On The Modeling of Bell-LaPadula Security Policies Using RBAC

Gansen Zhao*
University of Kent, UK
gz7@kent.ac.uk

David W Chadwick
University of Kent, UK
d.w.chadwick@kent.ac.uk

Abstract

The Bell-LaPadula security model is a hybrid model that combines mandatory access controls and discretionary access controls. The Bell-LaPadula security model has been widely accepted in military environments for its capability to specify military style confidentiality policies. The role based access control (RBAC) model has attracted extensive research effort and has been acknowledged as a flexible and policy natural model.

This paper investigates a way of modeling Bell-LaPadula security policies using the RBAC model. The capability of modeling Bell-LaPadula security policies using RBAC model means that applications that are implemented using the RBAC model can then be deployed in military environments and will meet their requirements for information confidentiality.

1. Introduction

The Bell-LaPadula model is one of the most influential security models in military environments. The Bell-LaPadula model was designed to impose strict confidentiality protection on critical information. Though it provides excellent protections over information confidentiality for military applications, it is, to some extent, too strict for use in commercial scenarios where information integrity is of greater importance. Thus applications built on the Bell-LaPadula model are mostly used in military environments or similar.

The RBAC model has been widely accepted as a policy natural access control model and it is suitable for most commercial scenarios. Applications built on the RBAC model have been implemented and widely deployed by commercial companies and educational institutes.

This paper investigates a way of modeling Bell-LaPadula security policies using the RBAC model. The ability to model Bell-LaPadula security policies using the RBAC model means that applications that are implemented based on RBAC model can be deployed in military environments to meet their requirements for information confidentiality. Our investigation leads to the development of a mapping algorithm that can map any given Bell-LaPadula security policy to an equivalent RBAC policy.

The rest of this paper is organized as follows. Section 2 provides an introduction to the RBAC model, the Mandatory Access Control Model and the Bell-LaPadula Model. Section 3 presents the mapping algorithm that can construct a RBAC security policy based on a given Bell-LaPadula security policy. Section 4 reviews previous related research on this topic. Section 5 concludes the paper.

*Gansen Zhao's PhD research is sponsored by Nexor.

2. Access Control Models

2.1. Mandatory Access Control

Mandatory access control imposes security control over subjects and resources based on the predefined attributes of the subjects and the resources. Permissions are not transferable from the owner to another subject.

The Lattice Based Access Control (LBAC) model [9], one of the most common used MAC model, uses lattices to describe multi-level security policies. Sensitivity levels and categories are combined together as security levels. Security levels are then organized as lattices that specify the orders between the security levels, which defines the dominate relation between the security levels. Subjects and objects are all associated with security levels. Each object is associated with a security level to indicate the security classification it is in. An object's security level should accurately describe the security classification according to the information it contains and the system's policy of classifying information. A subject's association with a security level follows the rules that the least possible sensitivity level is applied to the subject, which is no more than necessary to do his job, and the categories applied to the subject are only those categories needed to be known by the subject.

A multi-level security policy requires that subjects can only read objects of dominated security levels and can only write to objects of dominating security levels [1]. This is also known as the no read up principle and the no write down principle.

2.2. The Bell-LaPadula Model

A Bell-LaPadula policy [2] is a policy combines a multi-level security policy with a discretionary access control policy. Bell-LaPadula policies enforce both multi-level security policies (to ensure the confidentiality requirements) and the discretionary access control policies (to ensure the flexibility of access control policies). When enforcing a Bell-LaPadula policy, a subject is allowed to have access to an object if and only if the subject is allowed to have access to the object by both the multi-level security policy and the discretionary access control policy.

In the Bell-LaPadula Model, a system consists of the following components.

- S is the set of subjects of the system.
- O is the set of objects of the system.
- K is the set of security levels of the system.
- $\leq_d$ is the dominance relation defined over the set of K, which is a partial order.
- F is a set of triples (f_s, f_o, f_c) where f_s and f_c are the functions mapping subjects to their maximum security levels and current security levels, and f_o is a function mapping objects to their security levels.
- $P = \{e, r, a, w\}$ is the set of access methods, representing executing, reading, appending, and writing respectively.
- $B \subseteq S \times O \times P$ is the set of current accesses that are available to users.
- $M \subseteq S \times O \times P$ is the discretionary access control set.

Bell-LaPadula models a system state as a quadruple (b, m, f, h), where $b \subseteq B$ contains the current allowed access to users, $m \subseteq M$ is the current discretionary access control matrix, $f = (f_s, f_o, f_c) \in F$ is the security level function triple, and h is the object hierarchy.

