Role-Based Access Control (RBAC)

Nicolas Adrién | GeneXus Training

Role-Based Access Control (RBAC)

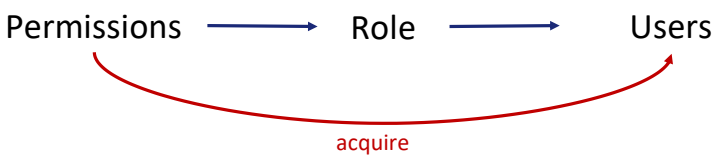
Confidentiality	Integrity	Information Availability

Role-Based Access Control, or RBAC for short, is an approach to restrict system access to authorized users.

It aims to ensure the confidentiality, integrity, and availability of information since, based on the principle of least privilege, it restricts users' access and the actions they can perform.

This reduces the risk of security breaches when a user's account is compromised.

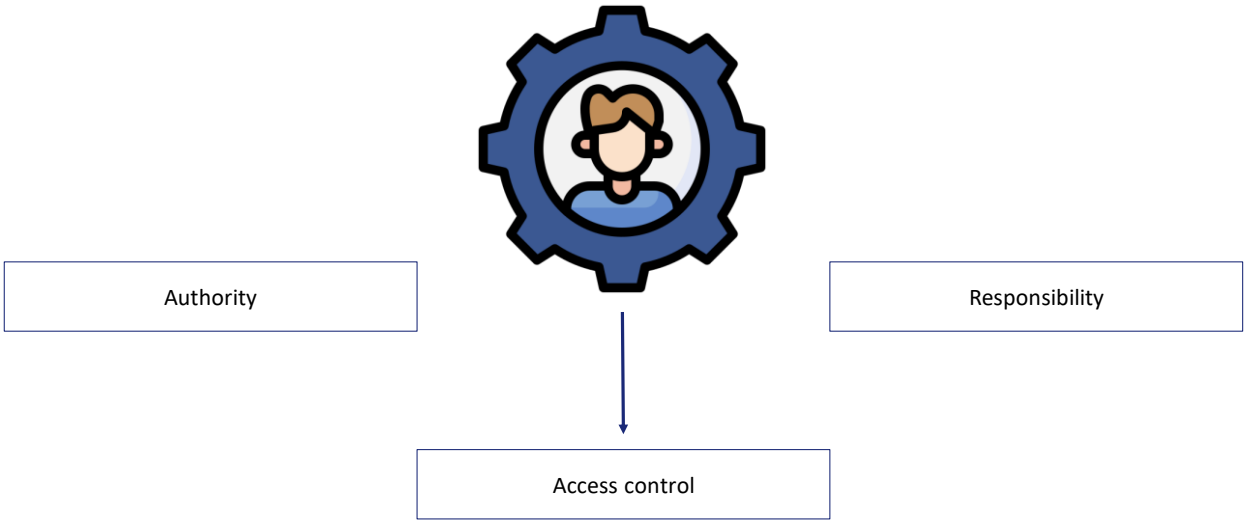
Role-Based Access Control (RBAC)



Within an organization, **roles** are created for various functions.
Permissions to perform certain operations are assigned to specific roles.
Users are assigned particular roles and, through these role assignments, they acquire permissions to perform specific computer system functions.

Since users are not assigned permissions directly, as they only acquire them through their role(s), managing individual user rights becomes a matter of simply assigning the appropriate roles to the user's account; this simplifies common operations, such as adding a user or changing a user's department.

Roles



Among the key elements of RBAC, first we find roles.

A role is a job function or title within an organization with some associated semantics regarding the authority and responsibility granted to a function member.

It is properly viewed as a semantic construct around which access control policy is formulated.
The particular collection of users and permissions brought together by a role is transitory, but the role is more stable because an organization's activities or functions tend to change less frequently.

With RBAC, it is possible to predefine the relationships between roles and permissions, which simplifies the assignment of users to predefined roles.

What is the difference between groups and roles?

