Brain Mentors Pvt Ltd
Authentication and Authorisation

Authentication : Verifying the identity of a user or entity requesting access to a system, network, or application.

Authorization: is the process of granting or denying access to specific resources or actions based on the authenticated identity of a user or entity.
Once a user or entity has been successfully authenticated, authorization determines what they are allowed to do within a system, application, or network.

Methods of

Authorizations:

1. DAC (Discretionary Access Control)
2. MAC (Mandatory Access Control)
3. RBAC (Role Based Access Control)
4. ARBAC (Attribute Based Access Control)

We will focus on: RBAC (Role-Based Access Control)

Access control

Access control is only one aspect of a computer security solution, but it is one of the most visible. Every time a user logs on to a multiuser computer system, access control is enforced.
To gain a better understanding of the

purpose of access control, Information security risks can be broadly categorized into the following three types:

1 **Confidentiality: -** refers to the need to keep information secure and private.
2 **Integrity: -** refers to the concept of protecting information from being improperly altered or modified by unauthorized users
3 **Availability: -** refers to the notion that information is available for use when needed.

Operation = Any action, such as CRUD operations
Object   = Reference to any object instance
Permission = Mapping of 'Operation' +

'Object'

Introduction

Role-Based Access Control (RBAC) is a widely adopted access control model that provides a flexible and scalable approach to managing user access in software systems. This document aims to provide a detailed understanding of RBAC, its core concepts, and how it can be implemented effectively.

Overview

RBAC is based on the concept of roles, which are collections of permissions that define the actions a user can perform within a system. By assigning roles to users, RBAC ensures that access rights are granted based on predefined roles rather than individual permissions.

Key Components: -

Roles

Roles represent a set of permissions that define the activities or operations a user can perform. Roles can be categorized into different levels, such as administrative roles, user roles, or custom roles specific to an application. It is important to define roles with granularity to ensure proper access control.

Permissions

Permissions define the specific actions or operations that a user can perform within the system. Permissions are associated with roles, and users acquire these permissions through their assigned roles. Examples of permissions include read, write, delete, create, or execute.

Users

Users are individuals who interact with the system and require access to perform their tasks. Users are assigned roles based on their responsibilities and the level of access they require. It is essential to authenticate and authorize users properly before granting them access to the system.

Resources

Resources refer to the objects or data within the system that users can access or manipulate. These can include files, databases, network services, or any other component of the software system. Access to resources is controlled through RBAC based on user roles and associated permissions.

RBAC Model

The RBAC model consists of several key elements that interact to control access within a system. Understanding these elements is crucial for implementing RBAC effectively.

Role Assignment

Role assignment involves associating roles with users based on their responsibilities and access requirements. This can be done manually by system administrators or automatically based on predefined rules and user attributes.

Role Authorization

Role authorization determines whether a user with a specific role is allowed to perform a particular action or access a specific resource. The RBAC model ensures that users are only authorized to perform actions associated with their assigned roles.

Role Hierarchies

Role hierarchies define relationships between roles, enabling inheritance of permissions and simplifying role management. Roles can be organized in a hierarchical structure, where higher-level roles inherit permissions from lower-level roles. This reduces the effort required to manage permissions for each individual role.

Separation of Duties

Separation of duties is a security principle that aims to prevent conflicts of interest and reduce the risk of fraud or malicious activities. RBAC allows defining constraints to ensure that certain roles or combinations of roles are not assigned to the same user, thus maintaining an appropriate level of control and accountability.

RBAC Implementation

Implementing RBAC involves several steps to ensure its successful integration within a software system.

Identify Roles and Permissions

Identify the roles and corresponding permissions required within the system. Collaborate with stakeholders, system administrators, and security experts to determine the necessary access levels and define roles that align with the organization's structure and security policies.

Define Role Hierarchies

Establish role hierarchies to simplify

role management and reduce administrative overhead. Determine the inheritance relationships between roles, ensuring that higher-level roles inherit permissions from lower-level roles where appropriate.

