IRBAC 2000: Secure Interoperability Using Dynamic Role Translation*

Apu Kapadia, Jalal Al-Muhtadi, Roy H. Campbell, Dennis Mickunas
Department of Computer Science
University of Illinois
Urbana, IL, U.S.A.

Abstract *The secure interaction between two or more administrative domains is a major concern. We examine the issues of secure interoperability between two security domains operating under the Role Based Access Control (RBAC) Model. We propose a model that quickly establishes a flexible policy for dynamic role translation. The role hierarchies of the local and foreign domains can be manipulated through our Role Editor which is used to set up associations between these hierarchies. These associations result in a combined partial ordering of the role hierarchies, which can be used to make meaningful access control decisions for secure interoperability.*

Keywords: RBAC Interoperability Security Policies Access Control

1 Introduction

We use the terms *domain* and *security domain* to refer to an *administrative domain*, which is defined as "A collection of hosts and routers, and the interconnecting network(s), managed by a single administrative authority [8]." This single administrative authority will include a *security officer*. We assume that the domains operate under the Role Based Access Control (RBAC) [11] model. In RBAC, a user is assigned to a *role* that indicates the user's function in his or her organization.

Consider the scenario when two security domains, A and B, desire to interoperate securely. In order to this, A and B need to establish a secure context. We define a *secure context* to be "a secure session between two entities subject to a domain security policy." To establish a secure context, both domains need to agree on a security policy. At the basic level, both domains can revert to a default security policy that provides a basic level of security. However, such naïve approaches are static and do not offer flexibility to security aware applications and domains. For example, if a client object, C, in domain A wants to establish a secure context with a target object, T, in domain B, it must rely on the underlying secure mechanism to establish the secure context. For a higher degree of flexibility, both the client and target objects, C and T, should be aware of each other's identities. This occurs naturally within a single domain. However since C and T are in different domains, their identities are generally unknown to each other. Specifically, in RBAC, if the client object, C, has the foreign role of *Professor*, and if the target object, T, usually interacts with client objects with the local role *Manager*, how would each domain determine the relevance of the foreign roles? This is the problem that we will address.

To solve this problem, we propose a policy framework that facilitates secure interoperability between two or more domains. Each domain is represented by a CORBA [9] *Object Request Broker (ORB)* in our testing framework. The policy framework works with a set of associations between the local and foreign role hierarchies. These associations form a combined

*This work is partially funded by DOD grant MDA-904-98-C-A895 to the University of Illinois.

hierarchy that is partially ordered. This combined partial ordering is used to attain the level of flexibility desired. Foreign roles can now be translated into local roles, which are understandable to local entities. At the very least it provides the default, or minimal, role translation. An example of the minimal translation is if all foreign roles are treated as a single Guest role in the local domain. At the other extreme, it allows the security officer to specify a highly explicit role mapping (or association). An example of an explicit mapping is if the security officer specifies that a Professor from domain A is equivalent to a Manager in domain B.

Once these associations are set up, all foreign roles are dynamically translated into local roles. This being done, applications can make meaningful access control decisions. These associations are managed through the *Role Editor*, which is at the heart of our policy framework, and is the security officer's tool for secure interoperability. Our model is named the Interoperable Role Based Access Control Model 2000 (IRBAC 2000).

2 Policy framework

Consider the simple role hierarchies H_0 and H_1, for domains D_0 and D_1 respectively, described in Figure 1. An arrow directed from a role x to a role y means that x is the parent of y and is hence higher than y in the hierarchy. Although the structures of the role hierarchies are similar, they differ in their semantics. If a client object from a foreign domain, with the role Manager, wants to interoperate with an application in the local domain, that usually allows only local Professor roles, the application must be able to translate the foreign Manager role into something meaningful. We also observe that both domains have the Guest role. If foreign roles are not understood, one can define a simple policy framework to treat all foreign roles to be equivalent to the local Guest role. However, this kind of an approach is not very flexible, since all foreign roles are considered to be of only one type, i.e., the local Guest role.

