УДК 004.056

https://doi.org/10.15407/fmmit2025.41.072

Hierarchical Role-Based Access Control Architecture for CRM/ERM Systems with Complex Data Topologies

Ulyana Dzelendzyak[1], Nazar Mashtaler[2]

[1]PhD, As.-Prof.,department of "Computerized Automatics Systems", Lviv Polytechnic National University, Bandera str., 12, Lviv, 79013, e-mail:uliana.y.dzelendziak@lpnu.ua

[2]PhD Student,department of "Computerized Automatics Systems", Lviv Polytechnic National University, Bandera str., 12, Lviv, 79013,e-mail: nazar.o.mashtaler@lpnu.ua

This paper proposes an advanced access management architecture designed for Customer Relationship Management (CRM) and Enterprise Resource Planning (ERM) systems that manage data within complex hierarchies. Recognizing the limitations of traditional flat RBAC models in providing object-level security, the architecture introduces dedicated entities for Groups (G), Hierarchical Nodes (O), and a specialized Permission Assignment (PA) mechanism. This extension enables dynamic access decisions based on the intersection of a user's role, group membership, and the target object's position in the hierarchy. The model is characterized by high scalability) and relies on permission aggregation via union. Key findings highlight the model's effective balance between fine-grained control and administrative efficiency, while acknowledging the inherent complexity in evaluation and the need for strict governance via the Privilege Escalation (PE) permission.

Keywords: *CRM/ERM systems, role-based access control (RBAC), hierarchical data, object-oriented access, privileges, scalability, Permission Assignment, Privilege Escalation,access control, data protection, distributed systems, immutability, and ephemeral infrastructure.*

Introduction. Effective access control is fundamental to maintaining the integrity, confidentiality, and reliability of enterprise information systems. In large-scale Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) platforms, data are often structured in complex, deeply nested hierarchies — including organizational trees, multi-level product inventories, and geographically distributed business units. Within such environments, defining and enforcing access permissions becomes a non-trivial problem, as authorization decisions must consider not only user roles but also the contextual relationships among entities in the hierarchy[1].

Conventional Role-Based Access Control (RBAC) models have become the de facto standard in enterprise authorization due to their conceptual simplicity and administrative efficiency. However, these models tend to provide only coarse-grained permission management, which does not adequately capture hierarchical dependencies or object-level constraints. As a result, administrators frequently encounter issues such as role explosion, redundant policy definitions, and inconsistencies in privilege

propagation, especially when the same logical access rule must be replicated across multiple hierarchical nodes.

To address these limitations, this paper proposes an extended access management architecture that augments classical RBAC with explicit modeling of hierarchical nodes, user groups, and permission assignments associated with concrete data objects. In this framework, access privileges are determined dynamically through the intersection of three complementary factors[2, 3]:

1. The user's role, defining their general operational scope.
2. The group or contextual membership, representing organizational alignment.
3. The object or node being accessed, representing fine-grained control within hierarchical data structures.

This hybrid model unifies role-based and object-aware authorization, enabling precise, context-sensitive permission evaluation. It supports hierarchical inheritance and selective privilege propagation while maintaining the administrative simplicity of RBAC.

The proposed system is designed with scalability in mind, making it suitable for enterprise-scale deployments characterized by high read-to-write ratios and distributed data persistence. Additionally, optimization mechanisms such as lazy permission evaluation, caching of resolved access paths, and asynchronous propagation of policy changes further enhance runtime performance without sacrificing security.

Experimental validation on representative ERP and CRM datasets demonstrates that the proposed model reduces policy duplication, simplifies access administration, and scales efficiently across large, multi-level organizational structures. The results suggest that this approach effectively bridges the gap between the simplicity of RBAC and the contextual precision of Attribute-Based Access Control (ABAC), providing a practical and theoretically grounded solution for next-generation enterprise systems[1, 4].