Groups

Roles

The concepts of Groups and Roles are often mixed up.
A significant difference between most implementations of groups and the concept of roles is that groups are usually treated as a collection of users and not as a collection of permissions.

Groups of users as the unit of access control are commonly implemented in many access control systems.

On the other hand, roles are both a collection of users and a collection of permissions.

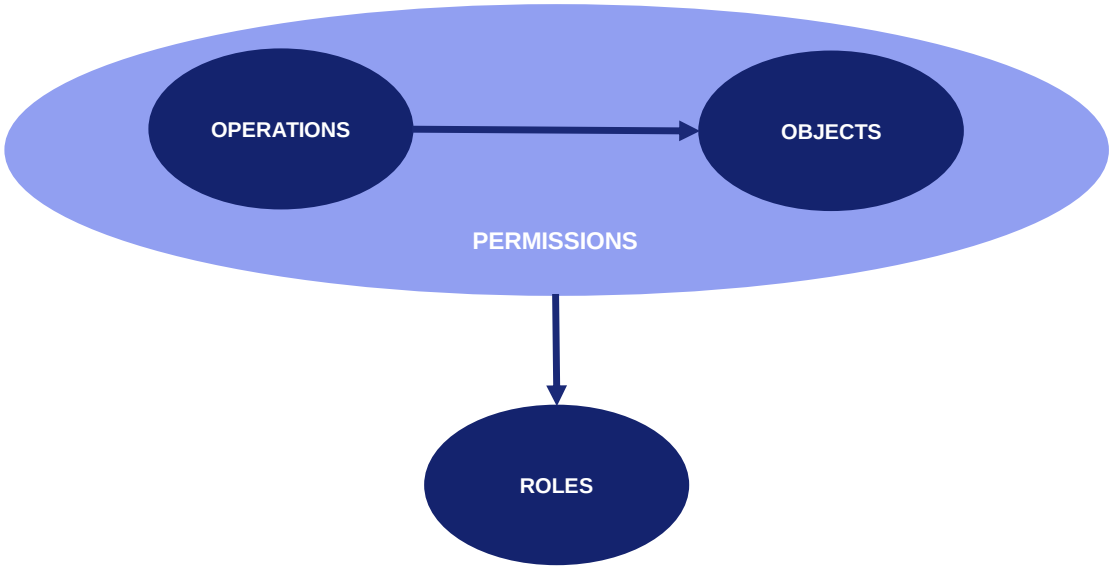
Roles act as intermediaries to link these two collections.

Users

Next, we have Users.
In the RBAC model, users are human beings.

The concept of users can be generalized to include intelligent autonomous agents such as robots, computers, or even computer networks.

Permissions



As for Permissions, they describe the possibility of performing an operation on a given object, such as the authorization of a particular mode of access to one or more objects in the system.
They can be associated with one or more roles, but the collection of permissions linked by roles is transitory.

Access decisions are made according to the permissions associated with the active roles in each session.

Some access control literature talks about “negative permissions” which deny, rather than confer, access. In our framework, denial of access is modeled as a constraint rather than a negative permission.

Sessions

Lastly, there are Sessions, which map users to their active roles.

As for users and sessions, one can have more than one session established at the same time. Each user establishes a session during which they activate some subset of roles for which they are authorized.

Sessions

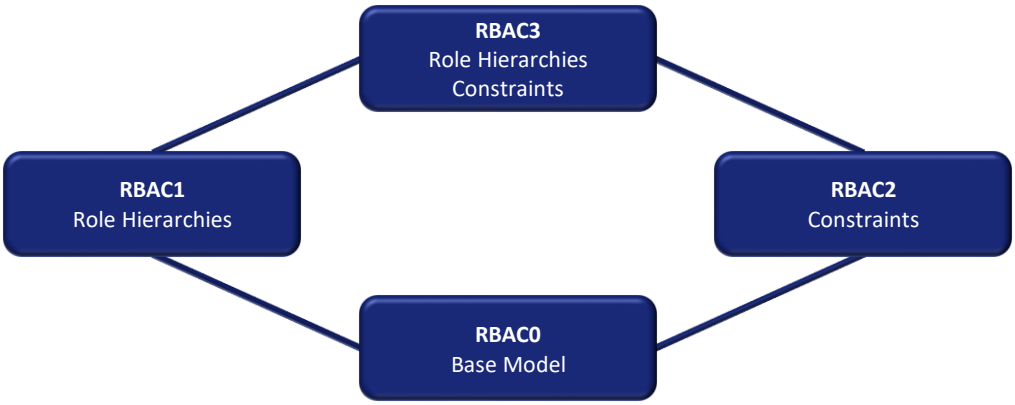
Set of valid session identifiers of Session type

