SANS ANALYST PROGRAM

Sponsored by FoxT

Extending Role Based Access Control

A SANS Whitepaper

Written by J. Michael Butler, Information Security Consultant – April 2011

Purpose and Benefits of RBAC

The Pain of Separate OS RBAC Administration

Heterogeneous Extended RBAC

Planning for XRBAC

Methods for Implementing XRBAC

Advisor:
Dave Shackleford, Voodoo Security

Introduction

Role based access control (RBAC), first conceived in the 1990s, is a mature and widely used model for controlling access to operating systems and software. Within the RBAC model, access is granted based on the roles individual users have in the organization using the system. For example, with RBAC, a user administrator can add, change or delete users without having access to more powerful commands a system administrator can execute and without having access to files a system administrator can access. RBAC solves the problem that many UNIX systems have where "root" is used to gain complete access in order to do the simplest administrative tasks, which do not require super user access.

While RBAC offers many benefits for implementing fine-grained access controls, one drawback is that it has been developed uniquely for each operating system. This makes centralized user control difficult without a customized front end and, possibly, customization for each system being accessed. Customization tools normalize the functions of RBAC so they look the same across the variety of systems that live in most data centers today. This enables centralized management of users and reduces errors in user administration that can occur in environments in which multiple systems are accessed. By extending RBAC, either through customization or the use of a third party tool, organizations can reduce requirements for mundane tasks that may currently be assigned to their technical, higher salaried operating system administrators, thus enabling them to concentrate on more important assignments. Ultimately, studies show a surprising long-term return on an RBAC extension investment in reduced employee downtime, as well as an obvious simplification of user provisioning tasks.

This paper discusses advantages and disadvantages of RBAC, along with options to consider when planning to extend RBAC to allow for centralization and standardization in a heterogeneous environment of multiple, diverse operating systems.

Purpose and Benefits of RBAC

The National Institute of Standards and Technology (NIST) defines RBAC as "Access control based on user roles (i.e., a collection of access authorizations a user receives based on an explicit or implicit assumption of a given role). Role permissions may be inherited through a role hierarchy and typically reflect the permissions needed to perform defined functions within an organization. A given role may apply to a single individual or to several individuals."[1]

The RBAC model was formally introduced in 1992 by Ferraiolo and Kuhn at the 15th National Computer Security Conference, Oct 13–16.[2] In their model, all users have a direct role and may have an inherited role. Those roles must be authorized, and transactions must be authorized for a role as well. The RBAC model was extended and updated in 1995 and 1996,[3] and then in 2000, the RBAC standard was proposed by Sandhu, Ferraiolo and Kuhn.[4] The standard was finally published for the first time in 2004 by the InterNational Committee for Information Technology Standards as INCITS 359:2004. The current standard is INCITS 359:2004[R2009] Information Technology – Role Based Access Control (RBAC) [CS1.1][5]

Currently there is a revision under way with another meeting scheduled in May 2011.[6] Kuhn, Coyne, and Weil have written about this upcoming revision and commented upon the advantages of extending RBAC for the IEEE.[7]

RBAC provides many advantages for organizations with requirements for more granular, role-based controls over who can access what data based on roles within their organizations. They can be used to protect data from the wrong employees, as well as to aid in compliance. There are also disaster recovery aspects to consider with RBAC.

[1] http://csrc.nist.gov/publications/nistpubs/800-53-Rev3/sp800-53-rev3-final_updated-errata_05-01-2010.pdf, page B-11 (Aug. 2009)
[2] http://csrc.nist.gov/groups/SNS/rbac/documents/ferraiolo-kuhn-92.pdf
[3] http://csrc.nist.gov/groups/SNS/rbac/documents/sandhu96.pdf
[4] http://csrc.nist.gov/rbac/sandhu-ferraiolo-kuhn-00.pdf
[5] www.incits.org/
[6] http://csrc.nist.gov/groups/SNS/rbac/rbac-standard-revision.html.
[7] http://csrc.nist.gov/groups/SNS/rbac/documents/kuhn-coyne-weil-10.pdf

