Indian Journal of Computer Science and Technology
https://www.doi.org/10.59256/indjcst.20260503114
Volume 5, Issue3 (September-December 2026), PP: 959-964.
www.indjcst.com
ISSN No: 2583-5300

Multi-Context Access Control Framework

Bhabani Prasad Mishra[1], Rounakk Sabat[*2], Krushna Sahu[3], Ansuman Das[4], Subhranshu Mishra[5]

[1]*Professor, Department of Computer Science and Engineering, NIST University, Berhampur, Odisha, India.*
[*2,3,4,5] *Department of Computer Science and Engineering, NIST University, Berhampur, Odisha, India.*
[*] *Corresponding Author.*

To Cite this Article: *Bhabani Prasad Mishra[1], Rounakk Sabat[*2], Krushna Sahu[3], Ansuman Das[4], Subhranshu Mishra[5], "Multi-Context Access Control Framework", Indian Journal of Computer Science and Technology, Volume 05, Issue 03 (September-December 2026), PP: 959-964.*

Copyright: ©2026 This is an open access journal, and articles are distributed under the terms of the Creative Commons Attribution License; Which Permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited.

Abstract: *Traditional Role-Based Access Control (RBAC) provides a structured mechanism for managing permissions but does not independently account for the changing circumstances surrounding an access request. Although Attribute-Based Access Control (ABAC) and existing context-aware authorization models support contextual attributes, challenges remain in handling unreliable context information, distinguishing environmental context from user and access-behavior context, and producing deterministic and explainable authorization decisions. This paper presents the Multi-Context Access Control (MCAC) framework, a context-aware authorization model that separates environmental context, including time, location, network and device information, from user and access-behavior context, including session assurance, request frequency, access history and navigation patterns. Each context item is assigned an explicit validity status: valid, stale, missing, unverifiable or contradictory. The framework evaluates baseline role permissions before contextual conditions, ensuring that contextual information can restrict but never expand the permissions granted by the baseline authorization model. It subsequently evaluates contextual conditions using three-valued logic and sensitivity-dependent severity aggregation to produce one of three outcomes: ALLOW, VERIFY or DENY. The framework also incorporates versioned policy management, decision traceability and replayable authorization records. A TypeScript-based domain engine and supporting policy and trace-persistence components have been developed and subjected to automated testing. A reproducible synthetic-scenario evaluation protocol is specified to assess the framework against RBAC and alternative context-aware configurations. The complete benchmark evaluation has not yet been performed; therefore, no empirical performance improvement is claimed.*

Key Word: *Access Control; Context-Aware Authorization; Multi-Context Access Control; Role-Based Access Control; Attribute-Based Access Control; Environmental Context; Behavioral Context; Policy Evaluation; Three-Valued Logic; Authorization Traceability.*

I. INTRODUCTION

Access control is a fundamental component of information security that determines which users or entities are permitted to perform specific actions on protected resources. Role-Based Access Control (RBAC) is widely adopted because it simplifies permission management by associating permissions with organizational roles rather than assigning them individually to every user. However, conventional RBAC primarily evaluates static role assignments and permissions and does not independently consider the circumstances under which an access request is generated.

Modern applications operate in dynamic environments where authorization decisions may depend on several contextual factors. These factors include the time of access, network environment, device identity, session assurance, request frequency and historical access behavior. A request made by an authorized user from a registered device during an expected working period may represent a legitimate operation, whereas a request using the same credentials from an unfamiliar device or exhibiting unusual access behavior may require additional verification.

Attribute-Based Access Control (ABAC) and existing context-aware authorization approaches already support the use of contextual attributes. Therefore, the use of context alone is not considered the primary contribution of this work. Instead, this research focuses on the structured separation of environmental context and user/access-behavior context, explicit representation of context validity, deterministic evaluation of authorization conditions and traceable decision generation.

The proposed Multi-Context Access Control (MCAC) framework organizes contextual information into two categories: environmental context (Ce) and user/access-behavior context (Cb). Environmental context includes time, coarse location, network and device information. User/access-behavior context includes session assurance, request frequency, access history, navigation patterns and requested actions. Every context item is associated with a validity state to represent whether its evidence is reliable and suitable for authorization evaluation.

The framework follows a baseline-first authorization strategy. It first evaluates the permissions granted by the existing role-based authorization model. Contextual conditions are subsequently evaluated to restrict access according to the validity and sensitivity of the requested resource. The framework uses three-valued condition evaluation and deterministic

severity aggregation to produce ALLOW, VERIFY or DENY decisions.

