Spectrum Journal of Innovation, Reforms and Development
Volume 18, August, 2023 | ISSN (E): 2751-1731
Website: www.sjird.journalspark.org

PERMISSION RESTRICTION METHOD AND ALGORITHM IN DISTRIBUTED DATABASE

Sadikov Sh.M.
Associate Professor of Tashkent University of Information Technologies named after Muhammad al-Khwarizmi

Abstract
The article analyzes the distributed database architecture, permission management methods, existing threats and vulnerabilities in the distributed database, features of permission restriction and their application in the distributed database, Role-based access control and Attribute-based access control methods. an enhanced, distributed resolution constraint algorithm is developed.

Keywords: distributed database, replication, access control, auditing and monitoring, role-based access control (RBAC), ABAC model, encryption, users, resources, roles, permissions, distributed access restriction.

Introduction

The architecture of distributed database systems is designed to store data across multiple interconnected nodes or servers. These nodes can be distributed geographically to form a distributed network. The architecture is designed to provide several advantages, including increased scalability, fault tolerance, and improved performance. However, this distributed nature creates difficulties in managing access and restricting permissions.

Although distributed database systems have many advantages, they present a number of problems related to access control and permission restrictions. Effectively addressing these issues requires sophisticated access control models, encryption, use of distributed coordination protocols, and careful planning of access control policies to ensure data security, privacy, and consistency in a distributed environment.

Due to their complex architecture and decentralized nature, distributed databases are susceptible to various security threats and risks. Addressing these threats is critical to ensuring data privacy, integrity, and usability. Some of the common security threats and risks associated with distributed databases are:

Unauthorized Access: Malicious individuals may attempt to gain unauthorized access to a distributed database to steal confidential information or disrupt its normal operation. Weak authentication mechanisms and poor access controls can leave a system vulnerable to unauthorized access.

Data Corruption and Data Leakage: A breach in any of the distributed nodes can lead to a breach of confidential data, especially when data replication is involved. Access to data by

unauthorized users may lead to data leakage, which may compromise the confidentiality and privacy of data.

Distributed denial of service (DDoS) attacks: DDoS attacks aim to overwhelm the resources of a distributed database, making it unavailable to legitimate users. Attackers can exploit vulnerabilities in the network or application layer to flood the system with excessive traffic, causing service interruptions.

Data Corruption and Integrity Attacks: Hackers can attempt to modify, alter, or corrupt data stored in a distributed database, leading to data integrity issues. Without strong integrity controls, such attacks are difficult to detect and remediate.

Insider Threats: Employees, contractors, or other insiders with access to a distributed database can abuse their privileges to compromise data or compromise the system. Insider threats can be difficult to detect because these users often have legitimate access to the system.

Replication and synchronization vulnerabilities: Replication of data across distributed nodes creates the risk of data inconsistencies or errors propagating through the system. Synchronization mechanisms must be secure and reliable to avoid replication vulnerabilities.

Man-in-the-Middle (MITM) Attacks: If communication channels are not adequately secured, attackers can intercept and modify data exchanged between distributed nodes. MITM attacks can lead to data corruption or unauthorized access to sensitive information.

Injection attacks: SQL injection and other types of injection attacks can use vulnerabilities in application code or database queries to gain unauthorized access to a distributed database. Access validation and parameterized queries are important to prevent injection attacks.

Leaking insider information: Malicious insiders may attempt to extract confidential information from a distributed database or share it with outside parties for personal gain. Monitoring and auditing of user activity helps detect and prevent data transmission attempts.

Lack of centralized control: The distributed nature of a database can make it difficult to maintain centralized control over access and security policies. Consistent implementation of security measures across nodes can leave certain areas vulnerable.

It is recommended to use hybrid models to achieve effective access control. Below are a few permission control models.

Role-based access control (RBAC) is an access control model that provides a systematic and centralized approach to managing user permissions in an organization's information systems, including distributed databases. RBAC is based on the concept of roles, where permissions are tied to specific roles rather than directly to individual users. This provides a number of advantages, particularly in distributed database environments where access control management can be complex due to the decentralized nature of the system.

Figure 1 presents the RBAC model.

Basic concepts of RBAC:

Roles: Roles represent different job functions or responsibilities within an organization. Examples of roles may include "Administrator", "Manager", "Employee", "Customer", etc.

Each role is associated with a set of permissions that determine what actions users assigned to that role can perform.

Permissions: Permissions are specific actions or operations that users can perform within the system. These include reading, writing, updating, deleting, executing, etc. depending on system requirements.