Session Mapping with User and active roles

The state of the system sessions is divided into two parts:
The first one is a set of valid session identifiers of Session type.

The second part is a function that maps each session to the user who owns it and the set of their active roles.

Levels



To understand the various dimensions of RBAC, a family of four conceptual models is defined in it.

First, there is RBAC0, the base model located at the bottom, which is the minimum requirement for any system that intends to support RBAC.

One step higher are RBAC1 and RBAC2, which include RBAC0, but add features to it:

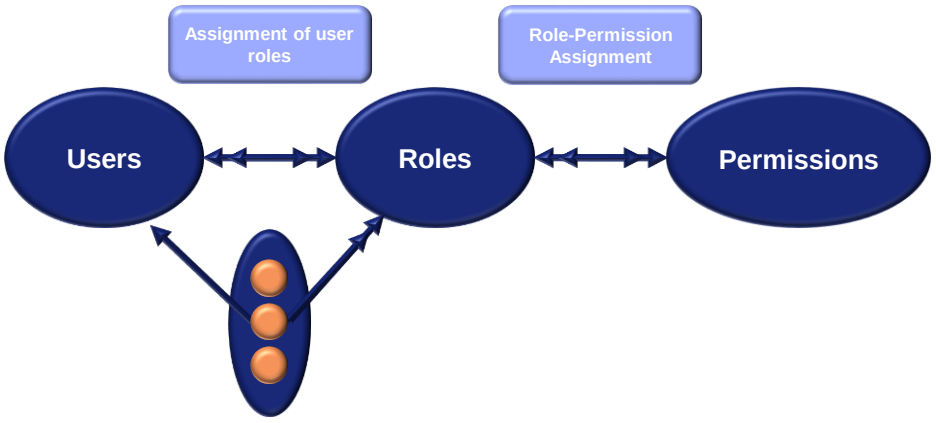
- RBAC1 adds the concept of role hierarchies, which are situations where roles can inherit permissions from other roles.
- RBAC2 adds constraints, which impose restrictions on the acceptable configurations of the different components of RBAC.

RBAC1 and RBAC2 cannot be compared to one another.

Finally, in the last step we have RBAC3, which includes RBAC1 and RBAC2, and by transitivity RBAC0.

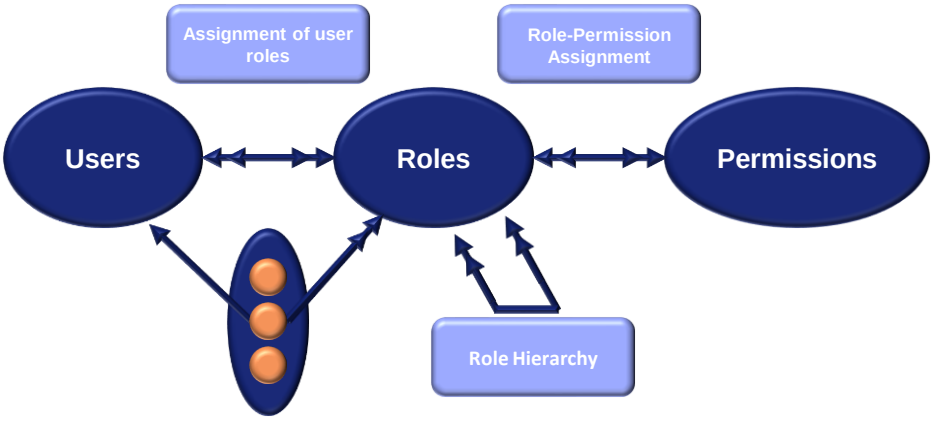
These models are intended to be benchmarks for comparison with systems and models used by other researchers and developers. They can also serve as a guide for product development and evaluation by prospective customers.