The primary objective of this research is to develop and formally describe a deterministic, explainable and reproducible multi-context authorization framework. The work also specifies a synthetic evaluation methodology for examining decision quality, context degradation handling, traceability and computational overhead against appropriate baseline configurations.

II. MATERIAL AND METHODS

Study Design: This research follows a design-oriented software engineering methodology involving formal system modelling, framework development, implementation and planned experimental evaluation.

The proposed MCAC framework is formally specified through contextual models, context-validity states, authorization conditions, severity mappings and deterministic decision rules. A TypeScript-based domain engine has been developed to implement the principal components of the framework. Supporting components include policy-version persistence, decision-trace persistence and an independent reference evaluator for selected conformance checks.

The research distinguishes implementation validation from experimental evaluation. Automated tests are used to examine the correctness of implemented components, whereas a separate synthetic-scenario benchmark is specified to evaluate authorization behavior against alternative configurations. Completion of the implementation does not imply completion of the benchmark.

Study Location: The research and prototype development were carried out under the Department of Computer Science and Engineering at NIST University, Berhampur, Odisha, India.

Study Duration: The study was conducted as part of the academic minor-project research and software development activities from 17th May to 12th November.

Sample size: The planned test configuration consists of 250 instances per scenario family per seed and 30 independent test seeds. Calibration seeds and test seeds are kept separate to reduce the possibility of tuning the framework against the evaluation data.

Sample size calculation: The sample size was estimated on the basis of a single proportion design. The target population from which we randomly selected our sample was considered 20,000. We assumed that the confidence interval of 10% and confidence level of 95%. The sample size actually obtained for this study was 96 patients for each group. We planned to include 300 patients (Group I- Control, Group II- Cases of 100 patients for each group) with 4% drop out rate.

Subjects & selection method: The experimental subjects are synthetic authorization entities generated by a deterministic scenario-generation process. These entities include users, roles, protected resources, device registrations, network classes, working-hour profiles and historical access records.

Each generated authorization request contains the information required for policy evaluation, including the requesting user, requested resource, action, timestamp, environmental context and user/access-behavior context.

Every generated instance receives an independent intent label identifying it as benign or violating. These labels are assigned by the scenario-generation specification before authorization decisions are evaluated and are not derived from the output of the MCAC framework.

No real user credentials, personal information or institutional access logs are used in the proposed synthetic evaluation.

Inclusion criteria:

1. Authorization requests containing a defined user, resource and requested action.
2. Requests with an associated baseline role-permission configuration.
3. Requests containing environmental context, user/access-behavior context or explicitly degraded context evidence.
4. Requests assigned an independent benign or violating intent label.
5. Requests generated using a recorded scenario-generator version, configuration and random seed.
6. Requests belonging to one of the twelve defined scenario families.

Exclusion criteria:

1. Malformed authorization requests that cannot be interpreted by the evaluation harness.
2. Instances without a valid scenario-generation configuration or reproducible seed.

Instances with missing or invalid independent intent labels.

3. Duplicate instances introduced by accidental data-processing or execution errors.
4. Test instances generated using calibration seeds when evaluating the frozen test configuration.
5. Results produced by an evaluator configuration that has not passed the required implementation validation.

Procedure methodology

The MCAC framework is organized around two principal categories of contextual information.

Environmental Context (Ce)

Environmental context represents the external circumstances surrounding an authorization request. It includes:

- Time: The timestamp and configured access window associated with a request.
- Location: Coarse location information associated with the requesting environment.
- Network: The network classification from which the request originates.
- Device: Device registration and associated device evidence.

2. User and Access-Behavior Context (Cb)

User/access-behavior context represents the characteristics of the authenticated session and the user's access patterns. It includes:

- Session assurance: Information describing the assurance level of the current session.
- Request frequency: The frequency of access requests within a defined interval.
- Access history: Historical resource-access information.
- Navigation: The sequence or pattern of resource access.
- Requested action: The operation requested against a protected resource.

3. Context Validity Classification

Every context item is associated with one of five validity states:

- V (Valid): Context evidence is available and satisfies the configured validity requirements.
- S (Stale): Context evidence exists but is older than the permitted freshness interval.
- M (Missing): Required context evidence is unavailable.
- U (Unverifiable): The integrity or reliability of the evidence cannot be established.
- X (Contradictory): Available evidence establishes a contradiction with another relevant context item or declared consistency constraint.

This classification allows the framework to distinguish a genuine contextual violation from unavailable or unreliable evidence.

4. Baseline Authorization Evaluation

The framework evaluates baseline role permissions before applying contextual conditions. If the requested resource or action is not permitted by the baseline authorization model, the request is denied.