Our policy framework creates a combined partial ordering by adding a set of associations between the two role hierarchies. By doing so, one can easily manipulate the levels of access for specific foreign roles.

To formalize this, let R_0 denote the set of roles in the local domain D_0. Let R_1 denote the set of roles in some foreign domain D_1. Then the role hierarchies H_0 and H_1 are partial orderings as shown in Figure 1 (solid arrows only). We define $x > y$ to mean that x is higher than y in the hierarchy, or, x is the *ancestor* of y. Role-names with subscripts, for example $Manager_{R_0}$, will read as $Manager$ from R_0. Let R_1R_0 denote the set of associations from R_1 to R_0. We note that $R_1R_0 \subseteq R_1 \times R_0$. Once these associations are defined, we obtain a dynamic role translation model for secure interoperability.

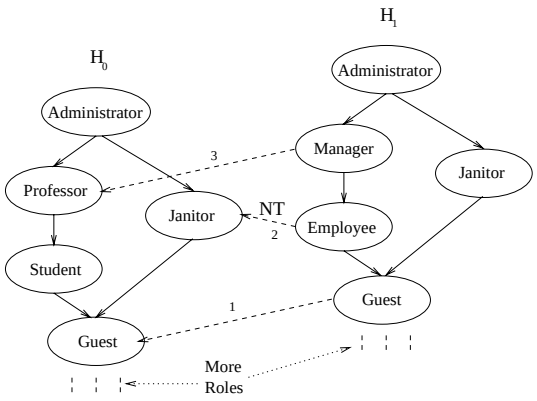

Figure 1: Associations between hierarchies - dotted lines

For example, in Figure 1 we have the association from $Manager_{R_1}$ to $Professor_{R_0}$ (labeled as 3). This implies that $Manager_{R_1}$ from the foreign domain D_1 will be translated to $Professor_{R_0}$ in the local domain D_0. We will use the following notation to symbolize an association: $Manager_{R_1} \mapsto Professor_{R_0}$. We can also write this as $(Manager, Professor) \in R_1R_0$.

We define two kinds of associations: *transitive* and *non-transitive* associations.

Transitive Associations: In Figure 1 we can see the association $Guest_{R_1} \mapsto Guest_{R_0}$ (la-

beled as 1). Hence $Guest_{R_1}$ will be translated to $Guest_{R_0}$. This also implies that all the ancestors of $Guest_{R_1}$ will map to the $Guest_{R_0}$. Consider the association $x_{R_1} \mapsto a_{R_0}$. Then, $\forall y \in R_1$, if $y_{R_1} > x_{R_1}$, then $y_{R_1} > a_{R_0}$. That is, we can say that y_{R_1} is the *ancestor* of a_{R_0} since, y_{R_1} implicitly maps to a_{R_0}. This is a transitive property in which roles inherit the associations of other roles that are lower in the hierarchy. Transitivity is also followed in the local hierarchy. Therefore, $\forall b \in R_0$, such that $a_{R_0} > b_{R_0}$, if $x_{R_1} \mapsto a_{R_0}$ then $x_{R_1} > b_{R_0}$. Hence these associations are called transitive associations.

Non-transitive Associations: The security officer may specifically want to grant access to a particular foreign role. At the same time, the security officer does not have the power to alter the foreign hierarchy. For example, in Figure 1, the security officer may want to specifically translate $Employee_{R_1}$ to the $Janitor_{R_0}$, and deny $Administrator_{R_1}$ and $Manager_{R_1}$ from inheriting this association. If the security officer had the power to alter the foreign hierarchy, the officer could semantically achieve this by altering the foreign role hierarchy. But the security officer of the local domain does *not* have the ability or power to do so. Hence, we introduce the concept of a non-transitive association. In Figure 1, we can see a such an association between $Employee_{R_1}$ and $Janitor_{R_0}$. We will use the following notation to denote such a mapping: $Employee_{R_1} \mapsto_{NT} Janitor_{R_0}$ or $(Employee, Janitor)_{NT} \in R_1 R_0$.