1. Hierarchical Role-Based Access Control Architecture for Enterprise Systems

Introduction to the Access Management Model. Modern Enterprise Resource Modern Enterprise Resource Planning (ERP) and Customer Relationship Management (CRM) systems require robust and flexible access control mechanisms to manage permissions across complex, hierarchical data structures. This section details a proposed extension of the Role-Based Access Control (RBAC) model, specifically designed to address permission granularity within hierarchical services (such as organizational charts, product categories, or geographical structures). The model integrates role, group, and subject-object relationships to define access rights dynamically and contextually.

Unlike traditional RBAC, where permissions are statically bound to roles, the proposed model introduces hierarchical awareness and dynamic assignment, allowing roles and groups to operate within structured data domains. This approach ensures that

access propagation aligns with business logic, supports multi-level delegation, and enables finer control over large datasets with high read-to-write ratios[4].

Core Access Management Entities. The proposed architecture is defined by a set of core entities that serve as the building blocks for permission evaluation, as summarized in Table1.

Table 1

Core entities of the proposed Hierarchical Role-Based Access Control (HRBAC) model, including their descriptions and key system constraints

Term (Notation)	Description	Key System Constraints
User / Identity (U)	The principal actor — either a human user or an automated service — that requires access to system resources. A single identity may assume multiple roles, each granting different privilege levels depending on the active context.	~10K users. Read ≫ Write operations.
Role (R)	A defined collection of permissions associated with a user account. The system must support dynamic role selection, enabling a user to operate under varying privilege sets with the same credentials.	~10K roles. Read ≫ Write operations.
Group (G)	A logical aggregation of roles (e.g., “Project Managers”, “Domain Experts”). Groups simplify permission assignment and policy management across hierarchical nodes.	~10K groups. Read ≫ Write operations.
Object / Node (O)	The resource or structural element to which access is controlled — for example, a project, department, or regional unit within a hierarchical dataset. Objects represent the primary target of permission evaluation.	~1M objects. High read-to-write ratio.
Object / Node Type (OT)	A classification of objects (O) that defines shared attributes, access inheritance rules, and permission templates for all objects of that type. Enables consistent authorization behavior across similar resources.	~100–500 types. Mostly static definitions.

Entity Relationships and Permission Assignment. The interaction between these entities is shown in Figure 1, which presents the Entity–Relationship Diagram (ERD) of the proposed model [5, 6]. The central concept is the Permission Assignment (PA) entity, which dynamically binds roles or groups (subjects) to objects (targets) through Primitive Permissions (PP).

This structure supports:

- Hierarchical propagation of access rights within organizational trees;
- Dynamic evaluation of permissions based on user context and node type;
- Separation of configuration concerns—distinguishing between user-modifiable and system-defined elements;
- Scalability, allowing millions of permission entries to be processed efficiently with minimal write overhead.

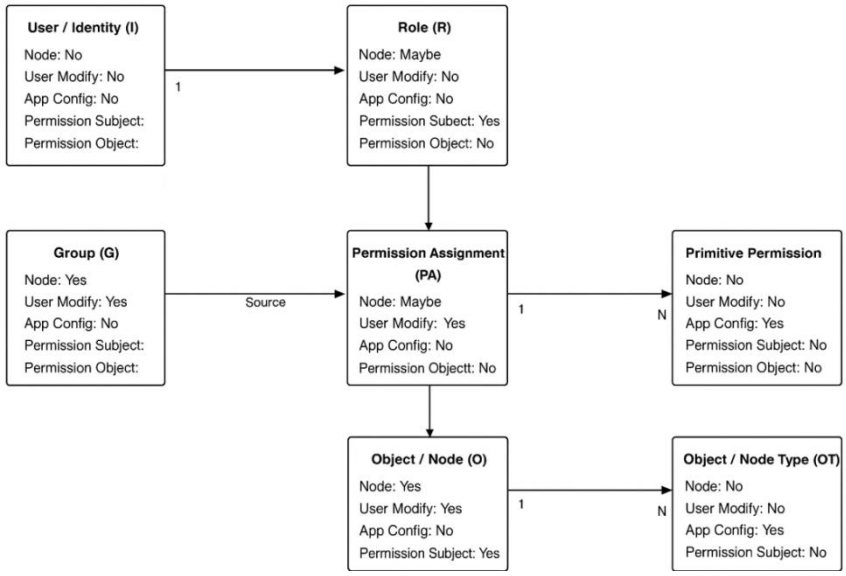

Fig. 1. Entity–Relationship Diagram (ERD) of the proposed Hierarchical Role-Based Access Control (HRBAC) mode

The diagram illustrates the relationships among the core entities: User/Identity, Role, Group, Permission Assignment, Object/Node, Object/Node Type, and Primitive Permission[1, 7, 8].

