World Journal of Advanced Engineering Technology and Sciences
eISSN: 2582-8266
Cross Ref DOI: 10.30574/wjaets
Journal homepage: https://wjaets.com/

(REVIEW ARTICLE)

Check for updates

ROLE-BASED GOVERNANCE AND DYNAMIC ACCESS CONTROL ARCHITECTURE FOR SECURE ENTERPRISE SERVICENOW IMPLEMENTATIONS

Vinod Kumar Kolli *

Chicago State University.
* Corresponding Author

ORCID Details
Vinod Kumar Kolli: https://orcid.org/0009-0004-3000-6339

World Journal of Advanced Engineering Technology and Sciences, 2026, 21(02), 053–064

Publication history: Received on 01 September 2026; revised on 02 November 2026; accepted on 04 November 2026

Article DOI: https://doi.org/10.30574/wjaets.2026.21.2.0468

Copyright © 2026 Author(s) retain the copyright of this article. This article is published under the terms of the Creative Commons Attribution License 4.0

ABSTRACT

Highly sensitive enterprise data, workflows, and administrative processes are critical to the success of an Enterprise ServiceNow implementation and to how users access services. While conventional role-based access control forms the basis of access control, it may not be enough in complex situations with multiple users, changing roles, distributed resources, and changing contexts. This study aims to develop a governance-based Dynamic Access Control Architecture for deploying an enterprise ServiceNow instance secured in the North American Marketplace. The proposed architecture enables role-based governance and contextual attributes, policy-driven access authorization, least privilege, separation of duties, continual access review, and an adaptive access decision-making process. It helps provision access to information based on roles, enforce access in various ways, monitor access, revoke access, and evaluate access based on context. It also links governance principles to security mechanisms (such as roles, groups, security matrix, security attributes, and policy conditions) in ServiceNow. The paper brings these together into a cohesive conceptual framework, offering a stepping stone for tackling privilege accumulation, improving access transparency, and strengthening enterprise authorization governance. The proposed architecture also lays the groundwork for future empirical validation.

Keywords: ServiceNow; Role-Based Access Control; Dynamic Access Control; Access Governance; Zero Trust; Enterprise Security

1 INTRODUCTION

Enterprise service-management (ESM) platforms, once seen as tools for logging and acting on specific IT service requests, have evolved into digital platforms that support key organizational functions. Their use has expanded to include enterprise IT service management, IT operations, cybersecurity, asset and configuration Management, Human resources, Customer service, Software development, and more. Access to these platforms is critical to an organization's security posture, as they have become more interwoven with enterprise applications, databases, identity, APIs, and external services. An authorized user's access to ServiceNow tables, records, configuration items, administration, and

APIs can affect downstream processes and interdependent services [1-3]. For this reason, access control must consider, in addition to an individual's authenticity, whether they are also entitled to perform a specific action on a specific resource in a given situation. One shift to meet this requirement is the use of what has become known as Role-Based Access Control (RBAC). RBAC is built on the idea of constraining access to a collection of objects associated with a user's role, rather than basing access only on the user [4]. The RBAC model formalizes the conventional concepts of user, role, permission, role hierarchy, role assignment, role authorization, and constraint, and can therefore support structured authorization and administration in enterprise environments. The principle of least privilege emphasizes that users, applications, and processes should be granted only the access necessary to perform their required tasks, without unnecessary permissions to unrelated features or functions. However, as business environments have become increasingly dynamic and complex, traditional role-based approaches may not always provide enough flexibility to determine appropriate access permissions. In such scenarios, Attribute-Based Access Control (ABAC) can complement conventional access-control models by considering additional contextual attributes, including those associated with the subject, requested operation, object, and operating environment. By incorporating these factors, ABAC enables more granular and context-aware access-control decisions. In this way, authorization can consider not only which user is requesting access to the object, but also which object they are accessing, which operation is requested on the object, and the context of the request. In ServiceNow, you can restrict access to applications, tables, records, fields, and operations in different ways. ServiceNow Access Control Lists (ACLs) can evaluate conditions, security attributes, and scripted logic based on role requirements to determine whether the specified operation is permitted [5] [6]. In addition, security attributes can provide context and support authorization decisions [7]. Thus, in the world of governance, we are talking about identity, identity roles and privileges, contextual authorization, policy enforcement, monitoring, reviewing and revocation; all of these represent aspects of access management in ServiceNow. From this larger vision, we establish a conceptual framework for an RBAC model that can expand into a more dynamic, governance-based RBAC model that better fits complex ServiceNow deployments.

As enterprise ServiceNow instances become larger and more diverse, some pitfalls of static role assignment are exacerbated. If the organization is relatively stable, then a user can be assigned a role based on the job description and have relatively stable authorization. However, in reality, an enterprise user may have multiple functions, participate in temporary projects, belong to multiple organizational units, be assigned new roles, be granted administrative rights for a certain period, and communicate with resources of different sensitivities. This can make a business's correlation between role and access more complicated, depending on whether specific or refined access is required during that time. Stating permissions each day may increase an employee's privilege level; extensive use of special roles may increase administrative complexity and the number of roles employees use. Role hierarchies and the permissions they inherit can introduce considerable complexity, particularly when multiple roles span different applications, organizational unit structures, or both [1][2]. This complexity increases further when access-control decisions must incorporate contextual factors such as resource sensitivity, device status, geographic location, time, operational conditions, and the associated risk level. Consequently, a purely role-centric model may effectively answer, “Who is the user?”, but may not provide enough context to determine, “Should this specific request be granted at this particular moment?” ServiceNow provides several mechanisms to address these scoped, context-sensitive access requirements. More context-specific authorization decisions [5]-[7] can also be made with ACLs, which can be linked to role needs, data conditions, security attributes, or scripted logic. However, relying on a single access-control solution may not provide a comprehensive approach to managing access throughout its entire lifecycle. Therefore, it is important to distinguish between access enforcement and access governance. Access enforcement determines whether a specific operation should be permitted or denied based on established policies and conditions. In contrast, access governance encompasses the broader access lifecycle, including requesting access, providing justification, obtaining authorization, granting access, monitoring its use, conducting periodic reviews, modifying permissions when necessary, and ultimately revoking access when it is no longer required. This lifecycle perspective is even greater in Zero Trust security architecture, where the belief of “never trust, always verify” is put into practice; there is explicit policy and context with which authorization decisions are made, and no assumption of trust based on previous successful authentication or the network (or service) a user previously accessed [8, 9]. However, governance challenges can arise even when ACLs are perfectly in place, such as excessive privileges, stale role assignments, inadequate access reviews, ambiguous access-role ownership, overlapping access roles and responsibilities, and delays in revoking access rights. The primary challenge does not necessarily lie in the access-control mechanisms themselves, but in the lack of a comprehensive framework that integrates diverse access points through consistent governance, coordinated policy enforcement, and context-aware decision-making. This paper aims to fill that void by proposing an architecture that treats RBAC as the basic mechanism for enforcing organizational authorization, with added layers of control provided by contextual attributes, policy evaluation, lifecycle governance, monitoring, and adaptive decisions. This approach aims to manage static permissions and complex authorization in enterprise ServiceNow environments through continuous governance.