Users: Users are individuals or legal entities who interact with the system and are assigned certain roles. A user can have multiple roles depending on their responsibilities and access needs.

Role Assignment: The process of assigning users to specific roles is called role assignment. It defines which roles are associated with each user in the system. Users are granted access to resources based on their assigned roles, simplifying permission management.

Role hierarchies: Some RBAC implementations support role hierarchies, where roles can be organized in a hierarchical structure. This means that roles can inherit permissions from roles higher in the hierarchy. For example, the Manager role can inherit some permissions from the Employee role.

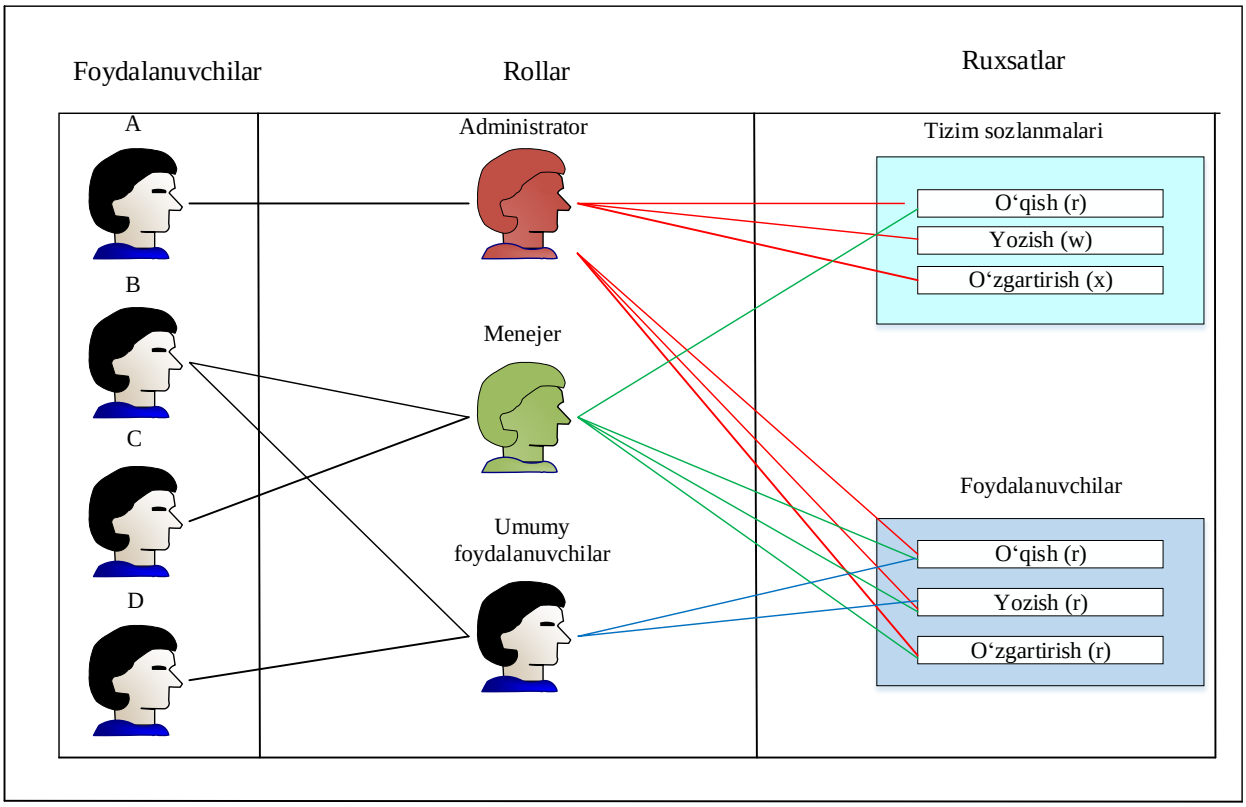

Figure 1. RBAC model of permission control

Advantages of using RBAC in a distributed database environment:

Simplified Access Control: RBAC simplifies access control in distributed databases by centralizing permissions and policy enforcement. Instead of managing permissions for each user individually, access control is based on roles assigned to users.

