Editors: Rick Kuhn, kuhn@nist.gov
Susan Landau, susan.landau@sun.com
Ramaswamy Chandramouli, mouli@nist.gov

RBAC Standard Rationale

Comments on "A Critique of the ANSI Standard on Role-Based Access Control"

David Ferraiolo and Rick Kuhn
US National Institute of Standards and Technology

Ravi Sandhu
University of Texas at San Antonio

As the authors of the original proposal for the role-based access control (RBAC) standard and developers of the models[1,2] from which it derives, we welcome Ninghui Li, Ji-Won Byun, and Elisa Bertino's gracious request to respond to their critique, which appears on p. 41 of this issue. This is an opportune time in the revision cycle to introduce proposals for changes to the standard, and we're grateful to them for pointing out some technical errors and raising additional issues that might be relevant to future versions. In fact, we considered most of their proposed changes when we initially drafted[3] and revised the standard prior to submitting it to the International Committee for Information Technology Standards (INCITS), and more than 18 months elapsed between the initial submission and final voting, during which time the standard received extensive attention from interested parties. With our response here, we hope to clarify the rationale for the choices and trade-offs we made between the time when we first proposed the standard in 2000 and when industry discussion led to its approval in 2004.

Interested parties can correct technical errors in the current standard by writing an amendment that they can then propose to INCITS for balloting. This relatively straightforward process typically takes a few months; revision suggestions require community discussion and time for consensus to develop. To advance this process, we offer comments here on each of Li, Byun, and Bertino's suggestions.

Suggestion 1: Sessions

> "The notion of sessions should be removed from Core RBAC and introduced in a separate component."

The *session* concept is a critical part of RBAC that distinguishes it from traditional group mechanisms. Sessions allow the activation of a subset of roles assigned to a user; without sessions, all user roles are always activated, which can potentially violate least privilege. Systems that allow the activation of a subset of roles necessarily support a session concept, as do systems that support dynamic separation of duties. Nevertheless, systems that insist on activating all roles all the time might be useful in practice and could be recognized in a revised standard in the future if the community exhibits sufficient interest.

Enterprise security management (ESM) systems alone don't include the notion of a session, as required by the RBAC standard, but combining their administrative features with those of the target systems does meet RBAC requirements. The NIST-developed Role Control Center (RCC), for example, is an ESM RBAC standard reference implementation (meeting core, general hierarchy, and static separation of duty (SSD) requirements, with advanced permission review) that doesn't directly support the concept of a session or role activation. RCC requires the existence of minimum target system features in its emulation of RBAC to include system support for groups, user accounts, and sessions. Through RCC, target system user accounts and groups are centrally created and deleted, and membership is created and deleted to correspond to RCC user roles and role-to-role relations. When a user logs in to the target system, he or she creates a local session with a security context that includes those groups (or the emulated roles) for which RCC granted the user membership. In terms of the RBAC standard, we say those roles that correspond to the groups included in the security context are activated in the session.

Suggestion 2: Single-role activation

> "The standard should accommodate RBAC systems that allow only one role to be activated in a session."

After considerable debate, we consciously chose not to include single-role systems in the RBAC standard. We determined that Core RBAC should include those systems that possess a robust group or ACL mechanism, but some ACL mechanisms allow a user to be a member of only one group at a time, and other systems restrict an ACL to include only one group. Fundamentally, if a group is to correspond to a role, it must facilitate the notion of a many-to-many relation among users and permissions. Systems that restrict a user's membership to a single role also activate one role at a time.

Although we could argue that single-role activation seems to support least privilege, we could also argue that it doesn't. Restricting a user to one role administratively and operationally results in roles with many permissions. Essentially, system administrators will create roles that attempt to correspond to entire job functions or users will need to log on and off many times during the course of a work session. However, if a user can be a member of multiple roles, administrators may create roles at the task level, allowing users to surgically activate them to support their current activities. Single-role activation could also lead to users having multiple accounts, each mapping to a different role. Clearly, this approach has its own problems with respect to separation of duty and accountability requirements.

Suggestion 3: Base and derived relations

> "The standard should make a clear distinction between base relations and derived relations."

In developing the draft standard, we felt that "derived relations" would make the model easier to read and understand, but no implementation requirement demands that these be maintained explicitly as a data structure. Developers are free to select data structures that work best for their products.

Suggestion 4: Role-dominance relationships

> "The reference model should maintain a relation that contains the role-dominance relationships that have been explicitly added, and update this relation when the role hierarchy changes."

Much of the discussion about role-dominance relationships centers on the standard's treatment of the role hierarchy as a partial order and some of the trade-offs that result. Li, Byun, and Bertino suggest that this structure is a carryover from the early mandatory access control (MAC) model, but we didn't adopt the partial order structure without consideration. David Ferraiolo and Rick Kuhn[1] incorporated a role hierarchy represented as a tree structure rather than a partial order, and later papers—particularly by Ravi Sandhu, Ed Coyne, Hal Feinstein, and Charles Youman[2]—described a partial-order structure to give the RBAC model greater flexibility. Other authors' widespread acceptance suggests that the added flexibility is useful.