The dimensions explored in this conceptual paper are thus organized around five interwoven aspects:

- Role Governance: Models life cycle management of ServiceNow roles, permissions, approvals, assignments, reviews and revocation.

- Dynamic Authorization: Discusses the concept of property and policy requirements as well as roles for dynamic decision-making.
- ServiceNow Enforcement: Conceptually correlates the proposed governance architecture to certain ServiceNow roles, ACLs, security attributes, data conditions, etc., such as access control.
- Continuous Governance: Addresses access review, privilege escalation, separation of duties and changes in contexts throughout the lifecycle of access.
- Enterprise Security Architecture: Defines a coherent conceptual framework for federated identity and governance, policy analysis, policy enforcement, monitoring and adaptive access in enterprise ServiceNow applications.

2 RELATED WORK

One trend in enterprise access control research is moving away from access control models based on static permissions toward structured, attribute-based, policy-based, and context-aware access control models. Early RBAC work centered on roles as an abstraction of users that could enable empowerment and scaling across enterprises, rather than a one-to-one user-and-permission paradigm. The NIST RBAC model then amalgamated most RBAC concepts, including role hierarchy, constraints, role assignment, and role authorization, and paved the way for enterprise access management standardization [10]. RBAC, however, mainly reflects organizational responsibilities and cannot accurately model the dynamic conditions associated with an access request. For this reason, ABAC (Attribute-Based Access Control) was invented, and Authorization by attributes, associated with subjects, Objects, operations, and Environmental conditions, enables policies based on organizational information and environmental information. Since then, more access-control studies have incorporated RBAC, ABAC, and risk- and policy-based aspects to provide greater adaptability in granting access. At the same time, Zero Trust Architecture emerged as a paradigm that questions and challenges trust in the user, device, application, and resource instead of network-centric trust, where trust is embedded in the user, application, and device to enroll in the network and pass [11, 12]. These theoretical advances are relevant to the real world of ServiceNow, which provides ACL-based authorization based on a combination of role requirements, security attributes, data conditions, and even scripted logic [13]. Expanding the set of properties of a subject or its environment into security-attribute capabilities also adds information to a subject's fine-grained access decision [14]. Recent platform developments suggest an evolution from predominantly role-based access control toward more context-aware authorization approaches. Although existing literature and platform documentation discuss individual access-control mechanisms in detail, they often address them independently. Consequently, a more integrated governance framework is still needed to connect role lifecycle management, contextual authorization, policy evaluation, continuous access review, and access-control enforcement within ServiceNow. This indicates the need for a conceptual representation of access as a life cycle that needs to be actively managed. Our envisioned work builds on these elements and introduces ServiceNow access control within a broader governance environment using RBAC and ABAC, as well as the principle of least privilege or Zero Trust [15].

Table 1: Summary of Related Work

Ref.	Primary Focus	Key Contribution	Relevance to Proposed Work	Identified Limitation / Gap
[9]	Role-Based Access Control	Established roles as an abstraction between users and permissions	Provides the foundational authorization model	Primarily represents organizational roles rather than continuously changing contextual conditions
[10]	Standardized RBAC	Defines users, roles, permissions, hierarchies, constraints, and authorization relationships	Provides the formal basis for role governance and separation of duties	Does not by itself provide comprehensive contextual or risk-adaptive authorization
[11]	Attribute-Based Access Control	Evaluates subject, object, action, and environmental attributes against policies	Provides the conceptual foundation for dynamic and context-aware decisions	Attribute management and policy complexity can increase governance requirements
[12]	Continuous / Explicit Authorization	Shifts security from network perimeter assumptions toward users, devices, resources, and explicit policy decisions	Supports continuous and context-sensitive authorization principles	Provides an architectural security model rather than a ServiceNow-specific governance framework

[13]	Platform-Level Authorization	Controls access to tables, fields, records, APIs, and operations through roles, conditions, attributes, and scripts	Provides the primary enforcement layer for the proposed architecture	Individual ACL mechanisms do not constitute a complete enterprise access-governance lifecycle
[14]	Contextual Authorization	Enables fine-grained access decisions using information about subjects and their environment	Supports contextual extension of conventional role-based controls	Requires appropriate attribute definition, ownership, maintenance, and governance
[15]	Adaptive Enterprise Authorization	Connects identity, context, policy, and resource-level authorization	Establishes the conceptual basis for combining role governance with dynamic access decisions	Existing approaches do not sufficiently integrate these principles into a unified ServiceNow governance architecture

3 PROBLEM FORMULATION AND RESEARCH GAP

In Enterprise ServiceNow environments, access control (as opposed to typical user-to-user and user-to-application association) is a problem. Implementations grow across business functions, applications, organizational units and administrative domains, and the user's assigned role relative to the access needed at a specific time becomes more dynamic. Multiple roles may apply to a single user, permissions may be granted ad hoc for specific, short-term needs, or users may move to a different department, be given access to documents with varying sensitivities, or operate in another environment. Conventional RBACs have done well to specify the responsibility within the organization but suffer from several problems as well, such as the unchangeable perspective of the role assignment, which may result in privilege accumulation, role proliferation, stale permissions, etc., and lack of information regarding what use, if any, exists for permissions that have been granted in the past. Although ServiceNow ACLs and security attributes enable granular and conditional access control, they primarily regulate access to specific resources rather than addressing the complete access-control lifecycle, creating a need for an application-centric ServiceNow provisioning framework that can support complex enterprise environments by making flexible authorization decisions based on user attributes, resource sensitivity, operational context, and organizational policies while integrating identity and role governance, contextual evaluation, policy-based decision-making, policy enforcement, continuous access monitoring, and ongoing access assessment into a unified governance framework.