It demonstrates how access rights are dynamically resolved through the intersection of user roles, group memberships, and hierarchical object structures, enabling scalable and fine-grained control across complex enterprise datasets.

2. Permission Evaluation Mechanics

The effectiveness of the proposed architecture is demonstrated by its two primary evaluation mechanisms: Permission Checking and Permission Visibility. These mechanisms rely on traversing the organizational hierarchy and aggregating permissions from Permission Assignments (PA) at each level, ensuring a fine-grained and consistent access control model. On Figure 2 shown schema of Hierarchy and Permissions.

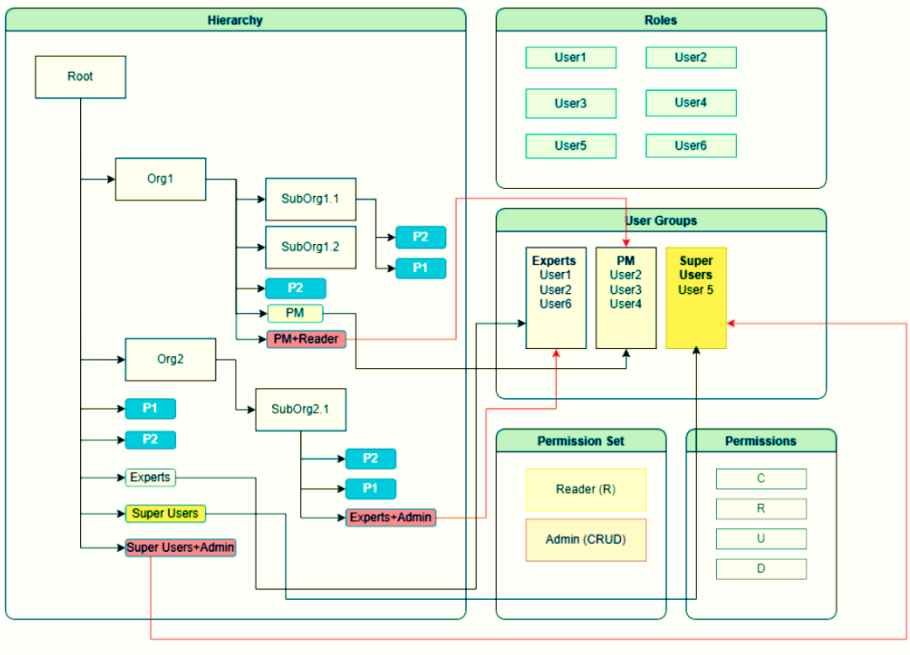

Fig. 2. Schema of Hierarchy and Permissions

Permission Checking (Binary Access Decision).Permission Checking is the core security function that determines whether a user can perform a specific action on a target node[9]. This mechanism enforces the principle of least privilege, ensuring that users are granted only the permissions explicitly allowed by their groups and the hierarchical structure.Evaluation Steps:

1. Subject Identification
   The system first identifies all groups associated with the acting user. This allows the architecture to map user roles to permission sets, providing the basis for further evaluation.
   Example: User5 → [Super Users]
2. Path Traversal and Aggregation
   The system then identifies the complete hierarchical path from the Root to the target node. At each level of the hierarchy, all relevant Permission Assignments (PAs) are collected and aggregated, allowing inheritance of permissions from higher-level nodes.
   Example: SubOrg1.1 path PAs → [PM+Reader@Org1, Super Users+Admin@Root]
3. Permission Resolution
   Aggregated permissions are expanded according to the user's group memberships. This step resolves potential conflicts and ensures that the user's effective permission set reflects both direct and inherited permissions.

