Engineering Authority and Trust in Cyberspace: The OM-AM and RBAC Way

Ravi Sandhu
ISE Department, MS 4A4
George Mason University
Fairfax, VA 22030
sandhu@gmu.edu, www.list.gmu.edu

Abstract

Information systems of the future will be large-scale, highly decentralized, pervasive, span organizational boundaries and evolve rapidly. Effective security in this cyberspace will require engineering authority and trust relationships across organizations and individuals. In this paper we propose the four-layer OM-AM framework for this purpose. OM-AM comprises objective, model, architecture and mechanism layers in this sequence. The objective and model (OM) layers articulate *what* the security objectives and tradeoffs are, while the architecture and mechanism (AM) layers address *how* to meet these requirements. The hyphen in OM-AM emphasizes the shift from what to how. These layers are roughly analogous to a network protocol stack with a many-to-many relationship between successive layers, and most certainly do not imply a top-down waterfall-style software engineering process. OM-AM is an excellent match to the policy-neutral and flexible nature of role-based access control (RBAC). This paper describes and motivates the OM-AM framework and presents a case study in applying it in a distributed RBAC application.

1 INTRODUCTION

Future information systems will be as different from today's systems as the Internet is from the telegraph. It is our obligation to understand how to effectively secure these systems as they develop and get deployed. Given the dramatic changes implied by the telegraph-Internet analogy, we cannot predict with much certainty exactly what form future information systems will take. Nevertheless there are salient characteristics we can postulate with confidence. Information systems of the future will be large-scale, highly decentralized, pervasive, span organizational boundaries and evolve rapidly. Current security doctrine is simply incapable of dealing with this complex and fluid environment that is inevitably emerging.

Cyberspace security is fundamentally about the control of authority and trust. Authority and trust are intermingled concepts. The authority to do something is coupled with trust that the privilege will be exercised appropriately. In particular, the authority to grant and revoke authority to other users and entities in the system is predicated on trust that this authority will be responsibly used. Authority needs to be administered and enforced, while trust needs to be monitored and verified. Enforcement of authority is done by system components which must be trusted to function correctly. In turn, without trusted people and components authorization cannot be effectively enforced. Future information systems must deal with this intermingling and mutual dependence of authority and trust in large-scale decentralized and distributed systems. We call this the problem of engineering authority and trust in cyberspace.

In this paper we propose a four-layer approach to address this problem. The four layers are objective, model, architecture and mechanism as shown in figure 1, surrounded by a sea of assurance which permeates all layers. Objective and model are concerned with articulating *what* the security objectives and tradeoffs are, while architecture and mechanism address *how* to meet these requirements. We call this the OM-AM framework or, more informally, the OM-AM way. The hyphen in OM-AM emphasizes the shift from *what* to

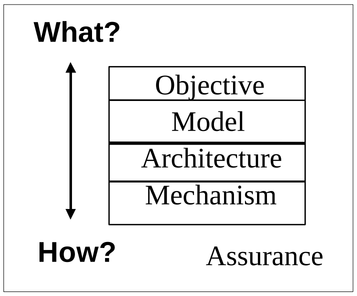

Figure 1: The OM-AM Framework

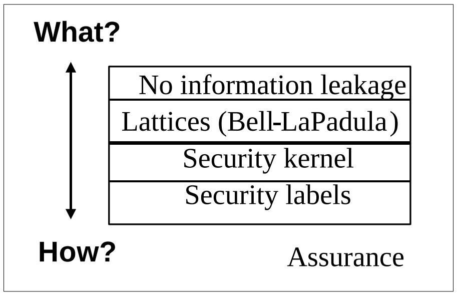

Figure 2: The OM-AM Framework for MLS Systems

how, as does the thicker line in figure 1. OM-AM is a framework. To pursue it in a given context takes a great deal of additional detail and supporting tools and techniques. Our focus here is on applying OM-AM in an RBAC context. In fact we were led to OM-AM because of our work in the RBAC arena. We have subsequently applied OM-AM in non-RBAC contexts quite productively, but this is outside the scope of this paper.