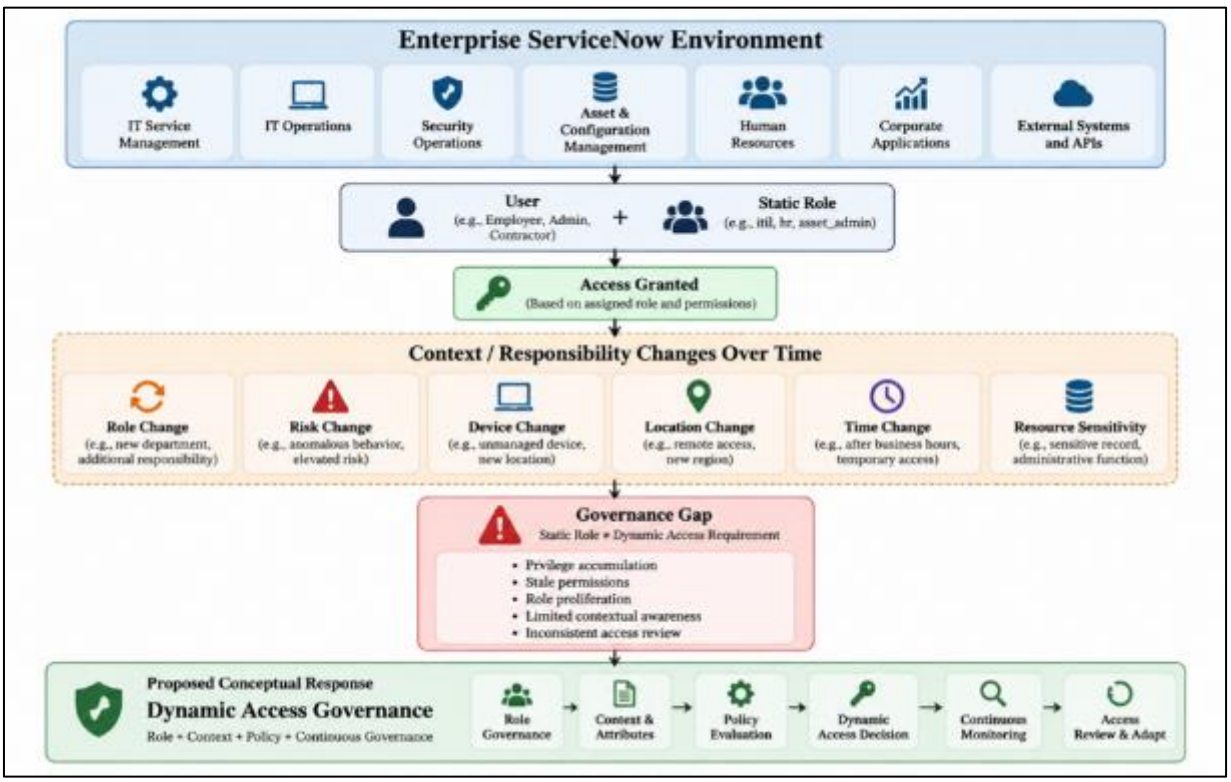


Figure 1: From Static Role Assignment to Dynamic Access Governance

Current approaches do not address the gap between access control enforcement and access governance, creating a research gap. These include structured access roles and permissions in RBAC, attribute-based, contextually driven authorization in ABAC, and the simple notion of explicit, continually reviewed authorization principles in Zero Trust [9]-[15]. Finally, ServiceNow provides platform-level control through roles, ACLs, security attributes and conditional authorization. But these are usually presented as individual security features rather than as part of a security architecture that spans the entire access lifecycle through governance. In this way, an enterprise can discover that a user is authorized or denied access to a specific resource, but without an integrated approach to understanding why this access is there, who the account holder is, and when the related privilege still makes sense due to changes in context, how this access should be influenced and when this access should be adjusted or denied. This creates a division between authorization based on known roles and identities (usually achieved through prior authorization) and identity-based, responsibility-, sensitivity-, resource-, context-, and policy-based authorization (dynamic authorization or access governance). The lack of RBAC, ACLs, or contextual security is not the problem; rather, the problem is the lack of a single model that provides a common governance model for all. This paper proposes a conceptual framework that extends traditional RBAC by incorporating contextual attributes, policy-driven decision-making, continuous monitoring, access certification, and privilege management throughout the access lifecycle. Accordingly, authorization is treated as a continuous and participatory process encompassing request, justification, decision, provisioning, evaluation, monitoring, review, modification, and revocation, rather than as a one-time activity focused solely on assigning a role to a user.

