maliciously operate his/her delegated role. The potential operation could lead to more risk. Thus, further research will explore more risk components in the measurable risk model.

A multi-level structure for fuzzy inference is an alternative way in dealing with more components in measurable risk. For example, Figure 4 shows a multi-level structure for fuzzy inference for measurable risk evaluation. In the multi-level structure, the risk inference is divided into two levels: the processor in the first level infers delegation risk, and the processor in the second level infers final risk_level. This design can limit every inference processor to no more than $3^2 = 9$ policies.

5.5.2. Define Membership functions of fuzzy subsets and Fuzzy Risk Evaluation Policy Set. To design well-defined membership functions of fuzzy sub sets and a **Fuzzy Risk Evaluation Policy** set is an important issue in *MRARD*, although the number of the membership functions and the policies is limited. The analysis in section 5.2 preliminarily gives us a potential method which uses real typical cases to check **Fuzzy Risk Evaluation Policies**. The future work will research how to simplify to the design of these membership functions and policies.

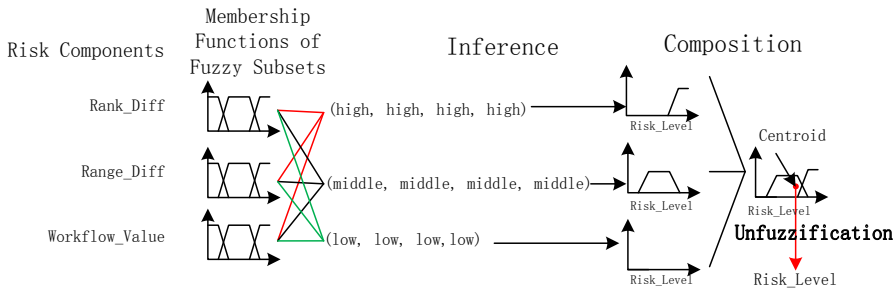

Figure 3. Evaluation process based on fuzzy logic

6. Determine an Access Request

6.1. Risk Mitigation

The decision in *MRARD* depends on both risk_level and a risk mitigation action. we define **Risk Mitigation** action as follows:

Definition 11: **Risk Mitigation**
$Risk\_Mitigation =$

$$Actions \times Risk\_Level \times MitigationEffect$$

Definition 11 means on which risk_level a risk mitigation action can be effective and how much effect the mitigation has. In Definition 11, *Actions* refers to all real risk mitigation actions, for example, "send an email to the SSOs board" or "cooperate by another user with a delegatee's highest-rank role which is assigned by SSOs"; *Risk_Level* refers to risk_level at which the risk mitigation action can work; and *MitigationEffect* refers to how much effect the risk mitigation action can produce, for example, "reduce 50%".

Cheng et al.[14] bound risk mitigation with a certain risk_level. But in our framework, some risk mitigation actions can independently work on a certain risk_level, *MRARD* will select the most efficient and valid risk mitigation action to implement risk_level according to the defined risk mitigation actions.

6.2. Decision Policies and Determine a Request

The decision polices are simple and variable. That is, a SSO can pre-define acceptable risk_level for critical tasks. In addition, a SSO can also define a policy that accepts any one of the three users of the lowest risk_level under a certain risk mitigation action.

MRARD can infer risk_level according to context (Rank_Diff, Range_Diff, Workflow_Value) of a request and a risk mitigation action. And based the above risk adaptive decision policies, *MRARD* can dynamically determine whether the request can be allowed under the risk mitigation action.

7. Efficiency of MRARD

MRARD can strengthen security in a workflow system which supports role based delegation. *MRARD* is a complementary mechanism to protect sensitive data in a role based delegation supporting workflow system. In front of *MRARD*, the delegation policies designed by professional SSOs will provide the workflow system with the hard boundary[1] protection. Furthermore *MRARD* explicitly measures the potential risk which the traditional mechanism can not provide. *MRARD* can efficiently identify risk according to task execution context (Rank_Diff, Range_Diff and Workflow_Value). And the system can make a dynamical decision due to different situations in delegation. Thus, *MRARD* provides a more flexible security complementary mechanism for a role-based delegation supporting workflow system.

8. Related Works

Researchers have developed some languages and methods to specify who can delegate which role to a certain user[5][4] and how to restrict the delegatee. But they could be too rigid in dealing with the role-based delegation. In these researches all users have the same priority to accept delegation if the policies and rules allow, and the delegatee can apply his/her delegated role to any workflow instance only if the delegated role is qualified for workflow designs. As a result, the SSOs can not deal with the potential risk when the workflow system and the users are drastically changing. This paper explicitly applies the measurable risk to deal with the risk that comes from delegation.

Risk is argued as a more flexible way to deal with a dynamic information system[2][1][15]. However, currently, many researches focus on risk in a traditional financial system[8], an information system in an enterprise[9][10]. Even though researchers have finished some newest researches on access control systems[16][1][7][15], there are some limitations in these researches as are analyzed in section 3.1, when these researches are applied to a role-based enterprise-oriented application.