The Bell-LaPadula model claims that if a system satisfies the following three properties, the system is secure.

Simple Security Property $(s, o, p) \in S \times O \times P$ satisfies the simple security property if and only if one of the following two rules holds:

- $p = e$ or $p = a$
- $p = r$ or $p = w$ and $f_o(o) \leq_d f_c(s)$

***-Property** $\forall(s, o, p) \in S \times O \times P$ satisfies the *-property if and only if one of the following rules holds:

- $p = e$
- $p = a$ and $f_c(s) \leq_d f_o(o)$
- $p = w$ and $f_c(s) = f_o(o)$
- $p = r$ and $f_o(o) \leq_d f_c(s)$

Discretionary Security Property A system state (b, m, f, h) satisfies the discretionary security property if and only if, $\forall(s, o, p) \in b$, $(s, o, p) \in m$.

2.3. Role based Access Control

Sandhu et al. [11] identified the motivation of using roles as basic constructs in access control models and introduced several models of RBAC, and conceptualized RBAC into four different models,the base model, the hierarchical model, the constrained model and the consolidated model.

Ferraiolo and Kuhn [4] presented a detailed description of the RBAC model, and provided the definition of roles, transactions and a formalization of RBAC. Roles were defined by using a set of transactions, and transactions were a set of high level activities that users could perform. A user had the right to perform a transaction if the transaction was a permitted transaction of his current active role set.

Oppliger et al. [7] and Chadwick et al. [3] proposed two similar ways of implementing RBAC based on Attribute Certificates. Attribute certificates are used as protected tokens to convey attribute information.

Gavrila and Barkley [5] formally specified the role management of a RBAC system, and defined the consistency of a RBAC system using a set of properties. Gavrila and Barkley also showed that given a consistent RBAC system, performing legitimate management operations maintained the consistency of the system.

NIST proposed a reference model of RBAC [8] which was subsequentlyapproved as an American national standard [1]. The RBAC reference model is defined in terms of four different model components which are: the Core RBAC, the Role Hierarchy, Static Separation of Duty Relations, and Dynamic Separation of Duty Relations. The Core RBAC specifies the essential elements of the RBAC model which are the minimum set of elements. The other three components can be integrated with the Core RBAC component to add more features.

Core RBAC [1] consists of five basic elements, which are the U, R, A, O, and S, and five relations, which are RA, PA, $U - S$, $S - R$, and PRM.

U refers to the set of legitimate users in the system. R is the set of roles existing in the system. A is the operations that are recognized by the system, and O is the set of objects that are protected by the system. S is the set of sessions in the system that are handling users' requests.

Operations and objects are bound to each other to construct permissions, denoted by PRM where $PRM \subseteq A \times O$. A permission is an approval for performing an operation

[1]The *write* operation here means **appending** which involves only adding new information but does not involve any retrieval of information.

on a specified target. Users are allocated roles, as specified by the RA relation where $RA \subseteq U \times R$, which is the user assignment relation. Permissions are allocated to roles, which are specified by the permission assignment PA relation where $PA \subseteq PRM \times R$. $U - S(s : S) \rightarrow U$ is a mapping of a session onto the corresponding user, and the $S - R(s : S) \rightarrow 2^R$ is a mapping of a session onto a set of roles.

The authorization decision making function $CheckAccess$ takes as input the current session, the requested operation, and the object that is the target of the operation. The $CheckAccess$ function will return a Boolean value as a result to indicate whether the request is authorized or not.

The Role Hierarchy is a component that can enhance the Core RBAC component by specifying relations between roles to support a role hierarchy. With role hierarchies, some roles can be superiors of other roles. Superior roles acquire the permissions allocated to subordinate roles.

The Static Separation of Duty component enhances the Core RBAC component with the capability of imposing constraints on the user assignment to roles, so that users can not be assigned memberships of conflicting roles.

The Dynamic Separation of Duty component augments the Core RBAC component with the capability of imposing constraints on the set of roles that can be activated by a user's session. In this way, users cannot activate certain specified roles at the same time though they can be members of all these roles.

3. RBAC Implementation of The Bell-LaPadula Security Policies

This section presents an algorithm that can map a given Bell-LaPadula security policy to a RBAC policy.