The OM-AM framework has an intuitive simplicity and appeal. It is so natural that it is almost self-evident once it has been articulated. Nonetheless OM-AM is original and valuable. Its roots lie in the long-standing objective-mechanism (or policy-mechanism) distinction in the security literature (see, for example, [LCC+75]). As we will argue there is a strong need to bring in the additional model and architecture layers. Existing security research and practice all too often focuses exclusively on one layer or confuses issues from multiple layers. OM-AM clearly demarcates the issues at each layer while allowing us to identify dependencies between different layers. This may be the biggest payoff of OM-AM.

Layered approaches to security have certainly been suggested in the past. The protection rings of MULTICS [Sal74] and the stacking of applications in the Trusted Database Interpretation [Dep91, Dep85] are just two examples of familiar layered approaches. However, the layering of OM-AM is very different from these previous approaches. In OM-AM we are not trying to build one abstraction on top of another. Instead we are dealing with very different kinds of concepts at each layer and pursuing very different activities. OM-AM does not imply a waterfall-style software engineering process that goes top down from objective to model to architecture to mechanism. To the contrary, the interaction between layers is more complex and not so much process driven. The name layer is chosen in rough analogy to layers in a network protocol stack. Each layer has a distinct function which can to some degree be carried out independently of adjacent layers. Each layer needs a different set of tools, notation and abstractions to articulate its concerns. Also the mapping between adjacent layers is many-to-many. In terms of OM-AM a single objective can be supported by multiple models, while a single model can support multiple policies. Likewise for the model-architecture and architecture-mechanism relationships. OM-AM is more flexible and attuned to current technology trends than previous layered approaches to security engineering.

In this paper we introduce and motivate the OM-AM framework. As we will see OM-AM is particularly compelling in context of role-based access control (RBAC). This is due to the policy-neutral and flexible nature of RBAC. We demonstrate this by presenting a case study in applying OM-AM to a distributed RBAC system.

2 THE OM-AM FRAMEWORK

Objective, model, architecture and mechanism are highly overloaded words and mean different things to different communities. In OM-AM we have used one word to name each layer, for sake of simplicity. Requirement or policy might be alternate names for the objective layer, while protocols could be used instead of mechanism. OM-AM does not seek to give airtight meaning to these words but rather is an informal and intuitive engineering framework. There is some fuzziness in exactly where we draw the boundary between successive layers. Precision in delineating the boundaries is not the end goal.

OM-AM roots lie in the long-standing objective-mechanism (or policy-mechanism) distinction. Why do we need model and architecture layers? In the pre-network era there was hardly any distinguishable architecture to speak of. In distributed systems, the concept of architecture allows us to describe the high-level security design in terms of its major components, servers, brokers, etc., and their interrelationships. In early security work the objective and model layers were typically fused into one, because the objective was either multilevel or discretionary security. The apparent simplicity of the past cannot be sustained, and we must deal with all four layers explicitly. Existing security research and practice all too often focuses exclusively on one layer or confuses issues from multiple layers. In contrast with previous layered approaches, OM-AM is a logical and obvious extension to the classical objective-mechanism-assurance triad.

Figure 2 illustrates OM-AM in context of classical multilevel security (MLS). Multilevel security is concerned with preventing information leakage in a classified military or national security setting. Thus there is a fixed objective of one-directional information flow which is being pursued. To articulate this objective formally we have the well-known lattice-based access control (LBAC) models [San93], also commonly known as the Bell-LaPadula model or mandatory access control. The standard architecture for implementing LBAC is a security kernel [Dep85] and a variety of mechanisms are used for this purpose. Multilevel security has been studied for three decades, and it is a positive confirmation that this classic area fits within OM-AM. In this context there is one objective and one model.[1] There are however multiple architectures in modern distributed systems in addition to the standard security kernel approach [Not94]. There are also many mechanisms that have been used by different implementors.

In much of the early work on multilevel security there was only one objective, one model and one architecture and a multiplicity of mechanisms. This has tended to foster a view of security engineering close to the classic top-down software engineering waterfall. Even with the recognition of multiple MLS architectures we still have only one objective and model.