The standard treats the role hierarchy mathematically as a partial order and doesn't address its internal representation. Historically, some system designers have addressed hierarchy modifications as changes to an underlying binary relation whose transitive, reflexive closure defines the hierarchy. Many different underlying binary relations can have the same closure, thereby defining the same partial order. In the standard, the hierarchy is maintained per se, not as the closure of some underlying relation. A dynamic view of the hierarchy is no doubt important, but it isn't clear that viewing the changes as applied to an underlying binary relation is the best approach. The example of a temporary relationship that Li, Byun, and Bertino give can be handled by other administrative methods such as delegation, so it isn't clear that temporary modifications to a role hierarchy are appropriate for such purposes. The authors' point that role hierarchies deserve further investigation is well taken, but as they note, no real consensus exists in the community.

Note, too, that we could define additional operations in specific implementations to add and delete edges from a hierarchy. The standard allows for deleting an edge, but states that implied edges will be retained; an operation that deletes an edge and all implied edges can also be defined. Vendors often use such enhancements to distinguish their products in the market. When consensus emerges on a desirable set of extended operations, we'll incorporate them into a revised standard.

Suggestion 5: Role inheritance

> "The semantics of role inheritance should be clearly specified and discussed."

Li, Byun, and Bertino's analysis of different interpretations of hierarchies is incomplete. More significantly, their proposal to interpret hierarchies differently in different circumstances isn't conducive to conceptual simplicity in the model and is likely to lead to considerable confusion among practitioners. A family of models should build on its components coherently and consistently without altering the meaning of basic concepts.

To address some of the specifics of their proposal, we note that the authors treat user inheritance and permission inheritance as distinct concepts rather than integrating them, as the standard has done. Given that these notions can be decoupled in different ways, the authors' approach is incomplete (because it considers only one approach), assuming we even accept that these notions should be decoupled. Furthermore, their statement about AI, "under this interpretation alone, *u* cannot activate *r*2 directly;" is incorrect because the standard allows *r*2 to be activated directly without activating *r*1. Consequently, the rest of the discussion is moot. In our experience, the major issues raised by the standard's users in practice have to do with the construction of roles and role hierarchies. Researchers and practitioners broadly agree that this task is difficult and costly, and the research community might have some insights for facilitating it.

Some notion of roles for access control predates the research papers cited by the authors by at least a decade—for example, some banking systems in the 1970s used roles even if they weren't called by that name. Our work[1,2] was designed to formalize RBAC and add features (such as hierarchies and constraints) to make it more useful to software developers and administrators. Extensive discussion of these and subsequent papers over many years led to the consensus standard for RBAC.

Readers interested in these or other recommendations for changes to the RBAC standard are encouraged to work with the INCITS CS1 working group (http://cs1.incits.org). For more information on RBAC and the RBAC standard, please see http://csrc.nist.gov/rbac. □

References

1. D.F. Ferraiolo and D.R. Kuhn, "Role-Based Access Control," *Proc. 15th Nat'l Computer Security Conf.*, US Nat'l Security Agency/Nat'l Inst. of Standards and Technology, 1992, pp. 554–563; http://csrc.nist.gov/rbac/ferraiolo-kuhn-92.pdf.
2. R. Sandhu et al., "Role-Based Access Control Models," *Computer*, vol. 29, no. 2, 1996, pp. 38–47.
3. R. Sandhu, D. Ferraiolo, and R. Kuhn, "The NIST Model for Role-Based Access Control: Towards a Unified Standard," *Proc. 5th ACM Workshop on Role-Based Access Control*, ACM Press, 2000, pp. 47–63.

***David Ferraiolo** is supervisory computer scientist at the US National Institute of Standards and Technology. His research interests include access control and authorization management. Ferraiolo has a BS in computer science and mathematics from the State University of New York, Albany. He is coauthor of* Role Based Access Control *(Artech House, 2003 and 2007). Contact him at david.ferraiolo@nist.gov.*

***Rick Kuhn** is a computer scientist at the US National Institute of Standards and Technology. His research interests include information security, software assurance, and empirical studies of software failure. Kuhn has an MS in computer science from the University of Maryland, College Park. He is a senior member of the IEEE and coauthor of* Role Based Access Control *(Artech House, 2003 and 2007). Contact him at kuhn@nist.gov.*

***Ravi Sandhu** is executive director of the Institute for Cyber Security at the University of Texas at San Antonio, where he holds the Lutcher Brown Endowed Chair in cybersecurity. His research interests include security models, architectures, and mechanisms. Sandhu has a PhD in computer science from Rutgers University. He is a fellow of the ACM and the IEEE. Contact him at ravi.sandhu@utsa.edu.*