2.1 Role translation policies

A set of transitive and non-transitive associations, between the foreign and local hierarchies, can be used to create a combined partial ordering and define a security policy. Such policies can be put into three categories:

Default Policy: This policy involves setting up a minimal number of associations between a set of roles in the foreign hierarchy and $Guest_{R_0}$. These associations will allow a particular set of foreign roles to interoperate at the *default* level of security. In Figure 1 we can see the association $Guest_{R_1} \mapsto Guest_{R_0}$. This corresponds to a default policy. $\forall x \in R_1$, if $x_{R_1} > Guest_{R_1}$, then $x_{R_1} > Guest_{R_0}$

This scheme is the easiest to set up, but is also the least flexible. This is because all foreign principals are treated as the same role, i.e., $Guest_{R_0}$. However, this scheme is important since it allows the basic level of interoperability, and can be used in conjunction with a partially explicit policy, which is described next.

Explicit Policy: In this policy, the security officer can specifically map each foreign role to a local role. This policy is an extreme case that illustrates the maximum amount of flexibility, in which the security officer can make the mappings explicit for *each* foreign role. An effective way to do this would be to make non-transitive associations from every role in R_1 to a subset of roles in R_0.

Partially Explicit Policy: This policy illustrates the true flexibility of our dynamic role translation model. A mapping is partially explicit if it is not an explicit policy, and when there are one or more associations, usually in addition to the default policy. In such partial hierarchies, foreign roles without explicit associations still have logical associations by means of the partial hierarchy.

The associations labeled 1, 2 and 3 in Figure 1 illustrate this policy. We can see that $Manager_{R_1}$ is higher than $Professor_{R_0}$ ($Manager_{R_1} > Professor_{R_0}$) since we have the association $Manager_{R_1} \mapsto Professor_{R_0}$. Hence, $Manager_{R_1}$ will be translated into $Professor_{R_0}$. Similarly, $Administrator_{R_1} > Professor_{R_0}$ since $Administrator_{R_1} > Manager_{R_1}$. Hence, $Administrator_{R_1}$ will be translated into $Professor_{R_0}$.

For example, applications in D_0 that usually grant access to $Student_{R_0}$ (Figure 1) also grant access to the foreign $Manager_{R_1}$ role since $Manager_{R_1} > Student_{R_0}$. It is important to note that $Employee_{R_1}$ will be treated as $\{Guest_{R_0}, Janitor_{R_0}\}$. Hence, applications that allow access to $Student_{R_0}$ will not grant $Employee_{R_1}$ access unless an association $Employee_{R_1} \mapsto Student_{R_0}$ is provided, for example. This is a demonstration of a partially

explicit mapping.

This model is highly flexible because it allows the security officer to specify the placement of specific foreign roles in the hierarchy, without enforcing a mapping for each and every role in the foreign hierarchy.

2.2 Dynamic translation

If new roles are added to the local or foreign hierarchies, these roles automatically fit into the translation model, and the security officer does not need to make any immediate changes. Hence the policy framework is dynamic in nature. By following an appropriate notification protocol for the deletion and addition of roles (see Section 3.2), the role translation model will dynamically respond to such changes.

3 Security issues

Once we have a role translation policy defined, one can ask the question, "How secure is this system?" We must first spell out the underlying assumptions for such a policy. This policy framework does not specify how a secure context is set up. Our role translation model provides a meaningful translation of foreign roles into local roles. This is a key step that must be followed before establishing a secure context across domains. Hence, our model is useful in establishing *meaningful* secure contexts. Our testbed uses the Secure Inter-ORB Protocol (SECIOP) [3] to establish a secure context between a client and target object operating within separate ORBs. This is explained in more detail in Section 5. In this section we will talk about the relative merits and shortcomings of our dynamic role translation model.