Enabling Separation of Duties

One of the important advantages of using RBAC is that it provides enough granular control that the organization is able to accomplish adequate separation of duties (SoD) without a kernel intrusion into the operating system. A crucial aspect of security is to separate duties in such a way that employees cannot take advantage of their access in order to hide malicious activity. For example, if an accounts payable (AP) entry clerk can also write the checks, it is a simple matter for the clerk to key in fake invoices and write checks to himself. In this case, SoD indicates one user should enter AP, while a second person can write the checks to pay the bills. RBAC can be leveraged to more easily achieve business-requirements for appropriate SoD. NIST explains SoD the following way:

> *RBAC is . . . well suited to separation-of-duty requirements, where no single individual has all permis¬sions needed for critical operations such as expenditure of funds. Proper operation of RBAC requires that roles fall under a single administrative domain or have a consistent definition across multiple domains, so distributed applications might be challenging.*[8]

Supporting Compliance Regulations

The payment card industry (PCI) now specifies in the PCI Data Security Standards (DSS) version 2.0 that RBAC must be used to control access to payment card data:

> ***7.1.2*** *Confirm that privileges are assigned to individuals based on job classification and function (also called 'role-based access control' or RBAC).*[9]

In addition, NIST SP800-66 specifies RBAC as an option for protecting electronic protected health information (EPHI). EPHI represents another instance of legally protected data that will be well served by controlling access by using RBAC. As NIST puts it, system owners should: "Select an access control method (e.g., identity-based, role-based, or other reasonable and appropriate means of access.)"[10] Healthcare industry regulations, in fact, require the use of RBAC as well.[11]

In short, regardless of the legal or contractual reason for protecting data — whether it's HIPAA, the Gramm-Leach-Bliley Act, PCI DSS, or internal policy — RBAC offers the best standard available today for controlling access for those with a need to know. The only thing lacking is consistency among systems.

Disaster Recovery Requirements

Remember that the RBAC system must be set up in such a way that it translates over to the Disaster Recovery (DR) environment as well. The last thing an organization wants is to be forced into a disaster situation that has an additional risk element because its access controls do not function correctly in the DR environment. Business continuity planning must consider how the RBAC system is replicated in the DR environment. That system must be fully tested during DR exercises to ensure that it protects data as well as it does in normal production.

[8] http://csrc.nist.gov/groups/SNS/rbac/documents/kuhn-coyne-weil-10.pdf, page 79 (June 2010)
[9] www.pcisecuritystandards.org/documents/pci_dss_v2.pdf, page 44
[10] http://csrc.nist.gov/publications/nistpubs/800-66-Rev1/SP-800-66-Revision1.pdf, page 23 (Oct. 2008)
[11] http://csrc.nist.gov/groups/SNS/rbac/standards.html

The Pain of Separate OS RBAC Administration

The implementation of RBAC is different in every operating system (OS). There is currently no standardization for implementation. This makes single console management and compliance difficult in most enterprises that administer heterogeneous systems. The following sections explain these differences in more detail.

Windows/Active Directory RBAC

In Microsoft Active Directory (AD), RBAC is implemented using *Groups* to allow users within those groups to perform specified functions based on those group roles. In a typical Microsoft environment, users are assigned to one or more groups to control their access to systems and functionality based on their job functions. Table 1 gives shortened descriptions for Active Directory default user roles (Groups), including Account Operators, Administrators, Backup Operators, Network Configuration Operators, Performance Log Users, Print Operators, Server Operators, and Users, as defined by Microsoft in its documentation.