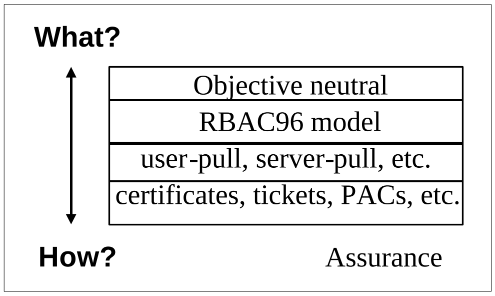
What?
Objective neutral
RBAC96 model
user-pull, server-pull, etc.
certificates, tickets, PACs, etc.
How?
Assurance

Figure 3: The OM-AM Framework for RBAC Systems

The four layer framework becomes more compelling when we look at RBAC as illustrated in figure 3. In this case we have a multiplicity of security objectives. The ability to configure security objectives and policy is one of the main features of RBAC. There are a number of RBAC models that have been published [FK92, FCK95, FBK99, Gui95, NO95, NO99, RS98, SCFY96, San98a, TDH92, ZSS99]. They differ in details but they all share some common characteristics. The RBAC96 model was the first comprehensive RBAC model to be published and has emerged as the best known and most authoritative model [SCFY96]. RBAC96 is very rich in the scope of security objectives that it can support. In particular it has been shown that RBAC96 can be configured to enforce LBAC (or MLS) on one hand and discretionary access control [OSM00, San96, SM98] on the other. This is strong confirmation of the tremendous range of policies that RBAC96 can accommodate. There are many architectures for implementing RBAC in distributed systems, including the user-pull and server architectures [PS99, PSG00] (as will be discussed in the next section). Conversely, each of these architectures can support models other than RBAC such as attribute-based access control. Finally, a variety of mechanisms and protocols can be used to support these architectures.

The OM-AM framework has emerged as a consequence of RBAC research pursued by us at all four layers, since 1993. Our initial emphasis was on RBAC models [SCFY96, San98b, SBM99], involving experimental research on specific platforms, including the Web, Oracle, Windows NT and Unix [SA98a, SA98b, SP98, SB99, PS99, PSG00]. This led to the formula-

[1] Strictly speaking this statement oversimplifies the situation. Issues of trusted subjects and inference, inference and aggregation, authorized downgrading and encryption, muddle the single-minded objective of one-directional information flow. Also there is a proliferation of non-interference style models to formalize the goal of one-directional information flow beyond LBAC [McL94].

tion of alternate architectures. Concurrently research on security objectives and their realization in RBAC was also pursued. Early in 1999 it became clear that these activities could be nicely packaged in the OM-AM framework. Although the OM-AM approach emerged from our RBAC research, we have subsequently applied to other security issues and have found it to be a very productive approach. Our focus in this paper is on its application in RBAC systems which we illustrate by means of a case study.

3 DISTRIBUTED RBAC (DRBAC) CASE STUDY

This case study deals with a distributed application containing multiple autonomous entities. It is based on the requirements of a real application for a client who will remain unnamed. We discuss this case study with respect to each of the four OM-AM layers in turn.

3.1 DRBAC OBJECTIVES

The system consists of a number of physical sites, each of which has a number of simulation-models.[2] Each simulation-model is an autonomous entity with its own security administrators. Access control to services of each simulation-model will be enforced with respect to the roles possessed by the user attempting to make access. A particular simulation-model may recognize only a subset of the entire set of roles in the system.

The main concern of the policy is with administration of the user-role and role-permission relations. The security administrators of each simulation-model will determine what permissions are assigned to each role on that simulation-model. Revoking these permissions is also entirely under control of these security administrators. User-role assignment requires approval of at least one security administrator of *all* simulation-models where that role has non-empty permissions. Conversely, a single security administrator can revoke a user from a role (provided the simulation-model of that administrator has non-empty permissions for that role).

It is assumed that the security administrators of a simulation-model can assign permissions for that simulation-model to any role at any time. In particular permissions can be granted to a role X even if X currently does not have any permissions for that simulation-model. By doing so the security administrator implicitly accepts all users of X and gains the power to revoke their membership from X.[3]

A user will employ only one role at a time to access a particular simulation model. The scale of the system is as follows.