3.1 Conflict resolution

It is easy to imagine a situation in which a particular association conflicts with another association. For example, consider Figure 2. If we had a partially explicit policy with the association $Manager_{R_1} \mapsto Student_{R_0}$, it is clear that $Employee_{R_1}$ will be translated to $Guest_{R_0}$

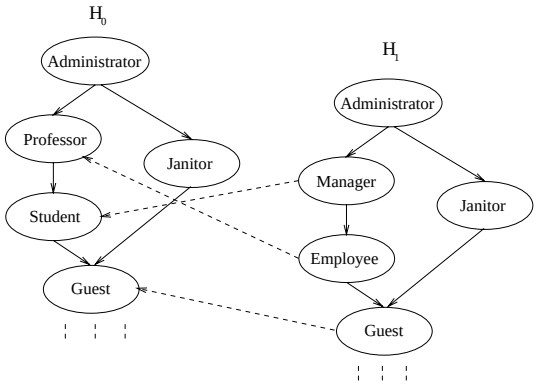

Figure 2: Conflicting associations

and not $Student_{R_0}$. Now consider a second association $Employee_{R_1} \mapsto Professor_{R_0}$. Now, $Employee_{R_1}$ will be translated to $Professor_{R_0}$. We can see that these two associations conflict since now $Manager_{R_1} > Professor_{R_0}$, even though the association $Manager_{R_1} \mapsto Student_{R_0}$ does not allow this. In such situations, conflicts are resolved by giving the foreign role the highest possible translation in the local hierarchy that the associations can allow. Hence $Manager_{R_1}$ will be translated into $Professor_{R_0}$ since $Professor_{R_0} > Student_{R_0}$.

This may result in a security hazard in which the security officer may grant a foreign role higher access without meaning to do so. To prevent such a situation, our Role Editor can be put into the *reachability mode*. In this mode, the security officer can select foreign roles, and a reachability graph will be drawn for that selection. More specifically, the Role Editor color codes all the local roles that are *reachable* from the selected foreign role.

3.2 Deletion of roles

When a role is removed from the local hierarchy, for every association that connects to the removed role, a set of new associations are added to all the children of the removed role. Transitive associations are replaced by a set of transitive associations, and non-transitive associations are replaced by a set of non-transitive associations. For example, in Figure 1, if

$Professor_{R_0}$ is removed from H_0, then the association $Manager_{R_1} \mapsto Professor_{R_0}$ is replaced by $\{Manager_{R_1} \mapsto Student_{R_0}\}$.

When a role is removed from the foreign hierarchy, for every association that begins at the removed role, a set of new associations are added from all its parents to the role that it was connected to. For example, in Figure 1, if $Manager_{R_1}$ is removed from H_1, the association $Manager_{R_1} \mapsto Professor_{R_0}$ is replaced by $\{Administrator_{R_1} \mapsto Professor_{R_0}\}$.

For this to be feasible, if there is a change in the role hierarchy of a particular domain, all the domains that it interoperates with must be notified through some protocol.

3.3 Domain crossing

When a principal attempts to interoperate with a target in another domain, it must *cross* one domain boundary. We call this a *domain crossing*. As outlined below, multiple domain crossings can be a security hazard because it may allow *infiltration* and *covert promotion*.

Infiltration: Consider the case when domains D_2 and D_1 decide to interoperate, and domains D_1 and D_0 decide to interoperate. This does not imply that D_2 and D_0 want to interoperate. Even though D_0 may not want to interoperate with D_2, principals from D_2 could first enter D_1 and consequently infiltrate into D_0. Hence, our role translation model cannot be used for translating roles for multiple domain crossings.

Covert promotion: Another problem with such domain crossings is that principals can cross domain boundaries and possibly return to the original domain with a role higher than their original role. Effectively, a principal can *covertly promote* itself in the role hierarchy by crossing multiple domains.

To prevent infiltration and covert promotion, the translation should be made valid for only one domain crossing. Each domain should translate the principal's *original* role, and not simply the role from the previous domain. This will ensure that irrespective of domain crossings, the role translation model will be valid. This can be done by including the original domain and role names within the principal's certificate. However, this requires the cooperation of all the domains because rogue domains can refabricate a principal's certificate. Hence, without such cooperation, it is possible to have covert promotion, and infiltration.