Group	Description	Default User Rights
Account Operators	Members of this group can create, modify, and delete accounts for users, groups, and computers located in the Users or Computers containers and organizational units in the domain, except the Domain Controllers OU. Members of this group can log on locally to domain controllers in the domain and shut them down.	Allow log on locally; Shut down the system.
Administrators	Members of this group have full control of all domain controllers in the domain.	Domain administrators can perform any function in any system, much like UNIX root.
Backup Operators	Members of this group can back up and restore all files on domain controllers in the domain, regardless of their own individual permissions on those files. Backup Operators can also log on to domain controllers and shut them down.	Back up files and directories; Allow log on locally; Restore files and directories; Shut down the system.
Network Configuration Operators	Members of this group can make changes to TCP/IP settings and renew and release TCP/IP addresses on domain controllers in the domain.	No default user rights.
Performance Log Users	Members of this group can manage performance counters, logs and alerts on domain controllers in the domain, locally and from remote clients.	No default user rights.
Print Operators	Members of this group can manage, create, share, and delete printers connected to domain controllers in the domain. They can also manage Active Directory printer objects in the domain. Members of this group can log on locally to domain controllers in the domain and shut them down.	Allow log on locally; Shut down the system.
Server Operators	On domain controllers, members of this group can log on interactively, create and delete shared resources, start and stop some services, back up and restore files, format the hard disk, and shut down the computer.	Back up files and directories; Change the system time; Force shutdown from a remote system; Allow log on locally; Restore files and directories; Shut down the system.
Users	Members of this group can perform most common tasks, such as running applications, using local and network printers, and locking the server.	No default user rights.

Table 1. Sampling of Microsoft AD User Groups and Resource Affiliations[12]

The Active Directory system is robust and contains many roles by default. However, without additional extensions and custom or third party applications, RBAC controls for AD-only systems that "speak" AD. LDAP functionality can be leveraged from an Active Directory, but the limitations of the AD LDAP implementation may still require tweaking to make it function with other applications or operating systems. For example, see the "Symantec Best Practices when using Microsoft AD as an LDAP Source" document in the footnote below.[13] Such customization will be resource-intensive when attempting to use AD as the central control for multiple systems and applications.

In an effort to simplify RBAC using AD, Microsoft has provided a tool called AzMan that can help administrators accomplish RBAC using AD more easily. Microsoft defines AzMan as "a role-based access control (RBAC) framework that provides an administrative tool to manage authorization policy and a runtime that allows applications to perform access checks against that policy."[14]

Microsoft claims that AzMan can be easily integrated with applications and provides a way to manage and enforce RBAC. This solution, too, will require resources to set up, configure, code for, and implement in any environment. So, despite these efforts by Microsoft, integrating AD and LDAP across multiple operating systems is a significant challenge.

Security Enhanced Red Hat Linux (SELinux) has features that enable RBAC implementation. The user's role is an attribute of RBAC. In SELinux, users are authorized for roles, and roles are authorized for domains.[15] For SELinux users, a role determines the relationship between domains and users. Access to a domain is controlled by the user's role. Ultimately, the object types a user can access are based on the user's role.

The SELinux Guidelines and Recommendations list the following default roles within Red Hat:

system_r — This role is for all system processes except user processes.

user_r — This is the default user role for Linux users.

object_r — In SELinux, roles are not utilized for objects when RBAC is being used. Roles are strictly for **subjects**. This is because roles are task-oriented, and they group together doers, which are **subjects**. For this reason, all objects universally have the role **object_r**, and the role is only used as a placeholder in the label.

sysadm_r — This is the system administrator role in a strict policy. In such a policy, switching to the root user with **su** gives you the role of **sysadm_r**.[16]

Other roles can be created and implemented, as well. While this is valuable for organizations running only Red Hat, it does not easily translate into RBAC implementation across multiple operating systems without custom programming.

[13] http://www.symantec.com/connect/articles/best-practices-when-using-microsoft-active-directory-ldap-source
[14] http://msdn.microsoft.com/en-us/library/bb897401.aspx
[15] http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/6/html/Security-Enhanced_Linux/chap-Security-Enhanced_Linux-SELinux_Contexts.html
[16] http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/4/html/SELinux_Guide/rhlcommon-section-0038.html

According to Hewlett-Packard (HP), RBAC is fully implemented within the HP-UX system, although it requires that the organization carefully plan their RBAC deployment because the roles are not automatically defined.

HP-UX RBAC negates the need for all-powerful root user access to conduct all operations and prevents access for non-root users for operations that should not require root. Using HP-UX RBAC, one can distribute administrative responsibilities with appropriate authorizations within the roles because they're created and assigned to non-root users and groups.[17]

The HP-UX documentation strongly suggests administrators follow specific planning steps before deploying HP-UX RBAC.

- Plan roles for users.
- Plan authorizations for the roles.
- Plan the authorization-to-command mappings.