The mapping of a Bell-LaPadula security policy to a RBAC policy is an algorithm of two parts. The first part is to simulate the mandatory access control of a given Bell-LaPadula security policy using a RBAC policy. The second part is to tailor the previously constructed RBAC policy to enforce the discretionary access control of the Bell-LaPadula security policy.

3.1. Simulating MAC

To simulate the mandatory access control in a Bell-LaPadula security policy, e and w access related permissions are organized by roles without subordinate roles or superior roles, and r and a access related permission are organized by two disjoint sets of roles with appropriate hierarchies. The hierarchies are constructed in accordance with the order of the security levels.

Step 1 : User Mapping. User mapping is the process of constructing a set of users in the RBAC Policy based on the set of subjects $M.S$ in the given military security policy.
The mapping can be denoted by a bijective function as follows:$\psi_s : M.S \rightarrow P.U$
To each subject $s \in M.S$, there exists a $u \in P.U$ such that $u = \psi_s(s)$ and $s = \psi_s^{-1}(u)$ where ψ_s^{-1} is the inverse function of ψ_s.

Step 2 : Object Mapping. Object mapping is to construct $P.O$, the set of objects in the RBAC policy, based on the object set $M.O$ in the military security policy. Each of the objects in the RBAC policy represents one and only one object in $M.O$ of the military security policy. This can be formalized as the following: $\psi_o : M.O \rightarrow P.O$ where ψ_o is a bijective function.

Step 3: Role Mapping. R, the set of roles in the RBAC policy, consists of four disjoint set of roles constructed based on the set of subjects and the set of security levels in the Bell-LaPadula security policy, which are the executing role set R_e, the appending role set R_a, the reading role set R_r and the writing role set R_w. Four bijective functions f_{re}, f_{rr}, f_{ra}, and f_{rw} are defined as below to implement the mapping.

- $\psi_{re} : M.S \rightarrow R_e$. To each subject $s \in M.S$, a corresponding role $r_e = \psi_{re}(s)$ is constructed in the set R_e. r_e is the executing role of the subject s.
- $\psi_{rr} : M.K \rightarrow R_r$. To each security level $k \in M.K$, a corresponding role $r_r = \psi_{rr}(k)$ is constructed in the set R_r. r_r is the reading role of the security level k.
- $\psi_{ra} : M.K \rightarrow R_a$. To each security level $k \in M.K$, a corresponding role $r_a = \psi_{ra}(k)$ is constructed in the set R_a. r_a is the appending role of the security level k.
- $\psi_{rw} : M.K \rightarrow R_w$. To each security level $k \in M.K$, a corresponding role $r_w = \psi_{rw}(k)$ is constructed in the set R_w. r_w is the writing role of the security level k.

Therefore, each subject is mapped to a role in R_e, and each security level is mapped into three different roles in R_r, R_a, and R_w respectively. Roles in R_e will be used to manage permissions related to e access, and roles in R_r, R_a, and R_w will be used to managed permissions of r, a, and w access respectively.

Step 4 : Action Mapping. The set of possible accesses are mapped to the RBAC policy as the Action set $P.A = \{e, r, a, w\}$ where e, r, a, and w have the same meanings as they have in the Bell-LaPadula model. The mapping function $f_a(x) = x$ where f_a maps accesses in the Bell-LaPadula model to actions in the RBAC model.

Step 5 : Permission Construction. The permission set of the RBAC policy, $P.PRM$, is constructed as follows: $P.PRM = P.A \times P.O$

Step 6 : Permission Allocation. Permission allocation mapping is to construct the permission allocation relation in RBAC policies. The RBAC permission allocation is divided into four different mappings, which maps the permissions related to e, r, a, and w to the roles in R_e, R_r, R_a, and R_w respectively. The mapping can be formalized as below.

- To a given subject s, if there exists an object o such that $(s, o, e) \in b$, then $PA = PA \cup \{(r_e, (e, o_o))\}$ where $r_e = \psi_{re}(s)$ and $o_o = \psi_o(o)$. Thus if a subject s is authorized to have e access to the object o, the subject s's executing role r_e is allocated with the permission (e, o_o).
- $\forall k \in K$ and $\forall o \in O$, if $f_o(o) = k$, then $PA = PA \cup (r_r, (r, o_o))$ where $r_r = \psi_{rr}(k)$ and $o_o = \psi_o(o)$. Thus for each security level k, its reading role r_r is allocated with the permission of having r access to all the objects whose original objects are of the security level k.
- $\forall k \in K$ and $\forall o \in O$, if $f_o(o) = k$, then $PA = PA \cup (r_a, (a, o_o))$ where $r_a r = \psi_{ra}(k)$ and $o_o = \psi_o(o)$. Thus for each security level k, its appending role r_a is allocated with the permission of having a access to all the objects whose original objects are of the security level k.
- $\forall k \in K$ and $\forall o \in O$, if $f_o(o) = k$, then $PA = PA \cup (r_w, (w, o_o))$ where $r_w = \psi_{rw}(k)$ and $o_o = \psi_o(o)$. Thus for each security level k, its writing role r_w is allocated with the permission of having w access to all the objects whose original object is of the security level k.