4 Example applications

Mobile Networking: Mobile networking involves the cooperation of various domains. Principals are continuously entering and leaving various domains. In such a situation, our role translation model will provide efficient role translation for the principals. Since the different domains need to interoperate securely, our role translation model will be beneficial in providing better security services.

Mobile Agents: When mobile agents traverse the network and cross various domains, it will be extremely helpful to provide these agents with role translation. Similar to mobile networking, our translation model provides mobile agents with access to better security services.

5 Architecture

We have incorporated our policy framework into an ORB called JacORB [5] as described in Section 5.1. In our architecture, each ORB corresponds to a domain. These ORBs interoperate using our policy framework. Client and target objects have certificates that contain their role names. The secure context establishment takes place through SECIOP [3], and hence the two ORBs can interoperate securely.

5.1 System components

JacORB [5] is a publicly available Java ORB. We are using JacORB as our test-bed for various interoperability schemes, namely the CSI Levels 1 and 2 [10]. JacORB, by itself, does not have any security built into it. As a part of our project we are developing a SECIOP compliant ORB by using UIUC SESAME [7]. UIUC SESAME is a Java implementation of SESAME

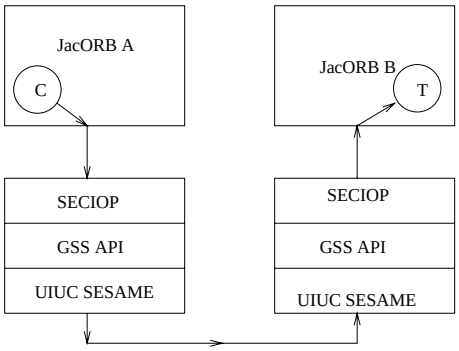

Figure 3: Architecture

[12], an extension to Kerberos that supports roles and delegation. UIUC SESAME was developed in an earlier project, and we are extending it to include our policy framework, and support delegation. GSS API [4] provides a generic interface to UIUC SESAME from JacORB.

Our policy framework is integrated into UIUC SESAME, and our Role Editor (Figure 4) interfaces with the policy server in real time. The Role Editor is used graphically to specify the role hierarchies, and to examine foreign role hierarchies. However, its most important use is to set up associations between the local and foreign role hierarchies.

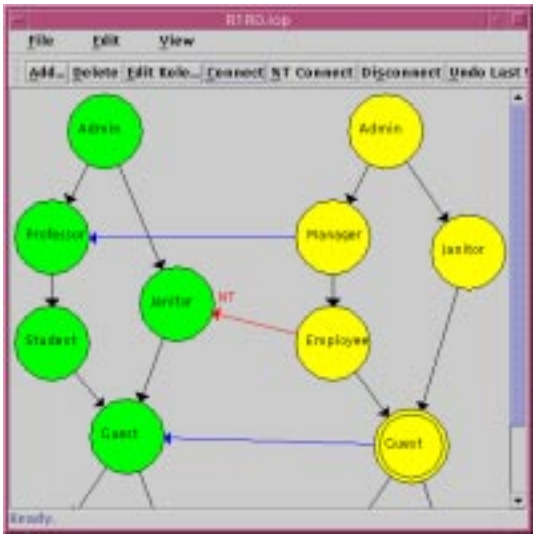

Figure 4: Role Editor

5.2 Implementation details and issues

Once the security officer has established the associations, a foreign principal obtains access to the local domain as follows.

Entry Points: The foreign principal presents its certificate to the policy server, through the GSS API. The policy server then follows all the possible associations and makes a list of all the *entry points* into the local hierarchy. For example, the entry points for $Manager_{R_1}$ in Figure 1 are $Professor_{R_0}$ and $Guest_{R_0}$. This list of roles is added to the certificate of the foreign role. Hence the translation takes place only once, after which the entry points are presented as local roles.