Level 0 – Base Model



In the base model are the four sets of entities mentioned above: users, roles, permissions, and sessions. As seen in the diagram, a user can be a member of many roles and a role can have many users. Likewise, a role can have many permissions and the same permission can be assigned to many roles.
The key to RBAC lies in these two relationships.
Finally, a session can be composed of one user and many roles.

Level 1 – Role Hierarchy



The RBAC1 model introduces role hierarchies, which can be defined as a natural means for structuring roles to reflect the lines of authority and responsibility in an organization.

Role hierarchies are almost inevitably included whenever roles are analyzed, and are also commonly implemented in systems that provide roles.

Let's look at an example...

Level 1 – Role Hierarchy - Example

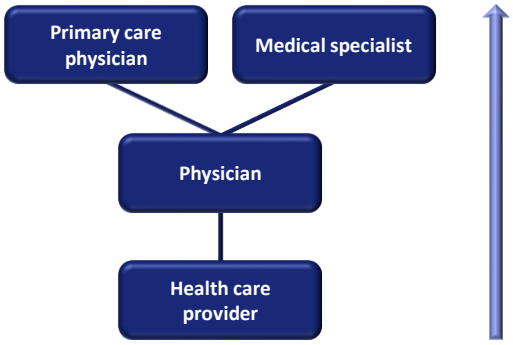


Figure A

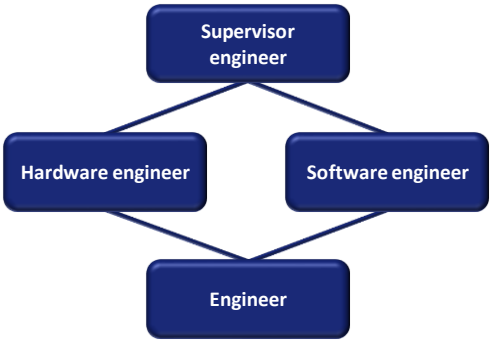


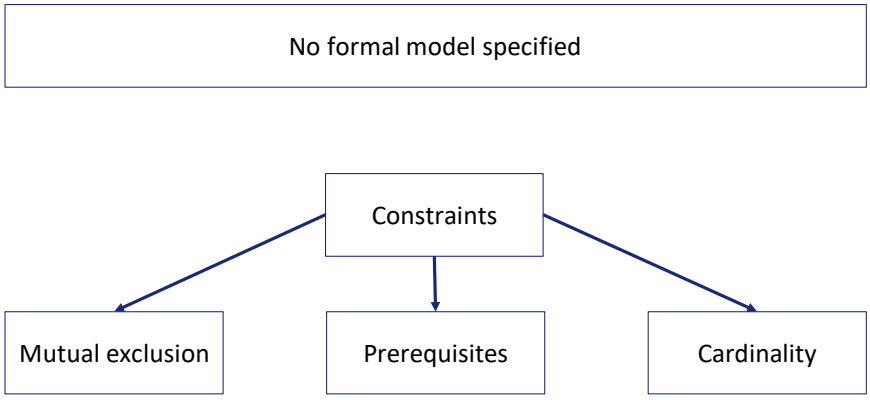
Figure B

By convention, the most powerful (or senior) roles are shown at the top of these diagrams and the less powerful (or junior) roles are at the bottom.

In Figure A, the most junior role is the health care provider. The physician role is senior to the health care provider and therefore inherits all permissions from it. The physician role may have permissions on its own, in addition to those inherited from the health care provider role.
Since permission inheritance is transitive, the primary care physician role inherits permissions from the physician and health care provider roles.
Both the primary care physician and the specialist physician inherit permissions from the physician role, but each one will be assigned different permissions directly.