Contextual evaluation is performed only after the baseline permission check. Consequently, the framework follows a restriction-only model in which contextual information can restrict existing permissions but cannot grant permissions that are absent from the baseline authorization configuration.

5. Contextual Condition Evaluation

The framework evaluates applicable policy conditions against the available context items. Conditions are evaluated using three-valued outcomes:

- TRUE: The condition is satisfied.
- FALSE: The condition is not satisfied.
- INDETERMINATE: The condition cannot be reliably evaluated.

The resulting outcomes are mapped to severity levels according to the condition strength and resource sensitivity. This enables the framework to apply different authorization consequences to soft and hard conditions and to resources with different sensitivity levels.

6. Context Interaction and Decision Generation

The framework specifies two contextual interaction mechanisms. The first is network-conditioned threshold selection, which selects an appropriate threshold based on the classified network environment. The second is cross-category consistency evaluation, which examines relationships between relevant context items.

After contextual evaluation, condition severities are aggregated independently for environmental and user/access-behavior context.

The resulting severity values are combined to produce one of three authorization outcomes:

- ALLOW: The request satisfies the applicable authorization requirements.
- VERIFY: Additional verification is required before access is granted.
- DENY: The request is rejected.

7. Policy and Decision Traceability

The framework incorporates versioned policy information and structured decision traces. A decision trace records relevant authorization inputs, policy-version information, condition outcomes, context-validity states, severity values and the resulting authorization decision.

The audit design uses SHA-256 hash chaining to link records and support integrity verification. Verification of the latest chain head requires an independently retained or externally anchored reference.

8. Prototype Implementation and Validation

The framework is implemented using TypeScript within a software application environment. The domain engine, policy-version persistence and trace-persistence components are evaluated through automated tests. An independent reference evaluator is also specified to check selected authorization cases against the formal decision model.

Implementation validation and live production authorization integration are treated as separate activities. The current prototype does not establish that the formal MCAC engine has replaced the application's existing live authorization workflow.

Statistical analysis

The proposed experimental evaluation uses quantitative metrics to assess authorization decision quality and computational overhead.

The violating request is treated as the positive class. The planned metrics include:

1. False Acceptance Rate (FAR): The proportion of violating requests that receive a direct ALLOW decision.
2. False Rejection Rate (FRR): The proportion of benign requests that receive a DENY decision.
3. Challenge Rate (CR): The proportion of benign requests that receive a VERIFY decision.
4. Precision: The proportion of requests classified as non-allowed that are actually violating.
5. Recall: The proportion of violating requests that do not receive a direct ALLOW decision.
6. F1-score: The harmonic mean of precision and recall.
7. Context Coverage: The proportion of applicable context evidence that is valid.
8. Decision Latency: The time required to generate an authorization decision.
9. Throughput: The sustained authorization request rate under the specified latency constraint.
10. Resource Overhead: CPU and memory consumption during evaluation.

The planned analysis computes metrics independently for each test seed and summarizes them across the test seeds. Mean values and 95% percentile bootstrap confidence intervals are planned, using 10,000 bootstrap resamples over seeds. Paired comparisons between evaluation configurations are planned using the same test seeds, with Wilcoxon signed-rank tests and Holm correction for multiple comparisons.

Latency measurements are planned to include the 50th, 95th and 99th percentiles. Statistical results will only be reported after execution of the benchmark. No statistical significance or performance improvement is claimed in the present implementation report.

III. RESULT

The development of the Multi-Context Access Control framework resulted in a formal authorization model and a TypeScript-based domain-engine implementation. The framework defines two contextual categories, environmental context and user/access-behavior context, and assigns explicit validity states to individual context items.

The authorization model establishes a baseline-first evaluation strategy in which contextual conditions can restrict but cannot expand the permissions granted by the baseline role-permission model. It also specifies three-valued condition evaluation, sensitivity-dependent severity aggregation and three authorization outcomes: ALLOW, VERIFY and DENY.

The implementation includes the principal domain-engine components, policy-version persistence and decision-trace persistence. An independent reference evaluator has also been developed for selected differential checks. Automated tests provide implementation-level evidence for the components covered by those tests.

The implementation audit identifies remaining formalization and implementation-to-specification conformance requirements. Production context providers are not fully integrated or verified, and the formal MCAC engine has not replaced the application's existing live authorization workflow.

A reproducible experimental evaluation protocol has been defined with twelve synthetic scenario families and six baseline or ablation configurations. However, the complete benchmark evaluation has not been executed. Therefore, false acceptance rates, false rejection rates, challenge rates, latency measurements and comparative performance results are not reported as experimental findings.

