HiLL PUBLISHING

Advances in Computer and Communication, 2025, 6(4), 183-188
https://www.hillpublisher.com/journals/acc/
ISSN Online: 2767-2875

Research on the Implementation of RBAC–ABAC Hybrid Models in Large-scale Distributed Architectures

Yizhou Meng

DRIVE VRI COGS 1010, Microsoft, Redmond, WA 98052, USA.

How to cite this paper: Yizhou Meng. (2025) Research on the Implementation of RBAC–ABAC Hybrid Models in Large-scale Distributed Architectures. *Advances in Computer and Communication*, 6(4), 183-188.
DOI: 10.26855/acc.2025.10.005

Received: July 29, 2025
Accepted: August 27, 2025
Published: September 30, 2025

***Corresponding author:** Yizhou Meng, DRIVE VRI COGS 1010, Microsoft, Redmond, WA 98052, USA.

Abstract

This study investigates the implementation of a hybrid access control model that integrates RBAC (Role-Based Access Control) and ABAC (Attribute-Based Access Control) within large-scale distributed systems. The objective is to overcome the limitations of single models by combining the structural clarity of RBAC with the contextual flexibility of ABAC. Methodologically, the research analyzes the integration of role-permission mappings with attribute-based decision logic and develops a decision engine and policy execution path tailored for multi-node architectures. Multi-source attribute synchronization and a layered decision framework were designed to support distributed deployment. Experimental validation demonstrates that the hybrid model effectively handles cross-domain access requests and dynamic authorization processes, while ensuring consistency through policy snapshots and attribute subscriptions. The results confirm that the integrated model reduces response latency, improves policy accuracy, and enhances adaptability under high concurrency, thereby providing a robust and scalable foundation for secure access management in distributed environments.

Keywords

RBAC; ABAC; access control model; distributed systems; policy integration; authorization management

Introduction

With the rapid evolution of computing platforms toward cloud computing, edge computing, and the Internet of Things, system architectures are becoming increasingly distributed and heterogeneous. Access control now faces challenges such as highly variable user roles, dispersed request sources, and dynamic authorization demands. Traditional RBAC models, while structurally simple, cannot meet fine-grained and context-aware security requirements in large-scale environments. Sole reliance on ABAC, on the other hand, introduces high policy complexity and performance overhead. Against this backdrop, integrating the structural clarity of RBAC with the flexible decision-making of ABAC has emerged as a critical pathway for improving distributed access control, requiring systematic research into integration mechanisms and deployment strategies.

1. Logical Foundations and Design Principles of the RBAC–ABAC Hybrid Model

In addressing the complex access control requirements of large-scale distributed systems, standalone RBAC or ABAC models often force trade-offs between efficiency and flexibility. RBAC offers structural clarity, centralized authorization, and low maintenance cost, making it suitable for environments with clearly defined roles and responsibilities,

yet its static nature limits responsiveness to dynamic contexts and individual variability [1]. ABAC, by incorporating multidimensional attributes of users, resources, and environments, achieves fine-grained, context-aware authorization but increases policy design complexity and obscures decision pathways. The hybrid model integrates role constraints with attribute evaluation within the authorization process, reinforcing static role control with dynamic attribute filtering. Within this logic, policies may operate by first constraining access through role-based filtering, followed by attribute-driven refinement, or by combining roles and attributes in parallel to broaden coverage and precision. This design not only enhances adaptability to diverse access requests but also establishes the structural foundation for layered deployment, policy caching, and cross-node consistency management, thereby alleviating bottlenecks and risks in distributed access control.

2. Implementation Pathways in Large-scale Distributed Architectures

2.1 Access Control Characteristics in Distributed Systems

In large-scale distributed systems, access control mechanisms must accommodate node autonomy and network instability. The RBAC–ABAC hybrid model adopts the principles of "decision offloading, lightweight policies, and state synchronization" to avoid bottlenecks inherent in centralized authorization engines. The logic is decomposed into three layers: role filtering, attribute evaluation, and contextual mapping. Preliminary authorization is executed locally for fast response, while global consistency is maintained via policy snapshots and attribute subscriptions, enabling effective coordination under weak consistency conditions [2].

For example, in a multi-site Content Delivery Network (CDN) management system, the hybrid model's decision logic is encapsulated within edge service proxy modules. During initialization, the central policy controller distributes baseline role-permission tables and environmental attribute templates to local caches at each node, forming lightweight policy mirrors. When a user submits a request, the service proxy parses the access token to extract the associated role and checks local policies for baseline authorization. The attribute evaluation module then validates contextual attributes, such as request time, device type, source IP, and data sensitivity level, before returning the decision. All computation occurs locally, with central synchronization triggered only when policies are updated or contextual conflicts are detected, ensuring deployability and logical consistency under high-concurrency and highly distributed conditions.