Once planned, roles and authorizations can then be configured. All of this must be done before the RBAC system can be used. Again, this is targeted for HP-UX systems and cannot be easily leveraged across heterogeneous OS environments.

AIX has documented three pre-defined roles:

- **ISSO** – Information System Security Officer
- **SO** – System Operator
- **SA** – Security Administrator[18]

As with the other systems, other roles can be planned, created, authorized, and implemented using the system tools. AIX documentation lists the following hierarchy for user and role definitions:

- **Authorizations** are assigned to commands.
- **Roles** are assigned to users.
- **Privileges** are associated with processes.
- **Explicit privileges** are assigned to commands.

The system has pre-defined authorizations to certain commands and roles. Administrators can provide an authorization to an executable program and assign the authorization to a role. Each user is assigned a role. Therefore, a user with a defined role should be able to execute authorized commands associated with their role.

[17] https://h20392.www2.hp.com/portal/swdepot/displayProductInfo.do?productNumber=AccessControl
[18] www.ibm.com/developerworks/aix/library/au-aix_rbac/index.html

User	Roles	Responsibilities
ISSO	ISSO	Establishing and maintaining security policy Setting passwords for user Network configuration Device configuration
SO	SO	System shutdown reboot File system backup, restore, and quotas System error logging, trace, and statistics Workload administration
SA	SA	User administration excluding password Filesystem administration Software Installation and Update Network Daemon management and device allocation

Table 2: Some of the Roles Defined in AIX Documentation

Once again, it is the implementation team's responsibility to set up specific roles that apply only to the AIX systems and applications, but these do not translate well to other systems.

Solaris has no pre-defined roles, so the organization must decide what types of roles should be set up. Instead, Sun documentation shows how roles can be configured using pre-defined rights profiles for the following administrator levels:

- **Primary Administrator** – Can perform all administrative tasks, grant rights to others, and edit rights associated with administrative roles. May also assign to others the Primary Administrator role.
- **System Administrator** – Can perform most non-security administrative tasks. For example, add new user accounts without setting passwords or granting rights to users.
- **Operator** – Can perform simple administrative tasks, such as backup, restore, and printer maintenance.[19]

[19] www.sun.com/software/whitepapers/wp-rbac/wp-rbac.pdf, page 6

Microsoft SQL Server

Microsoft SQL Server uses Lync Server to implement separation of duties for server administrators from SQL administrators with RBAC. This represents yet another application to implement RBAC for a unique environment that will not carry over to other systems. Microsoft claims Lync Server uses RBAC to mitigate delays in SQL implementation. This is done using the Lync Server shell, which provides a way for the SQL Server Administrator to use PowerShell to configure databases with correct data and log files. [20]

Oracle

The Delegated Administration Services component of the Oracle Identity Management infrastructure defines a delegation model based on RBAC. Application roles are implemented using Oracle Internet Directory for managing groups and roles. APIs and other interfaces are leveraged by Application Server components for managing objects.[21]

[20] http://technet.microsoft.com/en-us/library/gg398832.aspx
[21] http://download.oracle.com/docs/cd/B14099_19/core.1012/b13999/deploy.htm

Heterogeneous Extended RBAC

All the systems listed previously may help with security and compliance in single system organizations. Most organizations, however, have implemented multiple operating systems. Extending RBAC to accomplish improved security and compliance in such a heterogeneous environment is a challenging custom development task.

What happens when the organization assigns a role, such as user administrator (UA) to a single person who is then expected to add/change/delete users in every OS in the enterprise? Typically, this results in one UA having to learn every system's unique methods for user administration. It also means that unless the organization implements RBAC and maintains permissions separately in each system, UAs will end up having far more access than their job functions require — which is in direct violation of multiple compliance mandates. The UA must be added to each system separately and have access to all systems individually in order just to add users at the OS level. In such situations, there is no SoD. The user can also wipe out or modify crucial files, write/implement code, or any one of a thousand other things that could render the system, in a worst-case scenario, inoperable.

The answer to this problem is extending RBAC and normalizing the roles, authorizations and privileges between systems.