Step 7 : Role Assignment Mapping. Role assignment mapping constructs the role assignment relation of the RBAC policy based on the given Bell-LaPadula policy.

- $\forall s \in S$, $(r_e, u) \in RA$ where $u = \psi_u(s)$ and $r_e = \psi_{re}(s)$.
- $\forall s \in S$, $(r_r, u) \in RA$ where $u = \psi_u(s)$ and $r_r = \psi_{rr}(f_s(s))$.
- $\forall s \in S$, $(r_a, u) \in RA$ where $u = \psi_u(s)$ and $r_a = \psi_{ra}(f_s(s))$.
- $\forall s \in S$, $(r_w, u) \in RA$ where $u = \psi_u(s)$ and $r_w = \psi_{rw}(f_s(s))$.

Let $k = f_s(s)$. This mapping assigns each user u mapped from the subject s four role memberships, which are the executing role r_e of the subject s, the reading role r_r, the appending role r_a and the writing role r_w of the security level k.

Step 8 : Hierarchy Mapping. The Role Hierarchy is mapped based on the relation $\leq_d$ of the military policy. The $\leq_d$ relation in the military policy allows reading permission to increase from subordinate security levels to superior security levels, and appending permission to decrease from subordinate security levels to superior security levels.

The mapping creates relations between the roles in the reading role set R_r and relations between the roles in the appending role set R_a.

- The construction of relations between the reading roles is achieved by the function $\psi_{hr} \ : \leq_d \rightarrow H$, where the function is defined as: Let j and k both be in K and $j \leq_d k$,$(j_r, k_r) = \psi_{hr}((j, k))$, where $j_r = \psi_{rr}(j)$ and $k_r = \psi_{rr}(k)$.
- The construction of relations between the appending roles is achieved by the function $\psi_{ha} \ : \leq_d \rightarrow H$, where the function is defined as: Let j and k both be in K and $j \leq_d k$, $(k_a, j_a) = \psi_{ha}((j, k))$, where $j_r = \psi_{rr}(j)$ and $k_r = \psi_{rr}(k)$.

In summary, the first part of the algorithm maps a Bell-LaPadula security policy's MAC to a RBAC policy. Subjects are mapped as users in the RBAC policy and objects are mapped as objects in the RBAC policy. Four sets of disjoint roles are created, and they are used to organize the permissions related to e, r, a and w access respectively. A user is assigned membership to one role in each of the four sets of roles according to the corresponding subject of the user and the security level of the subject.

3.2. Enforcing DAC

The second part of the algorithm enforces the DAC part by tailoring the permission allocations constructed by the first part, which are authorized by the MAC part but not authorized by the DAC part. After the tailoring, users are assigned to roles that can invoke no more permissions than authorized by the Bell-LaPadula security policy. Details of the algorithm are as follows.

Modifying e related permission allocation. To each subject $s \in S$, if there exists an object o such that $(s, o, e) \in b$ but $(s, o, e) \notin m$, the subject is granted e access to the object o by the mandatory access control but not by the discretionary access control. In this case, the related permission shall be removed. The removal can be formalized as the following: $PA = PA - \{(\psi_{re}(s), (e, \psi_o(o))) | \forall s \forall o, \ (s, o, e) \in b \wedge (s, o, e) \notin m\}$

Modifying r related permission allocation. To any given subject $s \in S$, let $r_r = \psi_{rr}(f_s(s))$ be the reading role of s. Modifying r related permission allocation is to tailor the r permission allocation to a subject s such that only r permissions that are granted by both the mandatory access control and the discretionary access control will be allocated. Let R_s denotes the set of roles containing the role r_r and all its subordinate roles.

If there does not exist an object $o \in O$ such that $(s, o, r) \in b$ but $(s, o, r) \notin m$, the subject s is granted the same permissions to have r access by the mandatory access control as by the discretionary access control. In this case, there is no need to modify the r related permission allocation of the reading role r_r of the subject s.