In addition, Figure B also shows inheritance, but this time referring to multiple inheritance of permissions, where the supervisor engineer role inherits from both the hardware engineer role and the software engineer role.

Level 2 – RBAC0 + Restrictions



The RBAC2 model introduces the concept of constraints. There is no difference with RBAC0, except that it requires a set of constraints that determine whether the values of various components of RBAC0 are acceptable or not. Only acceptable values will be allowed.

Constraints are an important aspect of RBAC. Sometimes it is argued that they are the main drivers for access control because they are a powerful mechanism for designing high-level organizational policies.

Once certain roles are declared as mutually exclusive, it is not necessary to be concerned about assigning roles to individual users. The latter can be delegated and decentralized without the risk of compromising the organization's overall policy objectives.

We can mention some examples of Constraints:
One might be Mutual Exclusion, where a constraint might be that only one role in the set can be assigned or activated. This type of constraint can be used to enforce the separation of duties.

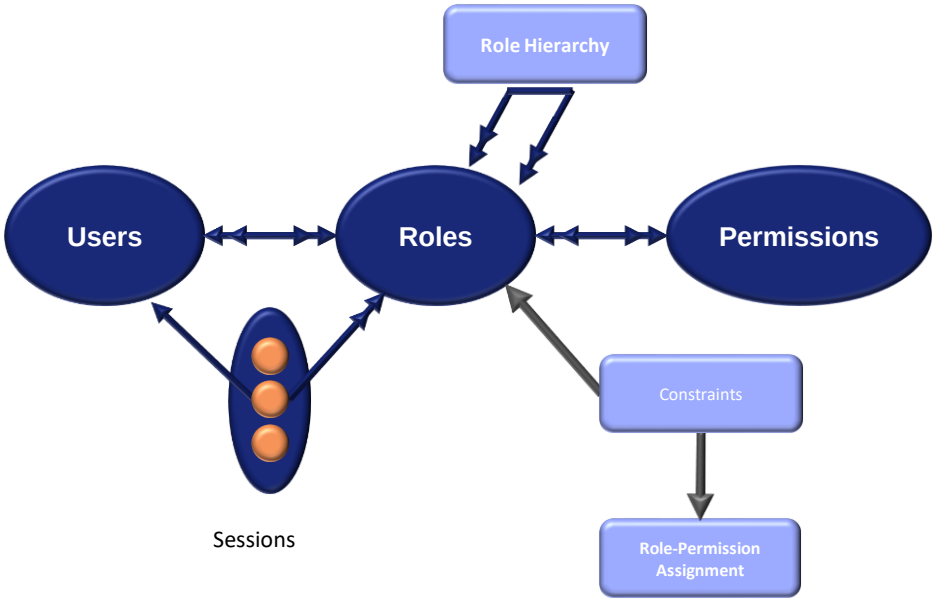
Another type can be Prerequisites, where an example of a constraint could be that a role can be assigned to a user only if the user has some other prerequisite role. In this case, these can be used to enforce the principle of least privilege.

Finally we have Cardinality, where there can be constraints such as:

- Maximum number of users that can be assigned a role.
- Maximum number of functions that any user can have (possibly, in a session).

- Maximum number of roles that have a certain permission.
- And more.

Level 3 – Consolidated Model



Finally, we have the RBAC3 model, which combines RBAC1 and RBAC2 to provide role hierarchies and constraints simultaneously. Therefore, several matters arise when these two concepts are brought together:
Constraints can be applied to the role hierarchy itself, and can limit the number of senior (or junior) roles that a given role can have.
They can also restrict two or more roles from having a senior (or junior) role in common.

These types of constraints are useful in situations where the authority to change the role hierarchy has been decentralized, but the security administrator wants to restrict the overall manner in which such changes can be made.

training.genexus.com
wiki.genexus.com
training.genexus.com/certifications