- Approximately a dozen physical locations
- Approximately 2-3 simulation-models/location
- Fewer than 100 roles structured in a very shallow hierarchy
- Fewer than 100 users
- Moderate rate of change

3.2 DRBAC MODEL

The above objectives are stated rather informally, but they are clearly role-oriented and focus on role administration in a system of autonomous systems. The purpose of a model is to formalize these objectives. The preferred approach is to build upon existing models, rather than reinventing the wheel every time. Several general RBAC models have been proposed in the literature. We construct the DRBAC model by building upon RBAC96 [SCFY96].[4]

The process by which a general model such as RBAC96 is reshaped to yield a specific model such as DRBAC is called *customization.* One of the basic assumptions in RBAC96 is the existence of a single all-powerful security administrator. This assumption is fundamentally inconsistent with the DRBAC objectives given above. Fortunately it turns out this assumption is not critical to RBAC96 and can be easily dropped without damaging the integrity of the model. The main customization of RBAC96 is in the administrative component.

The basic definition of RBAC96 are used essentially unchanged. We have limited each session to a single active role as indicated in the policy. This assumption

[2] We will be very careful to use the "simulation-model" to distinguish these models from the RBAC models. The term "model" will be used by itself only to mean access control model or security model.

[3] This policy has a somewhat undesirable feature. It is possible for any security administrator Alice to revoke any user Bob from any role X. If X already has non-empty permissions in Alice's simulation-model she can revoke Bob from X by the stated revocation policy. If X does not have non-empty permissions, she can assign some permission to X to make it have non-empty permissions and then revoke Bob from X. Then, she can revoke the permissions from X. Security administrators are presumably trusted not to do such mischief. A different or more elaborate policy can fix this problem but it may not be justified in this application. Misuse detection technology can also be used to detect such mischief.

[4] We assume the reader is generally familiar with RBAC96.

can be easily changed if desired without much impact on the rest of the model. Thus we start with the following definition.

Definition 1 [DRBAC: RBAC96 Components] The DRBAC model includes the following RBAC96 components.

- U, a set of users
  R, a set of (regular) roles
  P, a set of (regular) permissions
  S, a set of sessions
- $UA \subseteq U \times R$, user to role assignment relation
- $PA \subseteq P \times R$, permission to role assignment relation
- $RH \subseteq R \times R$, partially ordered role hierarchy
- $user : S \to U$, maps each session to a single user (which does not change)

  $role : S \to R$ maps each session s_i to a single role (which does not change)

  session s_i has the permissions of all roles r'' junior to $role(s_i)$, that is $\{p \mid (\exists r'' \leq role(s_i))[(p, r'') \in PA]\}$
- each session can have only a single role as stipulated above

The definition of DRBAC is completed below.

Definition 2 [DRBAC: RBAC96 Customization] The DRBAC model consists of the following extensions to its RBAC96 components defined above.

- $SM = \{sm_1, \ldots, sm_k\}$, a set of simulation-models
- $OP = \{op_1, \ldots, op_l\}$, a set of operations
- The set of permissions P is defined as $P = SM \times OP = \{(sm_i, op_j) \mid sm_i \in SM, op_j \in OP\}$
- $sm : P \to SM$, a many-to-one function that maps each permission to the simulation-model to which it applies so that $sm(p) = sm((sm_i, op_j)) = sm_i$
- $SMA = \{sma_1, \ldots, sma_k\}$, a set of administrative roles one for each simulation model
- The administrative roles are disjoint from the regular roles, so $R \cap SMA = \emptyset$. Also each session has exactly one regular role or administrative role (but not both) associated with it.
- $admin : SM \leftrightarrow SMA$, a one-to-one function that maps each simulation-model to an administrative role
- Each simulation-model sm_i has a unique user designated as its *chief security administrator*. The chief security administrator is the only one who can assign and revoke users to and from the corresponding administrative role $admin(sm_i)$.[5]
- Permission p can be assigned to or revoked from role r by a user who is a member of the administrative role $admin(sm(p))$. No other administrative role or user is authorized to perform this task.
- User u can be assigned to a role $r \in R$ if and only if this is approved by at least one member of every administrative role $ar \in SMA$ for which there exists $p \in P$ such that $(p, r) \in PA$ and $admin(sm(p)) = ar$.
- User u can be revoked from a role $r \in R$ if and only if the revocation is done by any member of any administrative role $ar \in SMA$ for which there exists $p \in P$ such that $(p, r) \in PA$ and $admin(sm(p)) = ar$.