2.2 Layered Deployment Mechanism of the Decision Engine

Deploying the RBAC–ABAC hybrid model in multi-node distributed environments requires an authorization engine architecture that balances autonomy with coordination. This is achieved through layered design, enabling localized responsiveness while maintaining global consistency. The decision engine is structured into three layers: the edge authorization layer handles rapid role-based filtering, the local decision layer performs attribute parsing and contextual evaluation, and the central coordination layer manages policy distribution and audit logging. This design ensures that most authorization operations are executed at the edge, reducing latency, while asynchronous synchronization with the center maintains consistency [3].

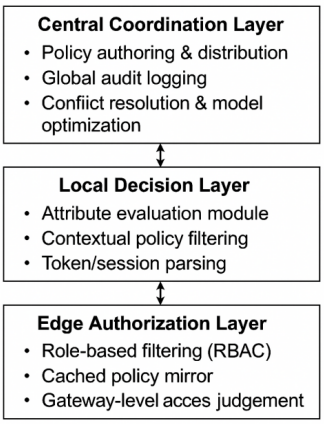

Figure 1. RBAC–ABAC Hybrid Model Access Control Engine Comprising a Three-Layer Architecture.

For instance, in a wide-area monitoring system, each edge gateway node integrates a local authorization proxy that receives role mappings and attribute templates from the central policy platform. When a user request arrives, the edge node first performs a role match against the local RBAC table, then activates the attribute evaluator to check multiple contextual parameters such as timestamp, device characteristics, and data labels. If the request matches, authorization is granted locally without invoking the central service. In cases of cache expiration or unresolved attributes, the local decision layer initiates a non-blocking callback to the central coordination layer for policy updates or log archival. As illustrated in Figure 1, the three-tier architecture ("edge–local–central") enables modular separation and message-driven interaction, ensuring clarity of control processes and portability of deployment across large-scale distributed systems.

2.3 Multi-source Attribute Management and Consistency Control Strategies

In the RBAC–ABAC hybrid model, attribute data serve as a critical foundation for access control decisions. These attributes typically originate from heterogeneous channels such as user identity directories, device status monitors, business context inputs, and environmental variables. Attribute management must ensure both real-time responsiveness and decision accuracy, while addressing challenges of distributed data, asynchronous updates, and inconsistent synchronization frequencies. The system architecture introduces attribute proxies and subscription mechanisms, abstracting external data sources into a unified attribute service interface [4]. Message queues and event triggers are then used to enable data-driven authorization actions. Attribute synchronization is maintained through version identifiers and state snapshots, thereby preventing decision errors caused by state drift or delayed writes.

For example, in a medical imaging retrieval system, an access request is evaluated not only on user roles and departmental affiliation but also on whether the device operates within the authenticated network, whether the request falls within an authorized duty period, and whether the imaging data are currently under a sensitivity constraint. The attribute service module collects data from LDAP directories, network gateways, scheduling systems, and imaging databases, injecting them into the local decision engine via a unified aggregation interface. Each attribute entry is bound with a timestamp and source identifier. When the central controller detects updates from the attribute source, it issues update notifications to edge nodes under a minimal-impact principle. Edge nodes determine—based on policy snapshots—whether to refresh their assertion templates, and newly updated policies take effect at the next request without disrupting active sessions. This mechanism enhances attribute validity while preserving authorization continuity and policy stability in asynchronous environments.

3. Key Implementation Techniques and Module Design

3.1 Policy Parsing and Dynamic Loading Mechanism

Within the RBAC–ABAC hybrid model, policy management must support unified expression of role constraints and attribute assertions, while enabling hot updates. Each policy can be abstracted into a triplet model:

$$P = \left\{ (r_i, A_j, C_k) \mid r_i \in R, A_j \in A, C_k : \varepsilon \to \{0,1\} \right\} \tag{1}$$

where R represents the role set, A denotes the attribute assertion space, and Ck is the contextual condition function. A policy is activated when Ck(e)=1, which then triggers the system to load or update the corresponding local authorization rule.

For instance, in a medical imaging retrieval system, the policy loading module employs an event-driven mechanism to listen for updates from heterogeneous attribute sources such as scheduling systems and network gateways. When a change in an attribute satisfies the condition function Ck, the parsing engine triggers structured policy loading, injecting the matched policy into the local execution table and updating the role-based rule chain [5]. Policy parsing follows a two-phase process: first, syntax parsing constructs an abstract syntax tree of attributes and condition nodes; second, state matching locates the minimal rule set affected within the policy snapshot. Internally, a dual-buffer mechanism is adopted: the old policy remains active to preserve session continuity, while the new policy is asynchronously validated and activated upon consistency checks. This design ensures that policy updates are highly controllable and non-disruptive, supporting fine-grained authorization across domains, heterogeneous sources, and multi-role environments.