Example: User5's [Super Users] group matches the PA at Root, granting Admin (CRUD) permissions

4. Final Verification
   The system checks whether the requested action is included in the user's resolved permission set. This produces the binary Allow/Deny outcome, which enforces the access control policy.
   Example: Update action → Allowed

The above steps highlight the systematic approach to evaluating user actions, ensuring security policies are applied consistently across complex hierarchical structures.

Use Case Examples.

The Table 2illustrates representative scenarios that demonstrate the effectiveness and granularity of the permission checking mechanism:

Table 2

Illustration of cenarios that demonstrate permission checking mechanism

Case	User	Action	Target Node	User Groups	Result	Rationale
1	User5	Update	SubOrg1.1	[Super Users]	Allowed	User5's group [Super Users] grants Admin (CRUD) at the Root level, which propagates down the hierarchy.
2	User4	Update	SubOrg2.1	[PM]	Denied	The [PM] group does not have Update permission on this node or along its path.
3	User4	Read	SubOrg2.1	[PM]	Allowed	The [PM] group has Reader permissions on the relevant path, allowing this action.
4	User2	Update	SubOrg2.1	[Experts]	Allowed	The [Experts] group has a PA granting Update (or Admin) on this specific path.
5	User2	Update	SubOrg1.2	[Experts]	Denied	Demonstrates granularity; the [Experts] group's permissions for SubOrg2.1 do not apply to SubOrg1.2.

Discussion

These use cases demonstrate several important aspects of the access management system[10, 11]:

- Inheritance and Propagation: Permissions assigned at higher levels (e.g., Root) are automatically propagated to child nodes, ensuring consistent policy enforcement.

- Granularity: Permissions can differ between sibling nodes, allowing fine-grained control over access.
- Conflict Resolution: When multiple groups assign different permissions, the system resolves conflicts in favor of the most permissive assignment for authorized actions, while maintaining overall security.
- Scalability: The hierarchical evaluation allows the system to handle large organizational structures efficiently, by evaluating only the relevant path for each user action.

Overall, Permission Checking ensures a robust, flexible, and transparent access control model, capable of supporting complex enterprise requirements while minimizing the risk of unauthorized actions.
Permission Visibility is crucial for user interfaces and auditing purposes, allowing the system to show users what actions they are allowed to perform at each level of the hierarchy, rather than simply returning a binary Allow/Deny[2, 12,13].
Evaluation Steps

1. Subject Identification
- Retrieve all groups associated with the user.
- Example: User2 → [Experts, PM]
2. Path Evaluation
- Evaluate the hierarchical path from Root to the target node, similar to Permission Checking.
- Example: SubOrg1.1 → [PM+Reader@Org1, Super Users+Admin@Root]
3. Node-Level Mapping
- Instead of producing a single Allow/Deny result, map the user's effective permissions at each specific node along the path.
- Example for User2:
    - Root: [Super Users] group does not match → NONE
    - Org1: [PM] group matches PM+Reader → R (Read)
    - SubOrg1.1: No specific PA → inherits R (Read)

Table 2

Illustration of permissions visibility

Case	User	Target Node	User Groups	Expected Visibility Output
1	User2	SubOrg1.1	[Experts, PM]	[{R@SubOrg1.1}, {R@Org1}, {NONE@Root}]
2	User2	SubOrg2.1	[Experts, PM]	[{CRUD@SubOrg2.1}, {NONE@Org2}, {NONE@Root}]
3	User5	SubOrg1.1	[Super Users]	[{CRUD@SubOrg1.1}, {CRUD@Org1}, {CRUD@Root}]

Discussion.

Permission Visibility Evaluation provides several benefits:

- Transparency: Users can see exactly which actions are available to them at each hierarchical level.
- Auditability: Facilitates internal audits by providing a clear map of permissions across organizational nodes.
- Consistency: Ensures that the UI reflects actual effective permissions, including inherited and group-based rights.
- Decision Support: Helps administrators identify permission gaps or misconfigurations in complex hierarchies.