The resulting model still leaves open a number of choices that need to made when a particular DRBAC system is deployed. This includes definition of sets such as SM and OP, and functions such as *admin*. The purpose of such a model is to help refine and make more precise the informal security goals of the objective layer. By doing so we also discover areas of omission and ambiguity. Eventually we might hope to use formal tools for this discovery process but in the current state-of-art we expect this to be predominantly an informal process.

3.3 DRBAC ARCHITECTURES

Next we identify different architectures for enforcing DRBAC and discuss issues arising in these architectures. There are four components whose architectures we need to consider. For convenience we consider each one separately but there are interdependencies. We also note that that the authentication architecture is not considered here. Our discussion is limited to the

[5] The chief security administrator can, of course, assign herself to the administrative role. More importantly, the chief security administrator can assign other users to the administrative role thus bringing more administrators as needed. These users, however, do not get the ability to add or revoke other administrative users. This power is solely vested in the chief security administrator. Various other polices are possible in this regard, but ultimately the model designers have to choose one. All too often such policy decisions are deferred to the mechanism layer or dealt with in ad hoc ways. The point of the model layer in OM-AM is to help identify these policy issues and deal with the important ones at the model layer.

authorization architecture. Moreover, hybrid architectures are also possible.

3.3.1 Permission-Role Assignment

By definition of the policy, permission-role assignment to regular roles is a local matter at each simulation-model. Thus each simulation model can do this task in whatever way it likes. When a permission is assigned to a role which previously had no permissions in the simulation-model in question, there may be a need to make this fact known to other places. Likewise, when the last permission is removed from a role.

3.3.2 Permission-Role Enforcement

Permission-role enforcement at a given simulation-model is also entirely a local matter. Once a simulation-model knows a user's role it can easily enforce appropriate permissions for that role since this is solely under control of the simulation-model.

3.3.3 User-Role Assignment

The policy requires user-role assignment to be approved by all relevant simulation-models. Thus this activity requires coordination. It also requires knowledge of which users are in the security administrator role for each simulation-model, as well as knowledge of which simulation-models need to be consulted. We can consider two extreme architectures here.

- No central coordinator. While this architecture is theoretically feasible it would not seem appropriate to have the required complex protocols in this application.

- With central coordinator. The central coordinator could be a special site for this purpose.

In either case we must accommodate unilateral user-role revocation by any simulation-model that has assigned permission to that role.

3.3.4 User-Role Enforcement

For user-role enforcement we have four architectures illustrated in Figures 4, 5, 6 and 7.[6] Each box in these diagrams represents an autonomous entity. In the server-mirror architecture the user-role database is replicated or mirrored at each simulation-model. Each simulation-model is responsible for maintaining and enforcing this information. In the server-pull architecture, each simulation-model consults a central user-role authorization server to look up a given user's roles. In the user-pull architecture each user first obtains digital credentials which securely specify the user's role. These credentials are presented to the simulation-models to obtain access.

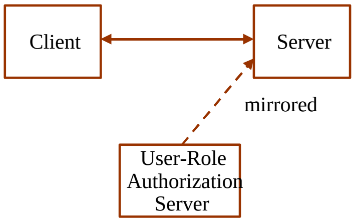

Figure 4: Server-Mirror Architecture for User-Role Enforcement

The proxy-server architecture is a hybrid of server-pull and client-pull. To the client it looks like a server-pull architecture, but the server-pull is actually being accomplished by a proxy-server. To the server it looks like a client-pull architecture, except that the "client-pull" is actually being accomplished by the proxy server. This architecture is very good for legacy systems.

3.4 DRBAC MECHANISMS

Many mechanisms can be used to implement these architectures. For purpose of security we would assume the use of SSL, IPSEC or some other standard network security protocol. Security credentials could be carried in X.509 certificates [PS99] or secure cookies [PSG00]. Alternately, depending on trust relationships these credentials could be carried in cleartext (protected on the network by SSL or IPSEC).