Assign Roles to Users

Assign roles to individual users based on their responsibilities and access requirements. This can be done through manual assignment or automated processes such as user provisioning or role-based provisioning systems.

Enforce Access Control

Implement mechanisms to enforce access control based on RBAC. This may involve integrating RBAC rules into the system's authentication and authorization processes, ensuring that users can only access resources and perform actions associated with their assigned roles and permissions.

Regular Review and Maintenance

Regularly review and update role assignments, permissions, and role hierarchies to accommodate changes in the organization's structure, responsibilities, and security requirements. Conduct periodic audits to ensure RBAC policies are being followed and to identify any potential vulnerabilities or access control gaps.

Conclusion

RBAC provides a flexible and scalable approach to access control, enabling organizations to manage user permissions effectively. By assigning roles to users and defining permissions at a granular level, RBAC improves security, simplifies administration, and enhances overall system integrity. Following the guidelines outlined in this documentation will assist in implementing RBAC successfully within your software system.

Example of Implementation of RBAC

1. **Identify Roles:**
   - Administrator: Has full access to all project management functionalities, including creating, editing, and deleting projects, managing user roles, and generating reports.
   - Project Manager: Can create and manage projects, assign tasks to team members, and generate project reports.
   - Team Member: Can view assigned tasks, update task status, and

collaborate within the project.

2. **Define Permissions:**
   - Create Project: Permission to create new projects.
   - Edit Project: Permission to modify project details, such as project name, description, and deadlines.
   - Delete Project: Permission to delete a project.
   - Assign Task: Permission to assign tasks to team members within a project.
   - Update Task Status: Permission to update the status (e.g., in progress, completed) of assigned tasks.
   - Generate Reports: Permission to generate reports for projects and tasks.
3. **Create Role Hierarchies:**
   - The Administrator role has all permissions.
   - The Project Manager role has permissions to create projects, edit

projects, assign tasks, update task status, and generate reports.
- The Team Member role has permissions to view assigned tasks and update task status.

4 **Assign Roles to Users:**
- The system administrator is assigned the Administrator role.
- Project managers are assigned the Project Manager role.
- Team members are assigned the Team Member role.

5 **Implement Access Control:**
- When a user logs into the project management application, their credentials are verified using username and password authentication.
- Once authenticated, the application retrieves the user's assigned role(s) from the database.
- Each page or functionality within the application checks the user's role and associated permissions

before allowing access.
    - For example, only users with the Administrator role can access the user management page to create or modify user roles.

6. **Regular Review and Maintenance:**
    - Regularly review and update role assignments, permissions, and role hierarchies as organizational needs and responsibilities change.
    - Conduct periodic audits to ensure that users are appropriately assigned roles and that access controls are working as intended.
    - Revise role assignments if employees change roles or responsibilities within the organization.

By implementing RBAC in this project management application, the system ensures that users can only perform actions that align with their assigned roles and permissions. This approach improves security, simplifies administration, and enables granular control over access to project-related functionalities.

Permissions Table: Mysql
id, name, active, isDeleted, Url

Roles
id, name,

Role_Permissions
rolePermissionId, roleId, permissionId

Users
Id, name, email, password, etc

User_Roles
userRoleId, userId, roleId

Extra

The central notion of RBAC is that permissions are associated with roles, and users are assigned to appropriate roles.

"What is the diference between roles and groups?"
A major diference between most implementations of groups and the 3 concept of roles is that groups are typically treated as a collection of users and not as a collection of permissions. A role is both a collection of users on one side and a collection of permissions on the other. The role serves as an intermediary to bring these two collections together.

A role is a job function or job title within the organization with some associated semantics regarding the authority and responsibility conferred on a member of the role.

A permission is an approval of a particular mode of access to one or more objects in the system.

The nature of a permission depends greatly on the implementation details of a system and the kind of system that it is.

A user may have multiple sessions open at the same time, each in a different window on the workstation screen for instance. Each session may have a different combination of active roles

Https://www.brain-mentors.com