By integrating Permission Checking and Permission Visibility, the proposed access management system achieves both operational security enforcement and user-friendly transparency, making it suitable for large-scale enterprise applications.

3. Conclusion and Future Work

The architecture successfully addresses the limitations of flat RBAC models in CRM/ERM systems by achieving object-level granularity and scalability through structuredassignments[10, 14, 15].

Architectural Summary: Advantages and Disadvantagesof described approach shown on Table 3.

Table 3

Advantages and Disadvantages of described approach

Category	Strengths	Limitations
Access Control	Granularity: Access is bound to specific Object/Node (O) instances, enabling precise object-level control.	Over-Privileging: Simple union aggregation lacks native support for explicit denials, potentially granting broader access than intended.
Management	Scalability: Reliance on Groups (G) minimizes administrative overhead and simplifies large-scale assignments.	Complexity: The multi-layered structure (PP, PS, PA, R, G) requires extensive administrator training and governance policies.
Security	Governance: The Privilege Escalation (PE) permission ensures administrative control remains auditable and restricted to authorized users.	Evaluation Performance: Access determination requires traversal of the hierarchical path, which may affect performance without optimized caching or indexing strategies.

Conclusion. This extended RBAC model provides a resilient, structured, and performant framework for managing sophisticated access control requirements in modern CRM and ERM systems. The design successfully balances the need for flexible permission aggregation with strict security governance, ensuring both operational efficiency and compliance with enterprise-grade security standards.

By integrating hierarchical Permission Assignments (PA), group-based inheritance, and dual evaluation mechanisms (Permission Checking and Permission Visibility), the model enables fine-grained, transparent, and dynamically scalable

access management[15]. Such an approach allows administrators to clearly define responsibilities, reduces redundancy in policy definitions, and improves maintainability in large, multi-level organizations.

The implementation also demonstrates high interpretability - users and system auditors can trace how each permission was derived, fostering trust and accountability[14]. Additionally, the path-based evaluation logic ensures that access decisions are computationally efficient, as only relevant segments of the hierarchy are processed during runtime.

Future work should focus on several enhancement directions. First, developing optimized indexing and caching strategies could further improve the performance of permission resolution in environments with millions of entities. Second, introducing explicit denial rules in a way that preserves scalability and avoids policy conflicts would make the model more expressive. Finally, integrating behavioral analytics and context-aware policies (e.g., time, location, or device-based constraints) could extend the system's adaptability to evolving enterprise security challenges.

In summary, the proposed model advances the traditional RBAC paradigm by offering a comprehensive, explainable, and efficient solution to hierarchical access control — one capable of meeting the demands of modern, distributed organizational systems.

References

[1] Mark G. Graff, Kenneth R. van Wyk, Secure Coding: Principles and Practices, O'Reilly Media, Inc., 2023. https://www.amazon.com/Secure-Coding-Principles-Mark-Graff/dp/0596002424

[2] Welcome to the OWASP Top 10 – 2021 OWASP 2022. [Online]. Available https://owasp.org/Top10/

[3] Paco Hope, Ben Walther, Web Security Testing Cookbook, O'Reilly Media, Inc., 2008. https://www.oreilly.com/library/view/web-security-testing/9780596514839/

[4] Secure coding guidelines, Microsoft 2021. [Online]. Available: https://learn.microsoft.com/en-us/dotnet/standard/security/secure-coding-guidelines

[5] Mark J. Price. C# 9 and .NET 5 – Modern Cross-Platform Development: Build intelligent apps, websites, and services with Blazor, ASP.NET Core, and Entity Framework Core using Visual Studio Code, 5th ed; Packt Publishing: 35 Livery Street Birmingham B3, 2PB, UK, 2020. https://www.amazon.com/NET-Cross-Platform-Development-intelligent-Framework/dp/180056810X

[6] Samuele Resca. Hands-On RESTful Web Services with ASP.NET Core 3 1st ed; Packt Publishing: 35 Livery Street Birmingham B3, 2PB, UK, 2019. https://www.amazon.com/Hands-RESTful-Services-ASP-NET-Core/dp/1789537614