Dimmock et al.[16] used risk thresholding predicates to deal with the multi-value problem of risk and cost in a distributed file storage and publication service. However, the predicates are not good at resolving the multi-value problem. Zhang[3] discussed a risk adaptive method which considered risk and benefit. Different from the papers of[16][3], our paper proposes a measurable risk as a vector which consists of measurable risk components, applies the theory of fuzzy set to define the **Fuzzy Risk Evaluation Policy**.

9. Conclusions and Future Work

This paper proposes a framework which explicitly evaluates risk_level and decides a request according to the risk_level and a risk mitigation action. Then, the paper proposes a novel framework, *MRARD*, that applies measurable risk to strengthen security in a role-based delegation supporting workflow system. *MRARD* explores a novel way to strengthen the security. When a user wants to execute a task in a workflow instance in *MRARD*, the system will explicitly evaluate the potential risk components. In *MRARD*, we leverage fuzzy set and fuzzy logic to resolve the multi-value problem in evaluating the risk_level. Different from rigid binary-decision of traditional access control models, *MRARD* can evaluate the risk_level of each delegation, and determine dynamically whether a risk-aware action can be allowed under a risk mitigation action.

Further research will design a general framework which trades off risk and benefit that come from a risk-aware action.

Acknowledgement

Thank Prof. Ninghui Li, Prof. Min Li for their contributions to our paper. Thank Anonymous Reviewers for their comments.

References

[1] P. Cheng, P. Rohatgi, C. Keser, P. A. Karger, G. M. Wagner, and A. S. Reninger, "Fuzzy multi.level security: An experiment on quantified risk.adaptive access control," in *Proceedings of 2007 IEEE Symposium on Security and Privacy(SP'07)*. Oakland, California, USA: ACM, May 2007, pp. 222 – 230.

[2] JASON, "Horizontal integration: Broader access models for realizing information dominance," MITRE Corporation, http://www.fas.org/irp/agency/dod/jason/classpol.pdf, Tech. Rep. JSR-04-132, 2004.

[3] L. Zhang, A. Brodsky, and S. Jajodia, "Toward information sharing: Benefit and risk access control (barac)," in *Proceedings of the 7th IEEE International Workshop on Policies for Distributed Systems and Networks (POLICY'06)*, 2006, pp. 45–53.

[4] L. Zhang, G.-J. Ahn, and B.-T. Chu, "A rule-based framework for role-based delegation and revocation," *ACM Transaction on Information and System Security*, vol. 6, no. 3, pp. 404–441, September 2003.

[5] N. Li, B. N. Grosof, and J. Feigenbaum, "Delegation logic: A logic-based approach to distributed authorization," *ACM Transactions on Information and System Security (TISSEC)*, vol. 6, no. 1, pp. 128–171, 2003.

[6] D. F. Ferraiola, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, "Proposed nist standard for role-based access control," *ACM Transactions on Information and System Security (TISSEC)*, vol. 4, no. 3, pp. 224–274, August 2001.

[7] I. Molloy, P.-C. Cheng, and P. Rohatgi, "Trading in risk: Using markets to improve access control," in *Proceedings of New Security Paradigms Workshop (NSPW'08)*. Lake Tahoe, California, USA: ACM, September 2008, pp. 1–19.

[8] P. Jorion, *Value at Risk: The New Benchmark for Managing Financial Risk*, 3rd ed. McGraw-Hill, 2006.

[9] W. Ru and J. Eloff, "Risk analysis modelling with the use of fuzzy logic," *Computers & Security*, vol. 15, no. 3, pp. 239–248, 1996.

[10] M. Neil, N. Fenton, and M. Tailor, "Using bayesian networks to model expected and unexpected operational losses," *Risk Analysis*, vol. 25, no. 4, pp. 963–972, 2005.

[11] D. Georgakopoulos, M. Hornick, and A. Sheth, "An overview of workflow management: From process modeling to workflow automation infrastructure," *Distributed and Parallel Databases*, vol. 3, no. 2, pp. 119–153, 1995.

[12] M. Kantrowitz, "Answers to questions about fuzzy logic and fuzzy expert systems," Carnegie Mellon University, http://www.cs.cmu.edu/Groups/AI/html/faqs/ai/fuzzy/, Tech. Rep., 1997.

[13] L. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, 1965.

[14] P.-C. Cheng, P. Rohatgi, and C. Keser, "Fuzzy multi.level security: An experiment on quantified risk.adaptive access control," IBM Thomas J. Watson Research Center, http://domino.research.ibm.com/comm/research_projects.nsf/pages/system_s_security.index.html/$FILE/ATTH4YZ5.pdf, Tech. Rep., 2007.

[15] L. Teo, G. Ahn, and Y. Zheng, "Dynamic and risk-aware network access management," in *Proceedings of the eighth ACM symposium on Access control models and technologies (SACMAT'03)*, Yorktown Heights, New York, USA, June 2003, pp. 156 – 162.

[16] N. Dimmock, A. Belokosztolszki, and D. Eyers, "Using trust and risk in role-based access control policies," in *Proceedings of the Ninth ACM Symposium on Access Control Models and Technologies , (SACMAT'04)*. Yorktown Heights, New York, USA: ACM, June 2004, pp. 156–162.