Consistency across distributed nodes: In a distributed database environment, data is stored across multiple nodes. Ensuring a consistent access control policy across all nodes is critical

to data security. RBAC enables access control policies to be defined and distributed centrally, ensuring that the same roles and permissions are applied throughout the system.
Scalability and flexibility: As distributed databases grow in size and complexity, managing individual user permissions can become difficult. RBAC's role-based approach is scalable and flexible, making it easy to adapt access control policies as an organization evolves.
Mitigating Insider Threats: Insider threats where authorized users abuse their privileges can be difficult to detect. RBAC helps mitigate insider threats by limiting users' permissions to only those necessary for their roles, reducing the potential for data breaches or unauthorized access.
Enhanced Auditing and Compliance: RBAC's centralized role assignment and policy enforcement enables comprehensive audit trails, facilitating compliance with regulatory requirements and internal security policies. Detailed logs of user activity can be generated, simplifying compliance audits and post-event investigations.
Improve data privacy: In distributed database environments, data can be replicated across nodes for redundancy and availability. RBAC ensures consistent application of access control policies across all instances, protects data privacy, and prevents unauthorized access to sensitive data.
It is more effective to use it together with other models to improve the RBAC system.
Attribute-Based Access Control (ABAC): Combining RBAC with Attribute-Based Access Control (ABAC) to include additional attributes and policies for access decisions. ABAC provides granular control that takes into account various attributes, such as user attributes, environmental conditions, and data sensitivity levels.

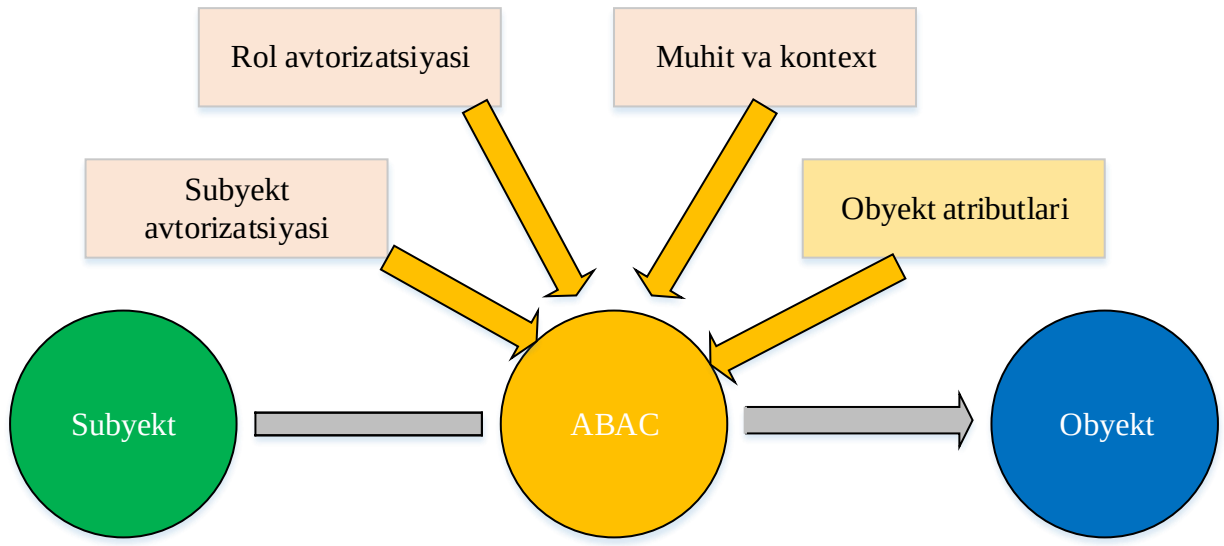

Figure 2. ABAC model of authorization management

These modifications and enhancements to traditional RBAC can tailor the access control mechanism to meet the specific requirements of distributed database environments. By combining RBAC with these enhancements, organizations can achieve a more robust and flexible access control system that ensures data security, privacy and compliance in a distributed environment.