4 PROPOSED ARCHITECTURE

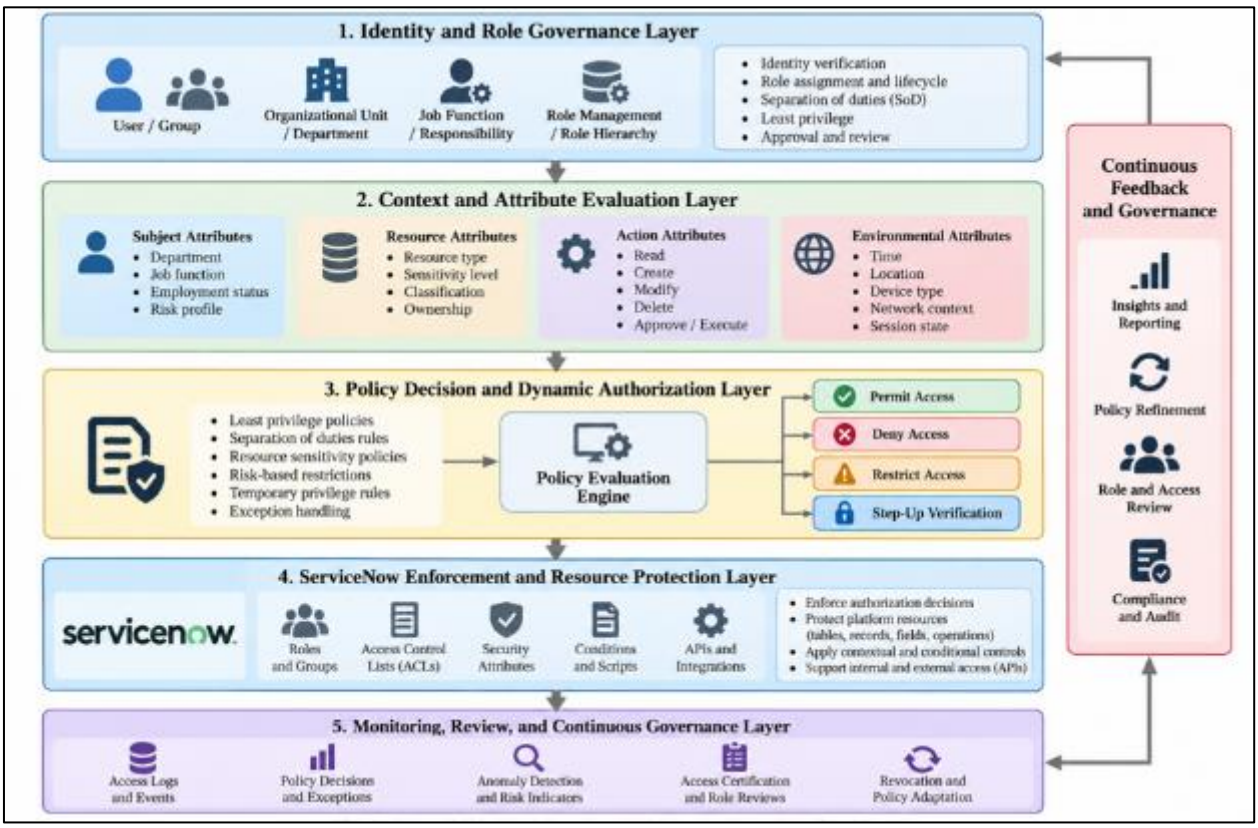


Figure 2: Proposed Role-Based Governance and Dynamic Access Control Architecture

The RG-DACA (Role-Based Governance and Dynamic Access Control Architecture) is an abstract concept that unifies traditional role-based access control and contextual access decision-making with continuous access governance in enterprise ServiceNow applications. RBAC remains a fundamental organizational abstraction, as roles provide a clear, familiar, and manageable way to structure business responsibilities and associated permissions. However, role membership alone may not provide enough information to make every access decision, particularly when additional contextual or risk-based factors must also be considered. Therefore, we propose adding complementary layers to the environment where access requests are assessed in increasing detail. The first layer establishes the user's identity and role context, including who the authenticated user is, their organizational affiliation, assigned roles, and the privileges associated with those roles. The second layer evaluates the access request context by considering factors such as the sensitivity of the requested resource, the operation being performed, device or session characteristics, location, time,

and other relevant conditions. Together, these layers enable access decisions to account for both the user's established permissions and the contextual requirements associated with a particular request.

The third tier is policy governance, the set of organizational rules that involve the principle of least privilege, separation of duties, temporary privilege, resource sensitivity, and risk-oriented restrictions. A dynamic decision layer then determines what to grant (or deny) and whether any restrictions or verification are necessary. In ServiceNow, authorization is determined by role, group, security attributes, conditions, and more. First, the architecture is not limited to enforcing authorization. Access activities are logged in an activity monitoring & governance system, with input for periodic access reviews or on an event basis. So, when users' responsibilities change, resource classification or migration changes, the risk context changes, or company policy changes, you can reassess granted access. Architecture thus changes access management from a static one of user → role → permission to an always-managed decision of identity → role → context → policy → decision → resource → monitoring → review. This structure creates a collaborative bridge between current RBAC paradigms and more sophisticated authorization mechanisms, and mirrors the nature of enterprise ServiceNow deployments.

4.1 Identity and Role Governance Layer

The identity and role governance layer lays the groundwork for the proposed architecture. It defines the entity requesting access and specifies the functional and organizational context of the called entity. This layer comprises users, user groups, organizational units, job functions, assigned roles, role hierarchies and identity attributes. Because roles provide a sensible abstraction between user must-haves and enterprise permissions, RBAC remains important. If you prefer not to establish permissions on individual users, you can establish a controllable role structure to represent organizational responsibilities. However, rather than maintaining a permanent authorization state on the servers as the architecture suggests, role assignment is a governed lifecycle. Each role should also have a stated business need, owners, permissions, and approvals or criteria for removal or retitling. The role-assignment lifecycle may encompass several stages, including access requests, justification, approval, provisioning, periodic review, modification, and eventual revocation. This structured approach helps minimize unnecessary privilege accumulation by keeping permissions closely aligned with users' responsibilities. The governance layer can also incorporate separation-of-duties (SoD) principles, identifying predefined combinations of roles or privileges that could create conflicts of interest and restricting their simultaneous assignment. Controlled assignments can also describe temporary responsibilities rather than privilege escalation. Thus, this layer can be used to specify the "authorization starting point" for the user, and subsequent layers determine acceptable and unacceptable uses of the user's authorization based on conditions.