The protocol for voting on assigning a user to a role would need to be developed because this is somewhat unique requirement of this example. In general in choosing mechanisms we would be looking to use standard COTS components to the extent possible and introduce new mechanisms only where they are really

[6] A user-mirror architecture is not considered reasonable, since the user-enforcement is done on the user's machine.

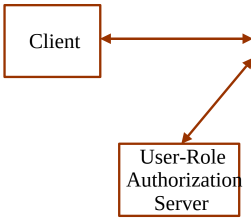

Figure 5: Server-Pull Architecture for User-Role Enforcement

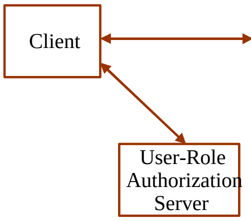

Figure 6: User-Pull Architecture for User-Role Enforcement

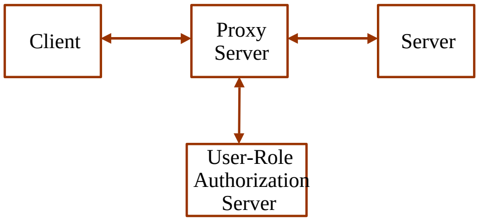

Figure 7: Proxy-Server Architecture for User-Role Enforcement

justified.

4 CONCLUSION

In this paper we have introduced and motivated the OM-AM approach to the engineering of authority and trust in cyberspace. We have illustrated it by means of a distributed role-based access control case study. There are numerous directions in which OM-AM can be developed. OM-AM is also a research agenda as well as a simple and intuitive methodology. We hope to have convinced the reader that it is worth serious consideration.

Acknowledgment

The work of Ravi Sandhu is partially supported by the National Science Foundation.

References

[Dep85] Department of Defense National Computer Security Center. *Department of Defense Trusted Computer Systems Evaluation Criteria*, December 1985. DoD 5200.28-STD.

[Dep91] Department of Defense National Computer Security Center. *Trusted Database Interpretation of the Trusted Computer Systems Evaluation Criteria*, April 1991. NCSC-TG-021.

[FBK99] David F. Ferraiolo, John F. Barkley, and D. Richard Kuhn. A role based access control model and reference implementation within a corporate intranet. *ACM Transactions on Information and System Security*, 2(1), February 1999.

[FCK95] David Ferraiolo, Janet Cugini, and Richard Kuhn. Role-based access control (RBAC): Features and motivations. In *Proceedings of 11th Annual Computer Security Application Conference*, pages 241–48, New Orleans, LA, December 11-15 1995.

[FK92] David Ferraiolo and Richard Kuhn. Role-based access controls. In *Proceedings of 15th NIST-NCSC National Computer Security Conference*, pages 554–563, Baltimore, MD, October 13-16 1992.

[Gui95] Luigi Guiri. A new model for role-based access control. In *Proceedings of 11th Annual Computer Security Application Conference*, pages 249–255, New Orleans, LA, December 11-15 1995.

[LCC^{+}75] R. Levin, E. Cohen, W. Corwin, F. Pollack, and W. Wulf. Policy/mechanism separation in Hydra. In *5th ACM Symposium on Operating Systems Principles*, pages 132–140, 1975.

[McL94] J. McLean. Security models. In John Marciniak, editor, *Encyclopedia of Software Engineering.* Wiley & Sons, Inc., 1994.

[NO95] Matunda Nyanchama and Sylvia Osborn. Access rights administration in role-based security systems. In J. Biskup, M. Morgernstern, and C. Landwehr, editors, *Database Security VIII: Status and Prospects.* North-Holland, 1995.

[NO99] Matunda Nyanchama and Sylvia Osborn. The role graph model and conflict of interest. *ACM Transactions on Information and System Security*, 2(1), February 1999.

[Not94] LouAnna Notargiacomo. Architectures for MLS database management systems. In M. Abrams, S. Jajodia, and H. Podell, editors, *Information Security : An Integrated Collection of Essays.* IEEE Computer Society Press, 1994.