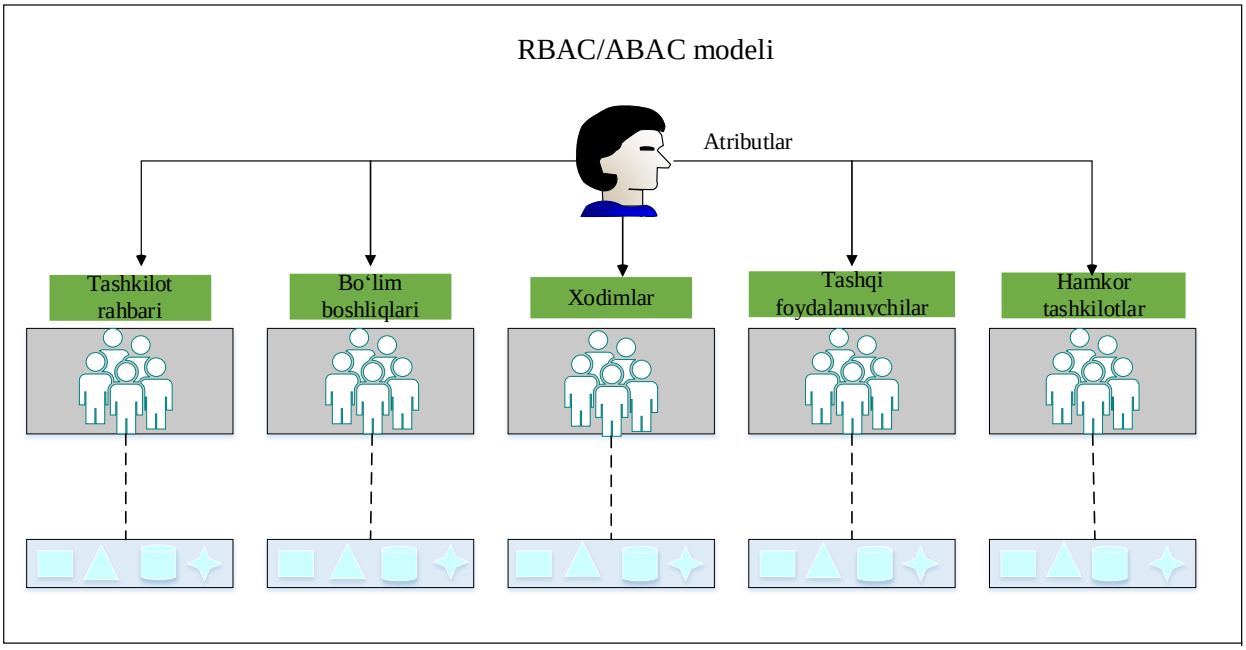

Figure 3. An improved RBAC model

Table 1. Permissions Management Matrix

Information / Subject	Open information	Confidential information	Information about employees	Financial information	Production information
The head of the organization	rwx	rwx	rw	rw	rwx
Department heads	rw	rwx	rw	rw	rw
Employees	r	w	-	-	r
External users	r	-	-	-	r
Partner organizations	r	-	-	-	r

Designing a new algorithm to limit access in a distributed database environment requires careful consideration of various factors, including data consistency, scalability, security, and performance. Below is an algorithm for restricting access in a distributed database:

The mathematical formulation of an access control algorithm in a distributed database involves defining various sets, relationships, and logical expressions for access control policy and decision making. Below are some basic mathematical formulas for a permission control algorithm in a distributed database:

Collections and Relationships:

Users (U): The collection of all users in a distributed database system.

Resources (R): Collection of all resources (data, files, etc.) in a distributed database.

Roles: A set of predefined roles to which users can be assigned.

Permissions (P): The set of all permissions that can be granted to users.

Role Assignment (UA: U → Roles): A relation that maps each user to the roles they are assigned.
Resource Ownership (OR: R → U): A relation that maps each resource to its owner user.
Access Control Policy:
Policy Concept (PE): A logical expression that defines an access control policy. For example, PE1: "If user U1 has role R1, read and write permissions are granted on resource R2."
Policy Set (PS): A set containing all policy expressions.
$PS = \{P_{E1}, P_{E2}, ..., P_{EN}\}$
Attribute-Based Access Control (ABAC):
Attribute set:
User Attributes (Attributes_U): Collection of all user attributes.
Resource Attributes (Attributes_R): Collection of all resource attributes.
Attribute values (AttValues_U: U → Attributes_U, Attributes_R: R → Attributes_R):
A function that compares each user with attribute values.
A function that maps each resource to its attribute values.
Attribute-based policy:
Policy expression using attributes and attribute values. For example, PE2: "If user U1.department = 'HR', allow writing on resource R3."
Context-based access control:
Context Collections:
Time (T): The set of all possible time values.
Location (Loc): The set of all possible location values.
Environmental Factors (Env): The set of all possible values of environmental factors.
Contextual attribute values (AttValues_T: U → T, AttValues_Loc: U → Loc, AttValues_Env: U → Env):
Functions that compare each user with contextual attribute values.
Context-based policy:
A policy expression considering contextual attributes. For example, PE3: "IF user U1.time = '08:00 - 17:00' AND user U1.location = 'Office' THEN allow read on resource R4."
Dynamic role assignment:
Dynamic role assignment function (RAF: U → Roles):
A feature that dynamically assigns roles to users based on certain conditions or events. For example, if U belongs to the "Management" department, RAF(U) = R1.
Policy evaluation and decision making:
Policy decision function (PD: U × R → P):
A function that takes user attributes, resource attributes, and context attributes as input and issues an access control decision (permission) for a given user and resource combination.
PD(U, R) = P1 if (user U has role R1 AND user U.time = '08:00 - 17:00') OR (user U.department = 'HR' AND resource R.owner = U)
These mathematical formulas are used in the permission control algorithm in a distributed database. The algorithm uses these formulas to evaluate access control policies, assign

dynamic roles, consider contextual attributes, and make access control decisions based on user attributes, resource attributes, and environmental factors.