4.2 Context and Attribute Evaluation Layer

The attribute evaluation layer builds on the role foundation and provides additional data that may affect the meaning of an access request. A role is typically an organizational responsibility, while contextual attributes provide additional information about whether the role is appropriate for a given resource, operation, session, or environment. The proposed architecture organizes access-related information into four primary attribute categories: subject, resource, action, and environment. Subject attributes capture relevant details about the requesting user or entity, including organizational affiliation, job responsibilities, employment status, functional role, and other identity-specific characteristics. Resource attributes can include the sensitivity, classification, ownership, or business criticality of a ServiceNow record, application, or service. Actions with attributes that represent the action being attempted, such as read, create, modify, delete, approve or invoke administrative action. Environmental attributes may encompass contextual factors such as time, geographic location, device characteristics, network conditions, and session information. Rather than replacing traditional role-based permissions, these attributes enhance them by adding context to access decisions. For instance, an administrator may possess a role that permits modifications to a sensitive configuration record, while contextual checks can further assess whether the request originates from an unusual location, device, network, or session. Similarly, if organizational responsibilities have changed, it may signal that permissions need reassessment. Enforcement points for these contextual considerations can be implemented through ServiceNow security attributes and conditional ACL mechanisms. Therefore, access can be decided based on the authority granted by the organization and the circumstances by associating this layer with the decision-making layer that grants a user access.

4.3 Policy Decision and Dynamic Authorization Layer

This section discusses the decision-making part of the proposed architecture, known as the Policy Decision and Dynamic Authorization Layer. This includes the user's baseline privileges, context, the resource's properties and characteristics, the requested operation, and organizational policies. This proposed model does not decide whether the requested access belongs to a specific role, but whether it conforms to applicable governance requirements. Policies may incorporate considerations such as separation of duties, resource ownership, resource sensitivity, temporary privileges, organizational responsibilities, identified risk scenarios, and approved exceptions. Based on these policy conditions, the authorization process can produce four conceptual outcomes: Permit, Deny, Restrict, and Step-Up. A Permit decision

grants access when the required authorization conditions are satisfied, whereas Deny prevents access when a request conflicts with established policies or lacks sufficient authorization. Restrict provides limited access by allowing only specified resources or operations, while Step-Up requires additional authentication, authorization, or approval before a sensitive or critical action can proceed. The policy layer can also distinguish between normal and exceptional operating conditions, enabling access decisions to adapt appropriately to changing contextual and risk factors. For example, if a user is authorized for an administrative role, they can perform routine access in routine circumstances, but they should have more controls when attempting access in an elevated-risk scenario with significant impact. However, governance policy is conceptually different from technical enforcement. ServiceNow mechanisms execute the policy decision, which reflects the organization's intent. This separation balances governance requirements, authorization decisions, and technical controls, and it fosters transparency around why access is granted, restricted, or denied.

4.4 ServiceNow Enforcement and Resource Protection Layer

The ServiceNow enforcement and resource protection layer implements authorization decisions and controls that can protect enterprise resources. This layer indicates the proposed concept and the corresponding ServiceNow security implementation layer. Mechanisms include Users, User Groups, User Roles, Access Control Lists, Security Attributes, Conditions and Scripted Authorization Logic. Access Control Lists (ACLs) enable granular authorization across specific platform resources, including tables, records, fields, and operations. Within the proposed architecture, ACLs function as Policy Enforcement Points (PEPs), implementing access decisions established by the higher governance and authorization layers. While application roles can provide a foundational level of authorization, ACLs can add conditions or rules to regulate access to sensitive resources, such as specific tables or fields. Where appropriate, Security Attributes can also be incorporated to provide additional context, enabling more precise, context-aware authorization decisions. Outside applications, automated processes can run in enterprise ServiceNow environments, and the same governance principles can apply to integrations and APIs. Access must therefore be uniformly controlled, whether through an interactive user session or a programmatic interface. The enforcement layer cannot decide the governance state of an entire access request on its own. It enforces decisions at the identity, role, context, and policy level, however. Conceptually, this separates governance logic, authorization decision-making, and platform enforcement, with native ServiceNow security capabilities as the real-world enforcement mechanism.

4.5 Monitoring, Review, and Continuous Governance Layer

The monitoring, review, and continuous-governance layer elevates the proposed architecture beyond an authorization mechanism to a lifecycle governance approach. Post-authentication changes may affect access, such as role changes, revised responsibilities, project terminations, changes in resources users can access, or reevaluation of access rights that may no longer be necessary. The monitoring layer therefore observes the correct invitation to action, the decision-making process for exceptions, role changes, and other governance activities. These monitors can determine whether unusual access behavior is occurring, whether exceptions repeat, whether privileges are used excessively or not used at all, or whether access shifts slightly and implies a reassessment. Based on these observations, the proposed framework supports both periodic and event-driven access reviews to ensure that permissions remain aligned with current organizational requirements. Such reviews may be initiated after events such as changes in service responsibilities, organizational restructuring, project completion, or the end of temporary assignments, at which point associated privileges should be reassessed and revoked when no longer needed. Similarly, repeated access to highly sensitive resources within a session may warrant additional governance review, even when each individual request is technically authorized. Accordingly, access governance is a continuous cycle of Monitor, Assess, Review, Modify, Revoke, and Re-evaluate, rather than a linear process that ends once permissions are granted. This lifecycle-oriented approach also strengthens accountability by establishing clear responsibility for access decisions, policy compliance, resource ownership, and ongoing access-management activities. The architecture does not assume an approved privilege is always appropriate; instead, it treats authorization/privilege as a state that must be justified throughout its lifetime.

5 COMPARATIVE ANALYSIS OF ACCESS CONTROL APPROACHES

The Dynamic Access Decision Model serves as a central component of the proposed architecture by integrating the user's established role with relevant contextual information, resource characteristics, the requested action, and applicable governance policies. This combined evaluation lets access decisions move beyond static role assignments and respond more effectively to the specific conditions of each request. The model addresses scenarios where a user is allowed to do what they want because of their role, but the surrounding environment needs further assessment. The decision process starts with a request from an authenticated user or service to access a ServiceNow resource. Identity and role information provides the user with default authority, and the model checks attributes related to the user, resource, requested operation, and environment. The model then analyzes the attributes against organization policies related to least privilege, separation of duties, resource sensitivity, temporary access, and risk conditions. The combined evaluation results in one of four conceptual outcomes: Permit, Deny, Restrict and Step-Up. A Permit decision means that

the operation was requested and that the role and context, along with the policy requirements, were met. Deny response is returned when the request does not meet or satisfy any policy, or is not sufficiently authorised. Restrict: Implement controlled access when only certain operations or resources are accessible. Step-Up adds another process to check or authorize a sensitive process within the system before it can take place. The model is designed to be dynamic, so the decision may change when context changes, even if the user's role hasn't. The system then applies the outcome to the appropriate ServiceNow security controls and documents it for monitoring and governance. This approach creates constant links between identity, role, context, policy, decision, enforcement, and monitoring, and allows consistent authorization to adapt to varying enterprise changes without losing the administrative model imposed by RBAC.