[OSM00] Sylvia Osborn, Ravi Sandhu, and Qamar Munawer. Configuring role-based access control to enforce mandatory and discretionary access control policies. *ACM Transactions on Information and System Security*, 3(2), May 2000.

[PS99] Joon Park and Ravi Sandhu. Smart certificates: Extending x.509 for secure attribute services on the web. In *Proceedings of 22nd NIST-NCSC National Information Systems Security Conference*, Arlington, VA, October 18-21 1999.

[PSG00] Joon Park, Ravi Sandhu, and SreeLatha Ghanta. RBAC on the web by secure cookies. In Atluri and Hale, editors, *Database Security XIII: Status and Prospects.* Kluwer, 2000.

[RS98] Chandramouli Ramaswamy and Ravi Sandhu. Role-based access control features in commercial database management systems. In *Proceedings of 21st NIST-NCSC National Information Systems Security Conference*, pages 503–511, Arlington, VA, October 5-8 1998.

[SA98a] Ravi Sandhu and Gail-Joon Ahn. Decentralized group hieraches in unix: An experiment and lessons learned. In *Proceedings of 21st NIST-NCSC National Information Systems Security Conference*, Arlington, VA, October 5-8 1998.

[SA98b] Ravi Sandhu and Gail-Joon Ahn. Group hierarchies with decentralized user assignment in Windows NT. In *Proc. International Association of Science and Technology for Development (IASTED) Conference on Software Engineering*, Las Vegas, Nevada, October 1998.

[Sal74] J.H. Saltzer. Information protection and the control of sharing in the Multics system. *Communications of the ACM*, 17(7), 1974.

[San93] Ravi Sandhu. Lattice-based access control models. *IEEE Computer*, 26(11):9–19, November 1993.

[San96] Ravi Sandhu. Role hierarchies and constraints for lattice-based access controls. In Elisa Bertino, editor, *Proc. Fourth European Symposium on Research in Computer Security.* Springer-Verlag, Rome, Italy, 1996. Published as *Lecture Notes in Computer Science, Computer Security–ESORICS96.*

[San98a] Ravi Sandhu. Role activation hierarchies. In *Proceedings of 3rd ACM Workshop on Role-Based Access Control*, pages 33–40, Fairfax, VA, October 22-23 1998. ACM.

[San98b] Ravi Sandhu. Role-based access control. In Zelkowitz, editor, *Advances in Computers, Volume: 46.* Academic Press, 1998.

[SB99] Ravi Sandhu and Venkata Bhamidipati. Role-based administration of user-role assignment: The URA97 model and its Oracle implementation. *The Journal Of Computer Security*, 1999. in press.

[SBM99] Ravi Sandhu, Venkata Bhamidipati, and Qamar Munawer. The ARBAC97 model for role-based administration of roles. *ACM Transactions on Information and System Security*, 2(1):105–135, February 1999.

[SCFY96] Ravi Sandhu, Edward J. Coyne, Hal L. Feinstein, and Charles E. Youman. Role-based access control models. *IEEE Computer*, 29(2):38–47, February 1996.

[SM98] Ravi Sandhu and Qamar Munawer. How to do discretionary access control using roles. In *Proceedings of 3rd ACM Workshop on Role-Based Access Control*, pages 47–54, Fairfax, VA, October 22-23 1998. ACM.

[SP98] Ravi Sandhu and Joon Park. Decentralized user-role assignment for web-based intranets. In *Proceedings of 3rd ACM Workshop on Role-Based Access Control*, pages 1–12, Fairfax, VA, October 22-23 1998. ACM.

[TDH92] T.C. Ting, S.A. Demurjian, and M.Y. Hu. Requirements, capabilities, and functionalities of user-role based security for an object-oriented design model. In C.E Landwehr and S. Jajodia, editors, *Database Security V: Status and Prospects*. North-Holland, 1992.

[ZSS99] M. Zurko, R. Simon, and T. Sanfilippo. A user-centered modular authorization service built on an rbac foundation. In *Proceedings of IEEE Symposium on Research in Security and Privacy*, pages 57–71, Oakland, CA, May 1999.