The present results demonstrate the development of the formal model, prototype components and evaluation methodology. They do not establish measured improvements in authorization accuracy, security, usability or performance.

IV. DISCUSSION

The proposed MCAC framework addresses the limitations of relying exclusively on static role-permission evaluation when authorization decisions are influenced by changing environmental and behavioral circumstances. By separating environmental context from user/access-behavior context, the framework provides a structured method for evaluating different categories of evidence and identifying their respective contributions to an authorization decision.

An important aspect of the framework is its explicit treatment of context validity. Conventional condition evaluation may produce unreliable outcomes when contextual evidence is missing, outdated or unverifiable. MCAC represents these conditions through separate validity states and uses three-valued evaluation to preserve uncertainty rather than automatically treating unavailable evidence as either valid or invalid.

The baseline-first authorization strategy also provides an important security property. Contextual conditions are evaluated only after the baseline role-permission check, preventing contextual information from granting permissions that are not already authorized. This establishes the restriction-only behavior of the proposed model.

The use of sensitivity-dependent severity aggregation allows the framework to distinguish between resources requiring different levels of protection. A condition that results in additional verification for a low-sensitivity resource may lead to denial for a high-sensitivity resource. This approach provides a deterministic mechanism for expressing security requirements without relying on a single weighted risk score.

Decision traceability and policy versioning further support inspection and reproducibility. By associating authorization decisions with policy versions and structured traces, the framework is designed to provide explanations of how individual conditions and context-validity states contributed to the final outcome. Hash-chained audit records provide an additional mechanism for detecting unauthorized modification when verification is performed against a trusted chain head.

Nevertheless, the current implementation and evaluation status impose important limitations. Automated testing provides evidence of implementation behavior but cannot establish real-world authorization accuracy or security improvement. Similarly, the proposed synthetic evaluation can provide controlled and reproducible comparisons but cannot fully represent the diversity of real users, institutional policies and operational environments.

The framework also depends on the reliability of contextual evidence and the correct configuration of policy rules. A compromised server, stolen device credentials, misleading context signals or incorrectly configured policies may affect authorization outcomes. Behavioral deviations do not necessarily indicate malicious intent, and legitimate changes in user circumstances may produce additional verification requirements.

The full benchmark has not yet been completed, and the framework has not been integrated into the live authorization workflow. Therefore, the practical effectiveness of MCAC must be established through further implementation validation, controlled experimental evaluation and, eventually, carefully governed real-world deployment.

V. CONCLUSION

This research presented the Multi-Context Access Control framework, a context-aware authorization model that separates environmental context from user and access-behavior context and explicitly represents the validity of individual context items.

The proposed framework evaluates baseline role permissions before contextual conditions, ensuring that context can restrict but never expand the authorized permission set. It employs three-valued condition evaluation, sensitivity-dependent severity aggregation and deterministic authorization outcomes in the form of ALLOW, VERIFY and DENY.

The work includes a formal system model, a TypeScript-based domain-engine implementation, policy-version persistence, decision-trace persistence and an independent reference evaluator for selected conformance checks. It also specifies a reproducible synthetic-scenario evaluation methodology for comparing the framework against RBAC and alternative context-aware configurations.

Although the implementation and automated testing provide initial evidence of technical feasibility, the complete experimental benchmark has not yet been executed. Consequently, no empirical improvement in authorization accuracy, security, usability or computational performance is claimed.

Future work will focus on resolving the remaining formal specification requirements, completing the experimental harness, validating implementation conformance, executing the planned benchmark and integrating the framework with production authorization and trusted context providers. The practical value of the framework will ultimately be determined by reproducible experimental findings rather than by the proposed design alone.