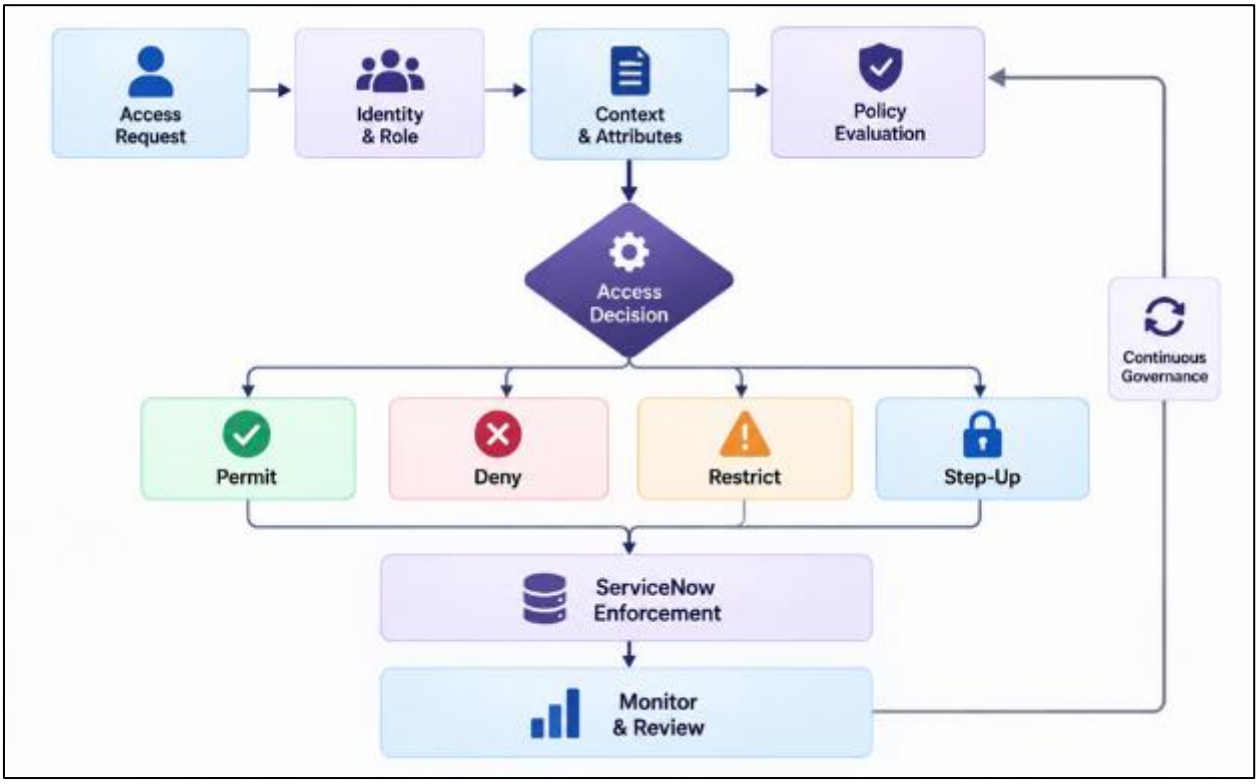


Figure 3: Dynamic Access Decision Model

6 EVALUATION DIMENSIONS AND RESEARCH OPPORTUNITIES

The Conceptual Evaluation Framework establishes dimensions through which the proposed architecture can be assessed using approaches such as prototype implementation, simulation, case studies, or enterprise deployment. One key dimension is security effectiveness, which evaluates the architecture's ability to minimize the likelihood of unauthorized, excessive, or inappropriate access while ensuring that legitimate business operations and required user activities remain uninterrupted. The second dimension, governance effectiveness, assesses whether clearly defined ownership, appropriate business justification, structured approval procedures, periodic reviews, and effective revocation mechanisms support access. Another important dimension is contextual adaptability, which evaluates how effectively access decisions can respond to changes in users, resources, operational conditions, and risk levels while continuing to incorporate established authorization factors such as roles and policy rules. Least-privilege effectiveness measures the ability to minimize unnecessary privileges and prevent privileges from building up over time. Policy consistency helps identify whether permission decisions can be executed consistently across different ServiceNow resources and organizations. Auditability and transparency focus on access decisions, and governance actions can be traced to relevant roles, attributes, policies, and approval processes. Scalability asks whether the model remains tractable as the number of users, roles, applications, resources, policies, and requests grows. Lastly, operational efficiency considers how well more stringent access governance works with the extra administrative effort or friction added by additional controls. Given these dimensions, the declarations would serve as a basis for further empirical validation, not as statements meant to appear as real achievements. The framework therefore sets out a potentially testable conceptual model of the proposed architecture, along with evaluation criteria that can reveal the practical effectiveness of future implementations.