[7] Secure development and deployment guidance, National Cyber Security Centre. [Online]. Available: https://www.ncsc.gov.uk/collection/developers-collection

[8] Adam Freeman. Pro ASP.NET Core 6: Develop Cloud-Ready Web Applications Using MVC, Blazor, and Razor Pages, 9th ed; Appres: London, UK, 2022. https://www.amazon.com/Pro-ASP-NET-Core-Cloud-Ready-Applications/dp/1484279565

[9] Cesar de la Torre, Bill Wagner, Mike Rousos, NET Microservices Architecture for Containerized .NET Applications, One Microsoft Way Redmond, Washington 98052-6399, 2022. https://learn.microsoft.com/en-us/dotnet/architecture/microservices/

[10] V. Samotyy, U. Dzelendzyak, N. Mashtaler,"A Comparative Study of Data Annotations and Fluent Validation in. NET", International Journal of Computing, Vol. 23, iss. 1,p.72–77, 2024, doi: 10.47839/ijc.23.1.3437.

[11] Suliman Alazmi; Daniel Conte De Leon,"A Systematic Literature Review on the Characteristics and

Effectiveness of Web Application Vulnerability Scanners", IEEE Access,Vol. 10, p.33200 - 33219, 2022, doi: 10.1109/ACCESS.2022.3161522

[12] Andreas Dann, Henrik Plate, Ben Hermann, Serena Elisa Ponta, Eric Bodden,"Identifying Challenges for OSS Vulnerability Scanners - A Study & Test Suite", IEEE Transactions on Software Engineering, Vol. 48, p.3613 - 3625, 2022, doi: 10.1109/TSE.2021.3101739.

[13] Ishan Siddiqui, Ankit Pandey, Saurabh Jain, Hetang Kothadia, Renuka Agrawal, Neha Chankhore,"Comprehensive Monitoring and Observability with Jenkins and Grafana: A Review of Integration Strategies, Best Practices, and Emerging Trends", Comprehensive Monitoring and Observability with Jenkins and Grafana: A Review of Integration Strategies, Best Practices, and Emerging Trends , Ankara, Turkiye, 26-28 October 2023, doi: 10.1109/ISMSIT58785.2023.10304904.

[14] Muhammad Usman, Simone Ferlin, Anna Brunstrom, Javid Taheri,"A Survey on Observability of Distributed Edge & Container-Based Microservices", IEEE Access,Vol. 10, p.86904 - 86919, 2022, doi: 10.1109/ACCESS.2022.3193102.

[15] U. Dzelendzyak, N. Mashtaler,"Comprehensive Approach To Protecting Data And The Information System Integrity",Measuring Equipment and Metrology, Volume 85, no.3: pp.47-53, 2024, doi: 10.23939/istcmtm2024.03.047.

Ієрархічна рольова модель керування доступом для CRM/ERM систем зі складними топологіями даних

Уляна Дзелендзяк, Назар Машталер

Ця робота пропонує вдосконалену архітектуру керування доступом, розроблену для систем управління взаєминами з клієнтами (CRM) та планування ресурсів підприємства (ERM), які працюють з даними у складних ієрархіях. Визнаючи обмеження традиційних пласких моделей RBAC у забезпеченні захисту на рівні об'єктів, архітектура вводить окремі сутності для груп (G), ієрархічних вузлів (O) та спеціалізований механізм призначення дозволів (PA). Це розширення забезпечує динамічне прийняття рішень щодо доступу на основі перетину ролі користувача, його належності до групи та положення цільового об'єкта в ієрархії. Модель характеризується високою масштабованістю та ґрунтується на агрегуванні дозволів через об'єднання. Ключові результати підкреслюють ефективний баланс між детальним контролем і адміністративною ефективністю, водночас визнаючи притаманну складність оцінювання та потребу в суворому управлінні за допомогою дозволу на підвищення привілеїв (PE).

Отримано 15.11.25