RBAC and XRBAC ROI

In this paper, extended RBAC (ERBAC or XRBAC) is defined as access control based on user roles regardless of operating system or application. Extending RBAC can allow administrators to control access across multiple operating systems from a central location. This requires creating and/or identifying identical roles in the different systems, and then providing a way for appropriate administrators to tie those roles to the users assigned the job function that goes along with the role. In this manner, the roles are coordinated to accomplish the same purpose, regardless of the system upon which that role will be executed.

Since a "pure" RBAC system designed for a single OS still has shortcomings, organizations need to build on the NIST definition of RBAC quoted earlier in order to overcome these issues.

Consider the following advantages of extending RBAC:

- Less skilled and lower-cost personnel can be utilized for simple administrative tasks such as adding, changing or deleting users. This is made possible by eliminating root and other powerful IDs and delegating the specific tasks that need to be performed.
- System administrators can be freed up to handle crucial maintenance tasks such as patches and updates, system enhancements, new system implementation, and so on.
- Dangerous shared roles/accounts (such as root, administrator, and others) and their passwords can be protected, disabled or removed as appropriate to each system.
- Consistency can be created and maintained in roles across multiple systems in a way that is understood by business personnel who request those role assignments.
- Roles can be remapped painlessly once the system is fully operational, thus allowing the systems to keep up with business needs.
- Access to sensitive data can be removed from system administrators while still allowing them to do their jobs.
- Granular, segregated access and permissions can be achieved, down to OS resource and object level, without kernel intrusion.

The problem is applying the same controls across different OSs. In a smaller enterprise, there are likely to be fewer diverse OSs, while larger organizations typically have multiple systems that have specific access control issues associated with them and their hosted applications.

SANS and other expert organizations say proper planning is crucial for extending role based access in a scalable manner.[22] This is further complicated as some users may require assignment to multiple roles, such as a senior HR administrator with access to the HR benefits, intranet portal, and the payroll systems. RBAC will be most effective when the roles are carefully designed to match the hierarchy of the organization for which the system will be used. While it is important to carefully weigh the cost and effort to plan and create the roles for an organization, it is also important to know the return on investment (ROI) that can be realized by the use of RBAC.[23] If implemented correctly, the cost reductions will be immediate in the form of reduced resource requirements, such as time to add, delete or modify user access. There will be a measurable reduction in errors, which translates directly into less downtime for system users and a positive impact on productivity.

[20] www.sans.edu/student-files/projects/200709_001.ppt, slide 11
[21] http://csrc.nist.gov/publications/nistpubs/800-36/NIST-SP800-36.pdf, page 18 (Oct 2003)

In February, 2011, NIST published a compelling document pertaining to ROI on RBAC. Even after deducting the cost of implementation of RBAC, NIST found that benefits of RBAC were significant across a wide spectrum of organizations and that similar results were true of the economy in general, to the tune of billions of dollars.

> *The study quantified economic benefits and costs, estimated the adoption of RBAC over time, and reviewed broader issues in identity and access management (IAM) for which using roles is advantageous. It offers an analysis of the economics of the myriad technological, business, and regulatory drivers underlying organizations' selection of which approach to access control is appropriate, given their organization's user base, staff turnover, workflow patterns, and regulatory considerations.*[24]

Benefits of RBAC manifest in two ways, according to the study: More efficient provisioning and more employee uptime, where employees can be doing their jobs instead of waiting for their resources initially and if access problems occur. The report calls out a 10,000-employee financial services organization with a nearly $300,000 cost savings in terms of downtime. Overall, by simplifying the process of extending RBAC, hypothetically the firm would save $1 million overall, according to the NIST study. Regulatory compliance and governance policies pointed out even greater benefits, such as enhanced insight into an organization's access control policy and more efficient maintenance of that policy.

While the NIST study does not necessarily address the difference between single system RBAC and XRBAC, the logical conclusion is that good ROI on RBAC, in general, will apply in the same ways and to an even greater degree to extended RBAC, except with a multiplier appropriate to the number of different systems brought under centralized control.

[24] http://csrc.nist.gov/groups/SNS/rbac/documents/20101219_RBAC2_Final_Report.pdf

Planning for XRBAC

The same implementation elements are required for XRBAC as for RBAC. Here is a suggested listing of actions that will lead to design and implementation of XRBAC in any environment.