Table 2: Conceptual Evaluation Framework for the Proposed Architecture

Evaluation Dimension	Primary Evaluation Focus	Conceptual Indicators	Expected Architectural Contribution
Security Effectiveness	Protection against inappropriate access	Unauthorized access, excessive privileges, policy violations	Strengthens resource-level access protection
Governance Effectiveness	Management of the access lifecycle	Ownership, justification, approval, review, revocation	Establishes accountable access governance
Contextual Adaptability	Response to changing conditions	User, device, location, time, resource, and risk changes	Enables context-aware authorization
Least-Privilege Effectiveness	Reduction of unnecessary privileges	Excess permissions, dormant access, privilege accumulation	Supports controlled and limited authorization
Policy Consistency	Consistent application of authorization rules	Policy compliance, decision consistency, exception handling	Reduces inconsistent access decisions
Auditability & Transparency	Traceability of access decisions	Access logs, policy decisions, approvals, reviews	Improves accountability and audit readiness
Scalability	Ability to operate across large environments	Users, roles, resources, policies, access requests	Supports enterprise-scale ServiceNow deployments
Operational Efficiency	Balance between security and usability	Administrative effort, review effort, user friction	Seeks stronger governance without unnecessary operational burden
Adaptability & Maintainability	Ability to evolve with enterprise requirements	Policy updates, role changes, architectural modifications	Supports long-term governance of changing environments

7 THREAT AND SECURITY ANALYSIS

The proposed architecture addresses security threats by adding multiple layers of governance and authorization, rather than relying on a static role-assignment service. One of the big ‘enemies' is when privileges for a user or an account are exceeded, which is known as privilege escalation. The identity and role governance layer addresses this risk through controlled role assignment and least-privilege principles, separation of duties, sign-off processes, and regular access audits. A similar risk is termed privilege creep, when users have their privileges slowly raised over time as their responsibilities evolve in the organization. Continuous governance helps identify roles that no longer make sense, and it can trigger revocation. Another major threat is credential misuse, where an unauthorised party uses legitimate credentials. In these cases, roles might still be relied on, and contextual assessment could add considerations for the session, device, location, requested resource, or operating state. project, or when a temporary project assignment is completed." When contextual conditions differ from expected patterns of use, the proposed architecture can respond by imposing additional access restrictions or requiring further verification before authorizing a request. This approach also helps mitigate insider misuse, where legitimate users may intentionally or unintentionally attempt to access resources beyond their authorized responsibilities. Context-aware policies, separation-of-duties controls, resource-sensitivity classifications, and continuous monitoring can further reduce the risk of excessive or inappropriate access, even when authorization is otherwise valid. Another significant governance concern is stale access, which can arise when users transition between projects or when temporary assignments conclude while previously granted permissions remain active. This is covered in the proposed 'Lifecycle' review, certification, modification and revocation. Incorrectly configured roles and ACLs remain exposures, as misconfigured security can inadvertently increase exposure. The split between governance, evaluation, enforcement and monitoring would provide an opportunity to identify/reassess such occurrences. So, the design is geared to manage threats through a mix of preventive measures, authorization in context, monitoring, and lifecycle governance, rather than relying on any one security tool or method.

Additional security considerations include resource-level compromise, excessive administrative privileges, unauthorized API access, and policy exceptions across interconnected ServiceNow environments. These risks may create entry points to enterprise resources, particularly when locations, applications, integrations, or privileged

administrative accounts are compromised or misconfigured. Therefore, the proposed architecture should account for these potential exposure points through appropriate access restrictions, contextual policy evaluation, monitoring, and governance controls. The proposed architecture provides sufficient techniques to address these concerns; for instance, contextual evaluation and resource-aware authorization (based on the sensitivity and operational value of the requested resource and the user's organizational role) can provide the necessary resolution. Increased approvals, role and responsibility segregation, temporary privileges, customizations, and monitoring are some ways to control administrative privileges. Similarly, API access should be authorized: API-based automated services and integrations can be an ongoing access point for enterprise resources. When users request unusual access or repeatedly ask for policy exceptions, are granted more privileges than they normally need, or show unexpected use patterns detected through monitoring, governance teams can investigate and re-evaluate access as needed. One is Policy conflict: when two or more roles, attributes, or authorization conditions (true or false) produce contradictory requirements. Thus, a governance-based strategy would enable clear, unambiguous policy ownership and prioritization, along with an audit trail of exceptions. The proposed access-control architecture does not directly address denial-of-service or availability threats, as its main goals are authorization and governance; however, access-control restrictions could reduce undesired admin actions that could affect a platform resource. Similarly, leaking privileged identities, misusing contextual information, or failing to set up enforcement mechanisms can affect the architecture. However, the security trade-off is that it increases the number of control points to evaluate, limit, monitor, and reconsider. The model's overall approach treats access control as defense-in-depth, where identity governance, contextual evaluation, policy enforcement, monitoring, and continuous review work together to reduce the likelihood and potential for inappropriate access to persist.

8 DISCUSSION

The proposed Role-Based Governance and Dynamic Access Control Architecture is a conceptual framework to enhance access governance in enterprise ServiceNow instances; however, it has a few drawbacks that warrant consideration. The challenge is that no testing of the design has been performed as part of any actual implementation, controlled testing or deployment into a production ServiceNow environment. For this reason, the relationships posited with role governance, contextual evaluation of an intervention, policy decisions, enforcement, and continuous review are architectural or assumed, rather than outcomes demonstrated in experiments. The model's effectiveness may also depend largely on the quality, accuracy, and availability of identity and contextual attributes. Therefore, organizations may lack sufficient information, have outdated role assignments, incomplete resource classifications, or even incorrect context information that could lead to incorrect authorizations. The other challenge lies with policies. When organizations implement many finer-grained rules for roles, resources, risks, temporary privileges, and exceptions, it can be difficult to keep policies consistent and present them in an easily understood way. Too much policy detail can also increase administrative burdens and create conflicting authorization requirements. Likewise, dynamic controls must be governed and aligned with the current landscape, especially when it includes legacy ACLs, inherited roles, custom apps, and third-party integrations. The proposed version also does not translate into numbers for authorization latency, effectiveness, user experience, or administrative workload. Moreover, the conceptual framework is unclear about an automatic algorithm for determining risk levels and alternatives for resolving conflicting policies. These restrictions imply that this picture must be understood here as a concept of a structure, or, even more, as a generic structure, rather than as an accepted blueprint of implementation. But it must be tested systematically to assess its real-world usefulness, scalability, and operational aspects, and to determine production-level performance.

