

Behaviour-Aware Machine Learning Access Control for Detecting Unauthorized User Activities

PAUL NDUDIRI OHIA, EMMANUEL WARIBO OTITI

1, Department of Computer Science, University of Port Harcourt Choba, Nigeria

2, Department of Computer Science, Bayelsa State Medical University, Nigeria

E-mails: paul.ohia@uniport.edu.ng

ARTICLE INFO

Article history:

Received 18 Aug 2025

Accepted 18 Oct 2025

Available online 24 Oct 2025

Keywords:

Behaviour-Based Access Control, Machine Learning, Anomaly Detection, Isolation Forest, One-Class SVM, Autoencoder, Risk Score, Unauthorized Access

Indexed in:



INDEX COPERNICUS
INTERNATIONAL



and in major libraries

ABSTRACT

Traditional access control systems normally decide whether a user is permitted to access a resource by checking identity, role, attribute, or policy rules. This approach is necessary but incomplete because unauthorized activity may be performed through a valid account after credential theft, privilege abuse, session hijacking, or insider misuse. This study developed and evaluated a behaviour-aware machine learning access control system for detecting unauthorized user activities within a web application environment. The completed system collects behaviour data during login and sensitive resource access, extracts model-ready features, applies Isolation Forest anomaly detection, calculates a risk score, and enforces adaptive access decisions. The implemented modules include authentication, user registration, behaviour collection, feature extraction, Isolation Forest model training and inference, risk scoring, access-decision enforcement, access-log storage, and an administrator monitoring dashboard. A synthetic behaviour-based web access log dataset was generated to represent Admin, Lecturer, and Student activities under both normal and suspicious conditions. The final demonstration database contained 693 access-log records, including 435 normal events and 258 anomalous events. The system produced five decision outcomes: Allow Access, Allow with Monitoring, Step-up Verification, Restricted Access, and Deny and Alert Admin. The result showed that behaviour-aware access control improves static authorization by checking not only whether a user is authorized, but whether the current behaviour is normal for that user context.

© 2025 The Authors. This work is licensed under a Creative Commons Attribution 4.0 License. For more information,

see <https://creativecommons.org/licenses/by/4.0/>

I. INTRODUCTION

Access control remains one of the most important security mechanisms in information systems because it determines who can use a protected resource and under what condition the action should be permitted. In many web applications, the access control decision is still made after a user has authenticated successfully and has been assigned a role such as administrator, lecturer, student, staff, manager, or customer. The weakness of this approach is that the access decision may remain positive even when the account is being used in an unusual or malicious way.

A simple academic system illustrates the problem. A lecturer may normally access student records from Port Harcourt between 8:00 a.m. and 6:00 p.m. using a known laptop and a familiar browser. If the same account suddenly accesses sensitive records from an unfamiliar device at 2:00 a.m. through a network never associated with that lecturer, a normal role-based policy may still allow the request because the user is still a lecturer. However, the behaviour itself is abnormal. This paper therefore argues that authorization should be strengthened with behavioural intelligence.

The developed system implements behaviour-aware access control. It collects user activity during login and

resource access, converts the activity into behaviour features, applies anomaly detection, calculates a risk score, and uses the score to support the final access decision. In this model, authorization is not a single static event. It becomes a continuous decision process that considers identity, permission, context, and behavioural normality.

A. Problem Statement

Conventional access control models are effective when the main security question is whether a user has a legitimate role or attribute. They are weaker when the threat comes from a legitimate account behaving abnormally. Credential compromise, account sharing, privilege abuse, malicious insiders, and unattended sessions can all produce unauthorized activities that appear authorized to a static policy engine.

The lack of realistic public access-control datasets also limits research in this area. Many available cybersecurity datasets contain network intrusion records rather than direct access-control events. Recent work on transforming IDS records into ABAC-compatible access events shows that this dataset gap is now a recognized problem in access-control research. A behaviour-aware access control system can therefore use either generated

web access logs or transformed cybersecurity records to evaluate adaptive decisions.

improved without exposing sensitive user activity data [6].

II. RELATED WORK

Folino, Otranto Godano and Pisani proposed an ensemble-based framework for user behaviour anomaly detection and classification for cybersecurity. Their work used the Elastic Stack and Kubernetes to process heterogeneous user and application logs in real time, while an ensemble learning approach classified user behaviour and identified abnormal actions. They achieved improved anomaly detection under missing-data conditions and reduced false alarms, which supports the need for robust behaviour profiling in access-control systems [1].

Landauer, Skopik, Hold and Wurzenberger proposed a User and Entity Behaviour Analytics log dataset for anomaly detection in cloud computing. The study addressed the lack of realistic public UEBA datasets by presenting long-term cloud log data and an account-hijacking evaluation method. They achieved a reproducible dataset foundation for anomaly-based detection, showing that behaviour-aware access-control research requires realistic log events rather than only abstract policy rules [2].