REFERENCES

1. R. S. Sandhu, E. J. Coyne, H. L. Feinstein, and C. E. Youman, “Role-based access control models,” Computer, vol. 29, no. 2, pp. 38–47, 1996, doi: 10.1109/2.485845.
2. D. F. Ferraiolo, R. Sandhu, S. Gavrila, D. R. Kuhn, and R. Chandramouli, “Proposed NIST standard for role-based access control,” ACM Transactions on Information and System Security, vol. 4, no. 3, pp. 224–274, 2001, doi: 10.1145/501978.501980.
3. V. C. Hu et al., “Guide to attribute based access control (ABAC) definition and considerations,” NIST, Gaithersburg, MD, USA, Special Publication 800-162, 2014, doi: 10.6028/NIST.SP.800-162.
4. V. C. Hu, D. R. Kuhn, D. F. Ferraiolo, and J. Voas, “Attribute-based access control,” Computer, vol. 48, no. 2, pp. 85–88, 2015, doi: 10.1109/MC.2015.33.
5. S. Lachmund, T. Walter, L. Gomez, L. Bussard, and E. Olk, “Context-aware access control: Making access control decisions based on context information,” in Proceedings of the 1st International ICST Workshop on Ubiquitous Access Control (IWUAC), IEEE, 2007, doi: 10.1109/MOBIQW.2006.361782.
6. A. S. M. Kayes, J. Han, W. Rahayu, T. Dillon, M. S. Islam, and A. Colman, “A policy model and framework for context-aware access control to information resources,” The Computer Journal, vol. 62, no. 5, pp. 670–705, 2019, doi: 10.1093/comjnl/bxy065.
7. A. S. M. Kayes, W. Rahayu, T. Dillon, E. Chang, and J. Han, “Context-aware access control with imprecise context characterization for cloud-based data resources,” Future Generation Computer Systems, vol. 93, pp. 237–255, 2019, doi: 10.1016/j.future.2018.10.036.
8. R. R. Sabat, “Context-Aware Access Control System: A Risk-Adaptive Authorization Framework,” TANZ Research Journal, vol. 12, no. 4, Apr. 2026.
9. J. H. Saltzer and M. D. Schroeder, “The protection of information in computer systems,” Proceedings of the IEEE, vol. 63, no. 9, pp. 1278–1308, 1975.
10. X. Jin, R. Krishnan, and R. Sandhu, “A unified attribute-based access control model covering DAC, MAC and RBAC,” in Data and Applications Security and Privacy XXVI (DBSec), Lecture Notes in Computer Science, vol. 7371, 2012, pp. 41–55.
11. OASIS, “eXtensible Access Control Markup Language (XACML) version 3.0,” OASIS Standard, 2013. [Online]. Available: http://docs.oasis-open.org/xacml/ 3.0/xacml-3.0-core-spec-os-en.html
12. A. K. Dey, “Understanding and using context,” Personal and Ubiquitous Computing, vol. 5, no. 1, pp. 4–7, 2001.
13. E. Bertino, P. A. Bonatti, and E. Ferrari, “TRBAC: A temporal role-based access control model,” ACM Transactions on Information and System Security, vol. 4, no. 3, pp. 191–233, 2001.

14. M. J. Covington, W. Long, S. Srinivasan, A. K. Dev, M. Ahamad, and G. D. Abowd, "Securing context-aware applications using environment roles," in Proceedings of the 6th ACM Symposium on Access Control Models and Technologies (SACMAT), 2001, pp. 10–20.
15. J. Park and R. Sandhu, "The UCONABC usage control model," ACM Transactions on Information and System Security, vol. 7, no. 1, pp. 128–174, 2004, doi: 10.1145/984334.984339.
16. R. W. McGraw, "Risk-adaptable access control (RAdAC)," in Proceedings of the NIST Privilege Management Workshop, NIST-PRIVILEGE-09, Gaithersburg, MD, USA, 2009.
17. P.-C. Cheng, P. Rohatgi, C. Keser, P. A. Karger, G. M. Wagner, and A. S. Reninger, "Fuzzy multi-level security: An experiment on quantified risk-adaptive access control," in Proceedings of the IEEE Symposium on Security and Privacy (SP), 2007, pp. 222–230.
18. S. Rose, O. Borchert, S. Mitchell, and S. Connelly, "Zero trust architecture," NIST, Gaithersburg, MD, USA, Special Publication 800-207, 2020, doi: 10.6028/ NIST.SP.800-207.
19. A. S. M. Kayes, J. Han, W. Rahayu, M. S. Islam, and A. Colman, "A policy model and framework for context-aware access control to information resources," arXiv:1703.02162, 2017.
20. A. S. M. Kayes, W. Rahayu, T. Dillon, E. Chang, and J. Han, "Context-aware access control with imprecise context characterization through a combined fuzzy logic and ontology-based approach," in On the Move to Meaningful Internet Systems (OTM 2017), Lecture Notes in Computer Science, Springer, 2017, doi: 10.1007/978-3-319-69462-7_10.
21. M. Jones, J. Bradley, and N. Sakimura, "JSON Web Token (JWT)," IETF, RFC 7519, 2015.
22. B. Efron and R. J. Tibshirani, An Introduction to the Bootstrap. New York, NY, USA: Chapman & Hall, 1993.
23. European Parliament and Council, "Regulation (EU) 2016/679 (General Data Protection Regulation)," Official Journal of the European Union, L119, 2016.
24. Government of India, "The Digital Personal Data Protection Act, 2023," Act No. 22 of 2023.