Going forward, the research task force will validate and implement this proposed conceptual architecture and quantitatively evaluate it in a representative enterprise ServiceNow setting. The first research path is to develop a functioning prototype by implementing each suggested architectural layer as ServiceNow roles, groups, ACLs, security attributes, ACL-matching conditions, and monitoring/certs, and to examine access workflows. Enterprise scenarios such as policy exceptions, contextual changes, access to sensitive resources, temporary privilege assignments, and user role modifications provide practical conditions for evaluating the proposed architecture. A prototype can simulate these scenarios to assess how effectively the architecture adapts to changing access requirements while maintaining appropriate security, governance, and operational controls. The second line of effort is to systematically identify and manage excessive or contradictory privileges, including automated role analysis, privilege discovery, Role Mining Techniques, and the validity of separation of duties. Another potential future study is to develop risk-adaptive authorization mechanisms that can incorporate changing contextual information when making access decisions while still guaranteeing predictable, auditable policy behavior. Another direction is leveraging AI to navigate access governance, from identifying deviations from privileged access to proposing access changes via role recommendations, prioritizing access reviews, and revealing access privilege. However, when applying such approaches, there would be some ‘detailed' differences relating to explainability, governance and the quality of the data and human oversight. Comparable measurable indicators for comparative studies include unauthorised-access protection for users, reduction of users' privileges, efficiency in reviewing policies, consistency of policies, administrative workload, and latency in authorization relative to conventional RBAC and the proposed model. Longitudinal enterprise case studies could assess

how the architecture performs across an organization over time, including changes in application and organizational design, as well as evolving user and security needs. Looking ahead, future studies could establish a benchmark for how dynamic ServiceNow authentication can be conducted and evaluated to determine how it fits into a larger Zero Trust / IGA / enterprise security authentication model. These guidelines would make this a conceptual effort that would become a continually tested and verified model.

9 CONCLUSION

This conceptual paper describes an Architecture for Role-Based Governance and dynamic access control to secure ServiceNow enterprise implementations. In short, while RBAC is a strong first step in managing user permissions in an organization, many permissions must change based on the operational context, user roles, resource sensitivity, or changes in organisational policy. To address this limitation, this proposed architecture uses identity and role governance, contextual and attribute evaluation, policy- and governance-based authorization, ServiceNow-based enforcement and monitoring, and continuous access review, all under one conceptual model. Access is therefore treated as a continuous lifecycle rather than a one-time authorization event. This lifecycle encompasses role and access requests, business justification, approval, provisioning, ongoing access evaluation and monitoring, periodic review and adaptation of permissions, and eventual revocation when access is no longer required. Its dynamic decision model can be both restrictive and flexible, including additional verifications where applicable, as well as authorizations to deny or permit. The security analysis establishes a conceptual foundation for layered access controls and continuous governance, designed to address risks such as privilege escalation, privilege creep, compromised credentials, stale access, insider misuse, and authorization misconfiguration. As the framework remains conceptual, its effectiveness is not yet expressed through quantitative or empirically validated measures. Instead, it provides an architectural basis and a structured evaluation framework that can support future implementation, prototyping, experimentation, and empirical assessment. Automatic governance and adaptable authorization techniques, enterprise case studies, quantitative security assessments and prototypes of services like ServiceNow can support the model. To summarize, continuous, contextual and accountable access governance is a proactive evolution of role-based security to a more complex enterprise ServiceNow environment.

REFERENCES

[1] Sandhu, R. S., Coyne, E. J., Feinstein, H. L., & Youman, C. E. (1996). Role-based access control models. *Computer*, *29*(2), 38-47.

[2] Sandhu, R., Ferraiolo, D., & Kuhn, R. (2000, July). The NIST model for role-based access control: towards a unified standard. In *ACM workshop on Role-based access control* (Vol. 10, No. 344287.344301).

[3] Ferraiolo, D. F., Sandhu, R., Gavrila, S., Kuhn, D. R., & Chandramouli, R. (2001). Proposed NIST standard for role-based access control. *ACM Transactions on Information and System Security (TISSEC)*, *4*(3), 224-274.

[4] Kuhn, R., Coyne, E., & Weil, T. (2010). Adding attributes to role-based access control.

[5] Hu, V. C., Ferraiolo, D., Kuhn, R., Friedman, A. R., Lang, A. J., Cogdell, M. M., ... & Scarfone, K. (2013). Guide to attribute-based access control (ABAC) definition and considerations (draft). *NIST special publication*, *800*(162), 1-54.

[6] Force, J. T. (2020). *Security and privacy controls for information systems and organizations* (No. NIST Special Publication (SP) 800-53 Rev. 5 (Withdrawn)). National Institute of Standards and Technology.

[7] Grassi, P. A., Fenton, J. L., Newton, E. M., Perlner, R. A., Regenscheid, A. R., Burr, W. E., ... & Theofanos, M. F. (2016). Draft NIST Special Publication 800- 63 B: Digital Identity Guidelines. *National Institute of Standards and Technology (NIST)*, *27*, 103.

[8] Rose, S., Borchert, O., Mitchell, S., & Connelly, S. (2020). Zero trust architecture. *NIST special publication*, *800*(207), 1-52.

[9] Gupta, A., & Mittal, S. (2026). Cloud Security Architectures for Hybrid Enterprises. In *Foundations of Modern Information Security: A Vendor-Neutral Guide to Securing the Cloud, Data, and Identity in the AI Era* (pp. 135-174). Berkeley, CA: Apress.

[10] Dean, J. R. (2025). *Streamlining Access Control: An Attribute-Based Auto-Provisioning Study* (Master's thesis, Texas A&M University-Corpus Christi).

[11] Acosta, F. (2023). *Managing the ServiceNow Platform: A comprehensive guide to ServiceNow administration (English Edition)*. BPB Publications.

[12] Hauser, S. (2017). *Learning ServiceNow: Get started with ServiceNow administration and development to manage and automate your IT Service Management processes*. Packt Publishing Ltd.

[13] Ayub, K., & AlShawa, R. (2025, July). Zero-Trust Architectures for SecOps in ServiceNow: Enabling Scalable Security in Enterprise Platforms. In *2025 Multimedia University Engineering Conference (MECON)* (pp. 1-5). IEEE.

[14] Carter, E. (2026). Responsible Enterprise Modernization through Secure AI and Cloud-Native ServiceNow Automation. *American International Journal of Computer Science and Technology*, *8*(4), 36-49.

[15] Tsegaye, T., & Flowerday, S. (2020). A Clark-Wilson and ANSI role-based access control model. *Information & Computer Security*, *28*(3), 373-395.