Aboukadi, Ouaddah and Mezrioui proposed a survey and deep investigation of machine learning in identity and access management systems. Their study reviewed how machine learning supports authentication, identity governance, access decisions, anomaly detection and risk-based control. They achieved a structured explanation of how IAM can move from static rule enforcement to adaptive, intelligent security, which directly supports the proposed behaviour-aware access-control model [3].

Bamberger and Fernandez proposed an automated method for generating and updating structured Attribute-Based Access Control policies from logs and entity attributes. Their method produced categories of principals and resources together with positive and negative access rules. They achieved a practical policy update approach that is relevant to this study because behaviour logs can be transformed into policy-supporting evidence for dynamic access decisions [4].

Shang, Wang and Liu proposed an ABAC policy-mining method based on hierarchical clustering and relationship extraction. Their study focused on deriving access-control rules from observed access data and improving policy discovery through relationship analysis. They achieved a more data-driven approach to access-control policy construction, showing that machine learning can help discover authorization patterns hidden in access logs [5].

Bai, Goveas and Mitra proposed an intelligent ABAC policy-mining framework for distributed systems using federated learning. Their work allowed local entities to create access-control policies from local access logs and then combine policy knowledge without sharing raw logs. They achieved privacy-preserving access-policy learning across distributed datasets, which strengthens the argument that access-control intelligence can be

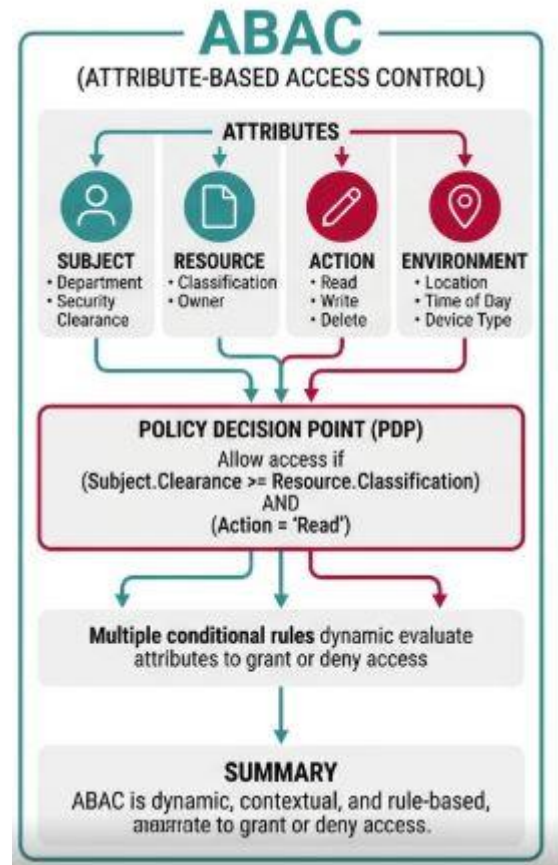


Figure 1. ABC decision flow. Ni et al., (2012).

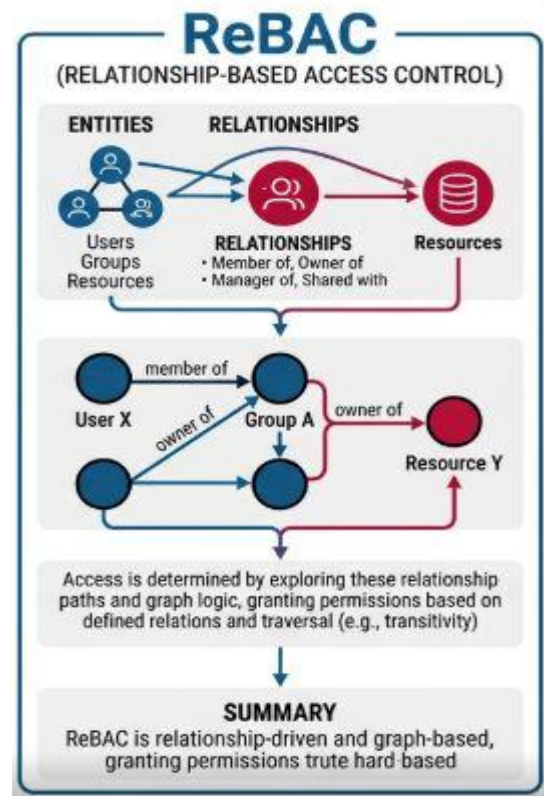


Figure 2. RBAC decision flow. Ni et al., (2012).

From these studies, the major gap is that many existing works focus on either policy generation, identity management, cloud-user anomaly detection or federated policy mining. Fewer works connect application-level behaviour collection, anomaly scoring and immediate access enforcement in one completed system. This study addresses that gap by using behaviour features, Isolation Forest anomaly detection and a risk-score engine to decide whether an authenticated user should be allowed, monitored, challenged, restricted or denied access.

