Rebuttal to the NIST RBAC Model Proposal

Trent Jaeger [*] Jonathon E. Tidswell [†]

1 Summary

In this abstract, we rebut the proposed RBAC unified reference model as defined by Sandhu, Ferriaolo, and Kuhn [4]. As a unified reference model, this proposal simply re-enforces some of the concepts that are fundamental to RBAC (i.e., roles, users, and permissions) without clarifying the more complex concepts. Also, the definitions of the concepts are too informal to drive any useful standards proposal. We suggest formalizing the base concepts, including the addition of role administration, and that more work is necessary for constraints to be useful.

2 Claims

The two main claims in the proposal are that: (1) the proposed RBAC model should be considered as a *unified reference model* and (2) the proposed RBAC model is a reasonable *foundation for future standards.*

First, the basic concepts of users, permissions, roles, hierarchies, and constraints have been part of the de facto RBAC model since the First ACM RBAC Workshop. Since that time researchers have tried to address the means of expressing these concepts effectively and develop the properties that can be enforced by these concepts. Unfortunately, little of this work or even these goals has been deemed mature enough to be included in the proposal.

Second, little of what is included seems useful as a foundation for standards. The concepts themselves are either so vaguely-specified as to be open to a large number of interpretations (many of the them wrong or useless as has been the problem with hierarchy specification [2]) or are specialized in unnecessary ways. A foundation for a standard must define primitives and operations such that a general API can be developed within which a variety of useful legal solutions can be generated.

3 Modeling Limitations

The main limitations in the unified reference model for RBAC in its choices to limit, ostensibly to simplify, which unnecessarily constrain the possible modeling choices.

First, the RBAC unified reference model consists of four levels each building upon the previous one. Other than the first level, these so-called levels are orthogonal extensions to the basic RBAC model (i.e., the first level). For example, it is not necessary for there to be any role hierarchy for the administrator to use constraints. Therefore, unlike the alternative proposed in Appendix A, we claim that the proper formulation is to have a base model (first level only) and RBAC concepts (levels 2 and 3). The fourth level should not be part of a unified reference *model.* It is an implementation detail that should be part of an interface in level one. If the situation prevents effective computation of the permission-role review, then the implementation should report this limitation.

Second, there has been much discussion about the formalization of permissions (e.g., aggregating objects to which permissions may be assigned), users (e.g., ag-

[*] IBM T.J. Watson Research Center, Hawthorne, NY 10532, jaegert@us.ibm.com

[†] School of Computer Science and Engineering, University of New South Wales, Australia

RBAC 2000, Berlin, Germany
© ACM 2000 1-58113-259-x/00/07 ...$5.00

gregating users into groups which may be assigned to roles), hierarchies (e.g., identifying role semantics), and constraints (e.g., defining constraint models that provide safety with minimal complexity and reasonable flexibility). The proposed model does not add any value to user or permission aggregation, and it only limits the expression of hierarchies and constraints. While role hierarchies are rightly identified as a partial order relation with different mathematical variants (e.g., DAG or tree), little value is added by the description of roles as organizational roles or job functions. This can lead to the type of role hierarchy design flaws identified by Moffett [2]. Limiting RBAC to the expression of separation of duty constraints is reasonable, but the big problem in constraints (and in hierarchies) determining how to express these constraints. With no guiding concepts the model is vacuous.

Third, some surprising modeling concepts are missing from the unified reference model. Most surprising is the omission of the administrative role hierarchy. Role administration can provide a simple approach to defining a safe access control model (as in the context of HRU [1]). Although not as flexible as general constraints, role administration is at least well defined and practical for use by administrators. Constraint languages proposed thusfar fail to provide the simplicity necessary to make them practically effective, so the inclusion of role administration provides a useful alternative concept to constraints which is fairly well-understood (even if it is not included in many systems).

4 Requirements for Standards

A standard API for a service (e.g., CORBA or POSIX) defines a set of data types and operations on those data types. These data types and operations define the mechanisms of a standard system. They must be agreed upon, so that systems of one vendor can communicate with systems of another vendor and applications can be built that can use the functionality regardless of implementation changes. However, the policy of the system and the mechanisms for implementing the individual operations must remain open, so innovation behind the standard interface is possible.

The key to defining a standard API is defining the data types and operations fundamental to the system without limiting the flexibility of the implementations. Unfortunately, the model proposed here is unclear both about the data types and the operations. Prior models by the authors (e.g., [3]) provide a richer definition of types and operations for the first level concepts. Here the key operations are the assignment of users and permissions to roles.

Work is still ongoing for role hierarchy operations and constraint operations. As noted above, Moffett identified several problems with particular hierarchy semantics [2], so multiple hierarchy object may be necessary. It does seem possible that a very high level interface for taking a constraint data type and evaluating it can be devised, but it is unclear if the interface will be useful. The complexity of constraint languages may require more specialized data types to become the basis of standards.

References

[1] Michael A Harrison, Walter L Ruzzo, and Jeffrey D Ullman. Protection in operating systems. *Communications of the ACM*, 19(8), August 1976.

[2] Jonathan D. Moffet. Control Principles and Role Hierarchies. In *Proceedings of* 3^{rd} *ACM Workshop on Role-Based Access Control*, November 1998.

[3] R. S. Sandhu, V. Bhamidipati, and Q. Munawer. The ARBAC97 model for role-based administration of roles. *ACM Transactions on Information System Security*, 1(2), February 1999.

[4] Ravi S. Sandhu, David Ferraiolo, and Richard Kuhn. The NIST Model for Role-based Access Control: Towards a Unified Standard. In *Proceedings of* 5^{th} *ACM Workshop on Role-Based Access Control*, July 2000.