1. To begin planning, carefully determine the following:
    a. The appropriate internal personnel to serve on an XRBAC Design Team
        i. IT Personnel — system administrators
        ii. Business Personnel — those who know the business roles
        iii. Management Personnel — those who will approve users for roles
    b. User roles based on job functions — separation of duties (SoD) requirements
    c. Appropriate role hierarchy — roles that can be inherited by other higher level roles
    d. Static attributes that may be applicable
        i. Physical location
        ii. Job function
        iii. Job position
        iv. Special clearance required to access certain data
    e. Dynamic attributes that may be applicable
        i. Time of day of access
        ii. Temporary work assignments
        iii. Circumstantial special clearance to access certain data
        iv. Other attributes appropriate to the environment
    f. Objects that need to be controlled, including mapping to access control lists (ACL)
    g. Commands required to perform the role upon the object
    h. Access and authorizations required to objects for each role
    i. How to map users to roles to ACLS/Objects
    j. Audit requirements
        i. System access
        ii. User add/change/delete
        iii. Object access
            1. File access/add/change/delete
            2. Records access/add/change/delete
            3. Printers

2. Set requirements for the XRBAC system.
    a. Determine the preferred method(s) within the organization for controlling access across the environment
        i. Standards that must be followed
        ii. Third party software
        iii. Internally developed code
    b. Create a detailed request for information/proposal (RFI/RFP) document(s) to provide to prospective vendors and contractors
3. Establish a team for vendor selection and proof of concept (PoC).
    a. IT team
    b. Business team
    c. Management for review

Methods for Implementing XRBAC

Two methods for implementing XRBAC are custom development or the acquisition of available third party tools. Custom development is the more expensive and resource-intensive choice. It will also take a longer time to complete implementation. The time and resources required varies with the number of different systems, the detail of the planning effort, the expertise of the resources involved, the number of roles, and other related factors. It is not within the scope of this paper to explain how to design and implement such a custom system. We can simply say that it can be done — given time, a good project management team, talented coders, and a commitment to maintaining the code into the future.

A number of tools currently exist in the category of enterprise access management (EAM) systems. It does not necessarily follow that an EAM tool is also an RBAC tool. When researching tools, an organization should find one that integrates cleanly with as many of the operating systems in use in their environment as possible. The solution should take advantage of the existing OS RBAC capabilities and provide methods for harmonizing role management across diverse operating systems according to the RBAC standard. Planning should begin with the determination of the roles, as suggested above, which should take place before a vendor search. To find the best vendor fit, it is necessary to develop a request for information (RFI) and/or a request for proposal (RFP) that clearly outlines the organization's expectations from an XRBAC system vendor.

Conclusion

As far back as 2003, NIST authors wrote that RBAC had emerged as a promising feature of many database, security, and operating system products. More recent NIST studies prove there are real cost savings associated with implementing role based access controls. The essential advantage of RBAC is the ability for system administrators to assign roles to individual users based on their job functions and other attributes including capabilities, work requirements, and responsibilities in the organization. With RBAC, some users may have multiple roles to meet their job requirements. RBAC also helps administrators better control users without having to explicitly assign users to specific resources.[25]

Unfortunately, so far RBAC has developed in silos, making it difficult to manage access for each operating system and version used in today's heterogeneous network environments. Extending RBAC takes the user management to the next level through normalization and centralization of access controls in these mixed environments. Once such controls are in place, administrators can specify that users have only the very granular controls they need to do exactly what they need to do and no more. Achieving this level of control is more compelling – even required – in today's IT-driven business environment.

[25] http://csrc.nist.gov/publications/nistpubs/800-36/NIST-SP800-36.pdf, page 17 (Oct 2003)

About the Author

J. Michael Butler, GSEC CISA GCFA EnCE, is an information security consultant with LPS, a leading provider of technical services to the mortgage industry. Butler's responsibilities have included computer forensics, information security policies, enterprise security incident management planning, internal auditing of information systems and infrastructure, service delivery, and distributed systems support. He has also been involved in authoring SANS security training courseware, position papers, articles, and blogs. Butler has more than 28 years of experience in the computer industry.

SANS would like to thank this paper's sponsor