System Concurrency: It is important to make sure that changes in the local hierarchy are reflected properly in the policy framework. As discussed in 3.2, changes in the local hierarchy must be reflected correctly in other interoperating domains. A special protocol is be needed to do this, and is suggested as future work in Section 7.

5.3 Performance

If n is the number of roles in the foreign hierarchy, and if p is the number of associations, then our algorithm for finding the list of entry points has a time complexity of $O(np)$. This translation is only done once, when a foreign principal enters the local domain. Hence, our model imposes negligible overhead on the overall performance of the system. To assess the time taken for a single role translation we created a local hierarchy of 15 roles, and a foreign hierarchy of 15 roles. We added a set of 5, 10 and 15 associations and made 10^5 role translation requests for a random set of foreign roles. The average role translation time for each case is shown in Table 1.

This performance can be further enhanced by caching the role translations. This is because the translations depend only the roles of the principals, and not on the principals themselves. Hence translations can be reused.

Table 1: Role translation performance

Associations	5	10	15
Avg. time	0.08ms	0.14ms	0.18ms

6 Related work

Very little research has been done with respect to secure interoperability between different domains. Campbell, et al. [6] discuss a security architecture for dynamic interoperability in active networks. They propose an architecture that can be used to "recursively install and support the secure deployment of new security mechanisms." In effect, they are dynamically able to install security policies on routers that may belong to different domains. This is an interesting approach where they discuss dynamic firewalls that can be used to combat denial of service attacks. Their system also operates under the RBAC model, among others, and hence our policy framework could be used to enhance their security services.

ORBAsec SL2 is a standard CORBA Security Level 2 [10] Java ORB developed by Adiron [1]. This is similar to our effort of adding interoperability services to a Java ORB. However, they lack a comprehensive policy for establishing secure contexts. Our policy framework could be used to enhance their system.

7 Future work

The policy server interacts with the *Role Editor* and responds to role translation requests. In the next phase of our project, we will be working on a protocol to ensure inter-domain concurrency, as explained in Section 5.2. We will have a SECIOP compliant JacORB that uses our policy server publicly available by September 2000 [2].

Our model can be extended to use a *risk model*. We have provided a framework for specifying risk values for each role. In the future, these risk values can be used in conjunction with dynamic role translation to make better role translation decisions.

8 Conclusion

We have provided an efficient and dynamic method for role translation. This makes secure interoperability more flexible than conventional models. We believe that this model will be extremely useful in mobile networking systems where the secure interoperability between different domains is essential.

References

[1] Adiron ORBAsec SL2. URL: http://www.adiron.com/downloads.html.

[2] Malakim Development Pages. URL: http://choices.cs.uiuc.edu/Security/malakim/.

[3] SECIOP - Security Service Specification, December 1998. URL: http://www.omg.org/docs/ptc/98-12-03.pdf.

[4] R. Braden, L. Zhang, S. Berson, S. Herzog, and S. Jamin. IETF RFC 1508: The Generic Security Services API, September 1993.

[5] Gerald Brose. JacORB - a Java Object Request Broker. Technical Report B-97-02, Freie Universität Berlin, April 1997.

[6] R. Campbell, Z. Liu, D. Mickunas, P. Naldurg, and S. Yi. Seraphim: Dynamic interoperable security architecture for active networks. *IEEE OPENARCH 2000, Tel-Aviv*, March 2000.

[7] M. Chandak. UIUC SESAME: Achieving a Portable Authentication, Access Control, and Delegation Protocol, 1999.

[8] G. Malkin. Internet Users' Glossary - RFC 1983, network working group, August 1996.

[9] Object Management Group. CORBA 2.3.1 /IIOP Specification. Technical report, 1999.

[10] Object Management Group. CSI: Common Secure Interoperability. Technical report, 1996.

[11] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role Based Access Control Models. *IEEE Computer*, 29(2), February 1996.

[12] M. Vandenwauver, R. Govaerts, and J. Vandewalle. Overview of Authentication Protocols: Kerberos and SESAME. In *Proc. 31st Annual IEEE Carnahan Conference on Security Technology*, pages 108–113, 1997.