If there exists at least one object o such that $(s, o, r) \in b$ but $(s, o, r) \notin m$, the following algorithm shall be applied to modify the r related permission allocation to the subject s.

- Construct a new role r_{sr}, and put it into the role set R by $R = R \cup \{r_{sr}\}$.
- Modify the role assignment relation RA. $RA = RA - \{(r_r, u_s)\} \cup \{(r_{sr}, u_s)\}$. This is to assign the user u_s of the subject s to be a member of the new role r_{sr}, and revoke the user u_s's role membership of the role r_{rr}.
- Allocate permissions to the role r_{sr}. To each object $o \in O$ that $(s, o, r) \in b$ and $(s, o, r) \in m$, then the permission $(r, f_o(o))$ is allocated to the role r_{sr}. Therefore $PA = PA \cup \{(r_{sr}, (r, f_o(o)))|\forall o \in O, \ (s, o, r) \in b \wedge (s, o, e) \in m\}$.

Modifying a related permission allocation. To each subject $s \in S$, let $r_a = \psi_{ra}(f_s(s))$ be the appending role of s. Modifying a related permission allocation is to tailor the a related permission allocation to a subject s's corresponding user such that only those a related permissions granted by both the mandatory access control and the discretionary access control will be allocated.

If there does not exist an object $o \in O$ such that $(s, o, a) \in b$ but $(s, o, a) \notin m$, the subject s is granted the same permissions of a access by the mandatory access control as granted by the discretionary access control. In this case, there is no need to modify the a related permission allocation of the appending role r_a of the subject s.

If there exists at least one object o such that $(s, o, a) \in b$ but $(s, o, a) \notin m$, the following algorithm shall be applied to modify the a related permission allocation for the subject s.

- Construct a new role r_{sa}, and put it into the role set R by $R = R \cup \{r_{sa}\}$.
- Modify the role assignment relation RA. $RA = RA - \{(r_a, u_s)\} \cup \{(r_{sa}, u_s)\}$. This is to assign the user u_s of the subject s to be a member of the new role r_{sa}, and revoke the user u_s's role membership of the role r_{ra}. The new role r_{sa} is now the appending role of the subject s.
- Allocate permissions to the role r_{sa}. To each object $o \in O$ that $(s, o, a) \in b$ and $(s, o, a) \in m$, then the permission $(a, f_o(o))$ is allocated to the role r_{sa}. Therefore $PA = PA \cup \{(r_{sa}, (a, f_o(o)))|\forall o \in O, \ (s, o, a) \in b \wedge (s, o, a) \in m\}$.

Modifying w related permission allocation . To each subject $s \in S$, let $r_w = \psi_{rw}(f_s(s))$ be the writing role of s. Modifying w related permission allocation is to tailor the w related permission allocation to a subject s's corresponding user such that only those w related permissions granted by both the mandatory access control and the discretionary access control will be allocated.

If there does not exist an object $o \in O$ such that $(s, o, w) \in b$ but $(s, o, w) \notin m$, the subject s is granted the same permissions of w access by the mandatory access control as granted by the discretionary access control. In this case, there is no need to modify the w related permission allocation of the writing role r_w of the subject s.

If there exists at least one object o such that $(s, o, w) \in b$ but $(s, o, w) \notin m$, the following algorithm shall be applied to modify the w related permission allocation for the subject s.

- Construct a new role r_{sa}, and put it into the role set R by $R = R \cup \{r_{sw}\}$.
- Modify the role assignment relation RA. $RA = RA - \{(r_w, u_s)\} \cup \{(r_{sw}, u_s)\}$. This is to assign the user u_s of the subject s to be a member of the new role r_{sw}, and revoke the user u_s's role membership of the role r_{rw}. The new role r_{sw} is now the writing role of the subject s.
- Allocate permissions to the role r_{sw}. To each object $o \in O$ that $(s, o, w) \in b$ and $(s, o, w) \in m$, then the permission $(w, f_o(o))$ is allocated to the role r_{sw}. Therefore $PA = PA \cup \{(r_{sw}, (a, f_o(o)))|\forall o \in O, \ (s, o, w) \in b \wedge (s, o, w) \in m\}$.

4. Related Work

Sandu [10] argued that lattice based access control model can be simulated by RBAC policies using dual role hierarchies. Permissions for reading and writing are allocated to roles in two separate sets of roles, and each user is assigned as members of two matching roles in the two separate sets of roles. The role hierarchy associated with reading is constructed in the same order as the security lattice, while the role hierarchy associated with writing is constructed in an inverse order as the security lattice. Sandu has not presented a proof of the algorithm, though the examples were convincing. Neither did Sandu provide any ways of analyzing if a RBAC policy complies with a given security policy based on lattice based access control.