3.2 Execution Logic of the Access Control Decision Module

The hybrid model adopts a four-stage decision pipeline: Identity Parser → Role Matcher → Attribute Evaluator → Policy Matcher. Each stage communicates through a shared execution context to form a complete authorization path from user identity to final decision. The core logic is governed by Boolean composition rules, expressed as:

$$C_k(e) = \bigwedge_{i=1}^{n} f_i(a_i) \ \ (f_i : assertion\ function, a_i : attribute\ value) \tag{2}$$

The process begins with parsing the access token to extract user ID and the associated role set Ru. The Role Matcher then retrieves the relevant policy subset Pr⊂P from the policy index. The Attribute Evaluator collects contextual values {a1, a2, ..., an} from local caches and event sources, executing each assertion function fi in sequence. A policy is activated only if Ck(e)=True; otherwise, access is denied.

In a wide-area monitoring system, the decision module is embedded at the edge control node. A request such as "remote device reboot" must satisfy multiple conditions: the request time treq∈T auth, the source IP IPsrc∈Rtrusted, and the target device state must be "idle." Upon successful evaluation, the authorization result is returned to the control gateway and simultaneously logged in the audit module. The system supports multi-threaded evaluation flows, short-term decision caching, and invalidation hooks, enabling fine-grained, low-latency authorization with strong contextual awareness.

3.3 Redundancy and Authorization Caching Mechanisms

In large-scale distributed architectures, the access control decision module must ensure high availability and stability. To achieve this, the system integrates redundancy and multi-level caching strategies. Redundancy covers both policy service redundancy and attribute source redundancy [6]. The former preserves continuous decision-making through primary–secondary policy replicas, while the latter ensures uninterrupted assertion logic via multiple attribute channels. Caching is divided into a Decision Cache, supporting fast request-time responses, and a Policy Mirror, responsible for versioned policy synchronization. The decision cache hit logic can be formalized as:

$$D(u,r) = \begin{cases} V_i, & if\ (u,r) \in C_t \wedge t - t_{u,r} \le \delta \\ \perp, & otherwise \end{cases} \tag{3}$$

Here, D(u,r) represents the cached authorization result for user u accessing resource r, Ctis the current cache set, tu,r is the entry timestamp, and δ denotes the time-to-live threshold.

In a multi-site CDN system, edge nodes first query the Decision Cache for the user–resource authorization state. If found and valid, access is granted immediately; otherwise, the system falls back to the full role–attribute evaluation chain, caching the new result. The central policy platform periodically distributes policy mirrors to CDN nodes, with version identifiers and invalidation signals to ensure consistency across sites. For critical attributes, such as "regional access legitimacy" or "service tier label," the system provisions primary and backup sources. If the primary identity provider fails to respond within timeout θ, the backup source is activated for assertion. Each cache entry is bound to a TTL and update hook, triggering revalidation upon state change to prevent authorization drift or outdated policies.

4. Application Verification and System Adaptation Effect Analysis

4.1 Case Deployment and Performance Validation

In a multi-site CDN system, to validate the deployment of the RBAC–ABAC hybrid model, access control engines were installed at three regional nodes worldwide, each configured with local Decision Caches and policy mirror synchronization modules. A test environment simulated 1,000 concurrent user requests, covering the complete pipeline of identity parsing, role matching, and attribute evaluation. Performance monitoring indicated that, with local caching and layered decision logic enabled, the average authorization response time at edge nodes stabilized below 40 ms, representing a 65% reduction compared to a purely centralized model. Moreover, policy mirror version control ensured consistent decision-making across nodes without conflicts. This deployment case demonstrates that distributed evaluation and caching mechanisms sustain real-time and stable access control under high concurrency conditions [7].

4.2 Efficiency and Accuracy Gains from Model Integration

In the validation process, a cross-node workload simulation was conducted using the multi-site CDN system to compare RBAC, ABAC, and the integrated hybrid model. Each edge node was equipped with a unified decision engine, incorporating local caches and attribute snapshots to accelerate request handling. The evaluation pipeline included concurrent request generation, real-time log tracing, and central aggregation of outcomes for consistency checks [8]. The integrated model applied hierarchical logic, where roles defined baseline scopes, attributes refined contextual conditions, and synchronized policy mirrors maintained node-level coherence.