Distributed Access Constraint Algorithm (TRCHA)

Step 1: Enable Role-Based Access Control (RBAC).

An RBAC model is established by defining roles, permissions, and users. A central repository is created to store roles and permissions policies.

Step 2: Data replication (replication) and synchronization

Data is replicated across distributed nodes for fault tolerance and availability. A synchronization mechanism is established to ensure data consistency between replicated data.

Step 3: Define a dynamic role

Dynamic role assignment is allowed based on user behavior, user attributes, and context factors. An automated mechanism is introduced to configure roles based on user activity and role-based rules.

Step 4: Context-based access control

Enhances the RBAC model with context-based access control that takes into account time, location, and device attributes. Enables file access control based on user context and attributes.

Step 5: Multi-Factor Authentication (MFA)

Integrate MFA into the access control process to improve user authentication. Requiring multiple authentication factors before allowing access to sensitive resources.

Step 6: Attribute-Based Access Control (ABAC)

Extending the RBAC model with ABAC to include additional attributes and policies for access decisions. Consider data sensitivity and environmental conditions when making access control decisions.

Step 7: Enable/Disable Time Based Role

Allow time-based activation and deactivation of roles to ensure access control changes at specific time intervals. Automate role activation/deactivation based on predefined schedules or events.

Step 8: Role-based encryption

Implement role-based encryption to protect sensitive data. Encrypt data based on user roles, ensuring that only authorized users can access and decrypt certain data.

Step 9: Attribute-level access control

Enable attribute-level access control to provide fine-grained control over access to sensitive information in records. Allowing certain attributes to be restricted based on user roles and permissions.

Step 10: Empowered management

Implement empowered management to decentralize role management while supporting central control. Allow authorized users to assign roles to specific subsets of users.

Step 11: Distributed Consistency and AuditingBarcha ruxsat o‘zgarishlari va rollar taqsimoti taqsimlangan tugunlar bo‘ylab izchil targ‘ib qilinishiga ishonch hosil qilish.

Kirishni kuzatish va foydalanuvchi faoliyatini kuzatish uchun keng qamrovli audit va jurnallar mexanizmlarini joriy qilish.

Step 12: Role Inheritance Restrictions

Set restrictions on role inheritance to control the propagation of permissions from higher-level roles to lower-level roles. Prevent inheritance of excessive permissions to prevent misuse.

Step 13: External object integrationAllow seamless integration of external objects into the permission constraint model. Implement secure mechanisms to manage access control collaboration with external partners or contractors.

Step 14: Security and Privacy Issues

Ensuring that all communication between distributed nodes is encrypted and secured. Implement secure authentication and access controls on each node.

The Distributed Access Restriction Algorithm Algorithm is a comprehensive access control approach in a distributed database environment that combines role-based, attribute-based, and context-based access control. It provides scalability, flexibility and security, solving various problems in distributed database systems while ensuring data consistency and confidentiality.

List of Used Literature

1. Sh.M.Sadikov “Ptotection of databases corporate information systems” Academic international conference on Multi-disciplinary studies and education, USA 2023.
2. Sh.M.Sadikov “Classification of information security thereats in the database” Innovate research in modern education, Canada 2023.
3. Cigdem Bakir, Mehmet Guchli “A New Scalable and Expandable Access Control Model for Distributed Database Systems in Data Security” Scientific Programming 2020 DOI:10.1155/2020/8875069
4. Maarten van Steen & Andrew S. Tanenbaum “A brief introduction to distributed systems” SpringerLink 2016
5. David W. Chadwick “Secure Multi-Party Non-Repudiation Protocols and Applications” 2015
6. Hassan A “Database Security and Auditing: Protecting Data Integrity and Accessibility” 2021
7. Ravi S. Sandhu “Database Security and Integrity” 1994
8. Alton Chung and Sheng-Uei Guan “Database Security: From Legacy Systems to Blockchain Technology” 2021
9. John R. Vacca “Computer and Information Security Handbook” 2021.