Nyanchama and Osborn [6] presented a way to model mandatory access control in RBAC systems. Nyanchama and Osborn argued that this could benefit from the flexible permission allocation management provided by RBAC models as well as the powerful modeling capability of RBAC models. MAC is realized in RBAC by treating roles as security levels of the MAC system and imposing an acyclic information flow requirement in the way roles are organized. The acyclic information flow requirement is based on five constraints, which ensure that information can not flow from a higher security level to a lower one.

The above mentioned research only investigated the relationship between the RBAC model and the MAC model, but did not cover the relationship between the RBAC model and the Bell-LaPadula model, which is the focus of this work.

5. Conclusions

This paper identifies the need for enforcing Bell-LaPadula security policies using the RBAC model and investigates the potential way of representing Bell-LaPadula security policies using the RBAC model. The investigation results in the development of a mapping algorithm that can map a given Bell-LaPadula security policy into a RBAC security policy.

The mapping algorithm shows that Bell-LaPadula security policies can be represented by RBAC security policies. Therefore applications built based on RBAC model have the potential to be applied in military environments.

The contribution of this work is as the follows. Firstly, we identify the need for modeling the Bell-LaPadula model using RBAC, which is caused by the industry having to implement separate systems for commercial applications and for military applications. Secondly, we develop a mapping algorithm that can construct a RBAC security policy based on a given Bell-LaPadula security policy. The mapping algorithm shows that it is possible to model military security policies using the RBAC model. Besides the academic importance of showing the flexibility and power of the RBAC model, it is also important to industry, as applications built based on the RBAC model can be deployed in military environments and enforce Bell-LaPadula policies. There is no need to build separate applications which use the Bell-LaPadula model just for military scenarios.

A weakness of our approach is that the RBAC policies constructed by the suggested algorithm are not very elegant, as there are no optimizations of the role sets and the role hierarchies used by the policies. There may be some unused roles in the policies and the number of roles can be unnecessarily large. But the existence of such an algorithm indicates that there could be other algorithms that can map Bell-LaPadula security policies into RBAC policies, which may construct RBAC policies that are more elegant than those constructed by the suggested algorithm. This is a subject for further research.

References

[1] American National Standards Institute, Inc. Role-Based Access Control. ANSI INCITS 359–2004., February 2004.

[2] D. E. Bell and L. J. LaPadula. Computer security model: Unified exposition and multics interpretation. Technical report, MITRE Corp., Bedford, MA, Tech. Rep. ESD-TR-75-306, June 1975.

[3] D.W.Chadwick, A. Otenko, and E.Ball. Implementing role based access controls using X.509 attribute certificates. *IEEE Internet Computing*, pages 62–29, March 2003.

[4] D. Ferraiolo and R. Kuhn. Role-based Access Control. In *Proceedings of 15th National Computer Security Conference*, pages 554–563, 1992.

[5] S. Gavrila and J. Barkley. Formal Specification for Role Based Access Control User/Role and Role/Role Relationship Management. In *Third ACM Workshop on Role-Based Access Control*, pages 81 – 90, 1998.

[6] M. Nyanchama and S. Osborn. Modeling mandatory access control in role-based security systems. In *Proceedings of the ninth annual IFIP TC11 WG11.3 working conference on Database security IX : status and prospects*, pages 129 – 144, 1996.

[7] R. Oppliger, G. Pernul, and C. Strauss. Using attribute certificates to implement role-based authorization and access controls. In S. T. K. Bauknecht, editor, *Sicherheit in Informationssystemen (SIS 2000)*, pages 169–184, Zurich, 2000.

[8] R. Sandhu, D. Ferraiolo, and R. Kuhn. The NIST Model for Role Based Access Control: Towards a Unified Standard. In *5th ACM Workshop on Role Based Access Control*, pages 47–63, July 2000.

[9] R. S. Sandhu. Lattice-based access control models. *IEEE Computer*, 26(11):9–19, 1993.

[10] R. S. Sandhu. Role hierarchies and constraints for lattice-based access controls. In *ESORICS '96: Proceedings of the 4th European Symposium on Research in Computer Security*, pages 65–79, London, UK, 1996. Springer-Verlag.

[11] R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, 1996.