As shown in Table 1, the hybrid model achieved significantly lower response latency compared to ABAC, with average authorization delays reduced to 38 ms. Cache hit rates improved to over 85%, while contextual accuracy reached 94%, effectively overcoming RBAC's limitations in fine-grained scenarios. Policy synchronization delays remained below 300 ms without conflicts, confirming notable gains in efficiency and precision.

Table 1. Performance Comparison of Different Access Control Models in CDN System

Model	Avg Response Time (ms)	Cache Hit Rate (%)	Policy Conflict Incidents	Accuracy in Contextual Authorization (%)
RBAC	95	60	0	72
ABAC	120	40	2	91
RBAC–ABAC	38	86	0	94

4.3 Adaptation Recommendations for Different Application Scenarios

In large-scale distributed systems, the adaptation of the RBAC–ABAC hybrid model must be tailored to specific application contexts [9]. For multi-site CDN services, where requests are highly concurrent and geographically dispersed, the focus should be on edge-level decision offloading and cache reuse. Regional role mapping tables and attribute snapshots are recommended to enable rapid authorization based on location and service tier. In wide-area monitoring systems, decision chains often depend on device states and operational contexts; here, the integration of attribute evaluators with event-driven triggers is critical, ensuring real-time metrics are incorporated into the decision logic without latency-induced failures. In IoT device access scenarios, where nodes are constrained in computing and storage capacity, lightweight policy mirrors and minimal attribute sets should be deployed to maintain portability and low overhead. Overall, effective adaptation requires flexible adjustment of decision hierarchy, attribute scope, and synchronization mechanisms, ensuring the hybrid model remains both scalable and context-sensitive across diverse distributed environments [10].

5. Conclusion

In large-scale distributed architectures, the complexity of access control requirements makes it difficult for a single model to balance efficiency and flexibility. By integrating role constraints with attribute assertions, the RBAC–ABAC hybrid model establishes a decision mechanism that is both stable and dynamically adaptive. This study examined its design, layered deployment, policy parsing, and redundancy strategies, with validation in multi-site CDN and wide-area monitoring systems. Results demonstrated that the hybrid model reduces response latency, improves cache utilization, and enhances contextual accuracy under high-concurrency and cross-node conditions. Looking ahead, integration with heterogeneous resource scheduling and intelligent policy optimization may further extend its applicability in IoT and cloud-native environments.

References

[1] Li B, Li W, Yang Y. Data-Driven Distributed Fault Detection and Fault-Tolerant Control for Large-Scale Systems: A Subspace Predictor-Assisted Integrated Design Scheme. IEEE Trans Cybern. 2025. [Epub ahead of print].

[2] Yin J, Ding J, Yang C. Distributed Estimation and Algorithm for Distributed Outcome Dependent Subsampling in Generalized Linear Regression with Large-scale Data. Stat Comput. 2025;35(5):156.

[3] Fan X, Xiang G, Liao W, et al. Decoding yeast transcriptional regulation via a data-and mechanism-driven distributed large-scale network model. Synth Syst Biotechnol. 2025;10(4):1140-9.

[4] Chen L, Qi N, Huo M, et al. Growing fast and steady in space: Distributed rapid self-reconfiguration motion planning optimization methods for swarm intelligent space modular self-reconfigurable satellites. Aerosp Sci Technol. 2025;166:110536.

[5] Shah P, Pahari S, Bhavsar R, et al. Hybrid modeling of first-principles and machine learning: A step-by-step tutorial review for practical implementation. Comput Chem Eng. 2025;194:108926.

[6] Siebeck SO, Hoving C. Characteristics of a Successful Nurse Peer Champion in the Implementation of Innovative Digital Technologies in Hospitals: A Qualitative Study. PEC Innov. 2024;5:100339.

[7] Roberts AN, Pelecanos A, Alexander K, et al. Implementation of Patient-Reported Outcomes in a Medical Oncology Setting (the iPROMOS Study): Type II Hybrid Implementation Study. J Med Internet Res. 2024;26:e55841.

[8] Quanbeck A, Chih YM, Park L, et al. Testing support models for implementing an evidence-based digital intervention for alcohol use disorder: results of a pragmatic hybrid implementation-effectiveness trial. Res Sq [Preprint]. 2024.

[9] Polestico LD, Bangcale LA, Velasco CL. Forecasting Implementation of Hybrid Time Series and Artificial Neural Network Models. Procedia Comput Sci. 2024;234:230-8.

[10] Stavrinides LG, Karatza DH. The impact of workload variability on the energy efficiency of large-scale heterogeneous distributed systems. Simul Model Pract Theory. 2018;89:135-43.