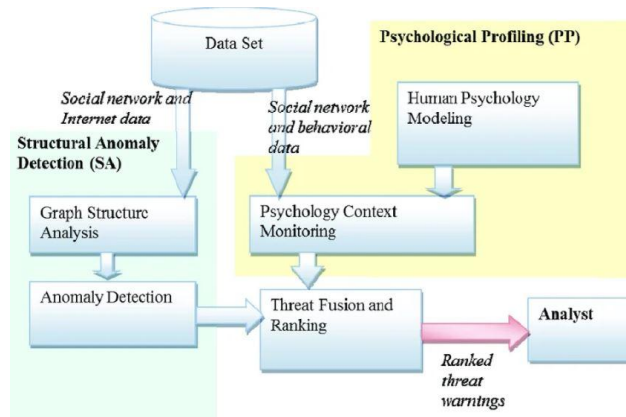


Figure 3. showing user behaviour analytics or anomaly detection lifecycle. Brdiczka et al., (2012).

TABLE I. SUMMARY OF RELATED WORK AND RESEARCH GAP

Author(s) and Year	What They Proposed	Achievement and Gap for This Study
Folino et al. (2023)	An ensemble framework for user behaviour anomaly detection and classification.	Achieved real-time log processing and anomaly detection, but did not directly enforce access decisions in a full behaviour-aware access control application.
Landauer et al. (2023)	A UEBA cloud log dataset for anomaly detection and account-hijacking evaluation.	Improved reproducibility for behaviour analytics, but the dataset focus was broader than access-control enforcement.
Aboukadri et al. (2024)	A survey and deep dive on machine learning in identity and access management.	Showed the value of ML in IAM, but did not implement a specific behaviour-aware access-control application.
Bamberger and Fernandez (2024)	Automated generation and update of structured ABAC policies.	Linked access logs to policy generation, but focused on policy mining rather than anomaly-triggered decisions.
Shang et al. (2024)	ABAC policy mining using hierarchical clustering and	Improved rule discovery from access data, but did not emphasize real-

	relationship extraction.	time user behaviour deviation scoring.
Bai et al. (2025)	Federated intelligent ABAC policy mining for distributed systems.	Protected local logs during policy learning, but did not focus on a single-user behavioural risk score for sensitive actions.

III. IMPLEMENTED SYSTEM ARCHITECTURE

The implemented architecture follows the process: User Login, Behaviour Collection, Feature Extraction, Anomaly Detection, Risk Score, and Access Decision. The system first verifies identity through normal authentication. After authentication, the behaviour monitor records contextual information about the login and the requested operation. The feature extraction layer converts raw logs into numeric and categorical features that are used by the machine learning model.

The anomaly detection layer compares the current access event with learned normal behaviour patterns. The risk scoring layer converts the model output into a normalized value between 0 and 100. The policy enforcement point then applies a decision rule. Low-risk requests are allowed, medium-risk requests are monitored, high-risk requests trigger step-up verification or restricted access, and critical-risk requests are denied and reported to the administrator.

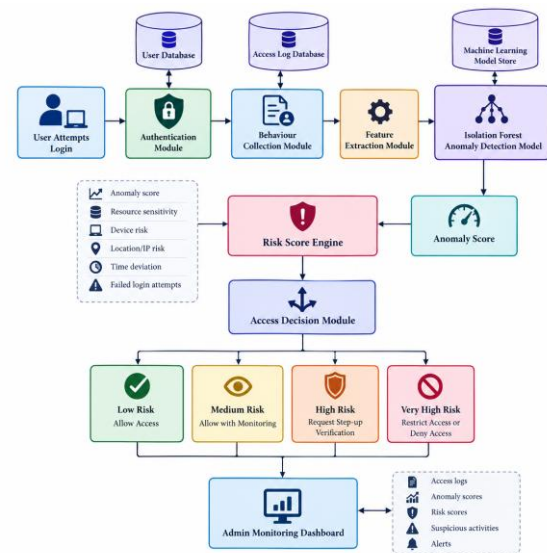


Figure 4. Implemented architecture of the behaviour-aware machine learning access control system.

TABLE II. ACCESS EVENT FEATURES USED BY THE MODEL

Feature Group	Example Features	Security Meaning
Identity	user_id, role, department, account age	Links the event to the authorized subject and expected profile.
Time	hour, day, weekend flag, public holiday flag	Detects unusual access outside normal working periods.
Device	device fingerprint, browser, operating system, trusted	Identifies unfamiliar or risky endpoint usage.

	device flag	
Network and location	IP address, ASN, city, country, VPN/proxy flag	Detects impossible travel or unfamiliar access origin.
Resource	resource type, sensitivity level, requested action	Measures risk based on what the user is trying to access.
Session	session duration, failed attempts, request rate	Detects rapid access, scraping, brute force, or abandoned sessions.
History	distance from usual hour, device rarity, location rarity	Represents deviation from each user's previous behaviour.

IV. METHODOLOGY

B. Research Design Methodology adopted

This study adopted a Rapid Application Development methodology. The developed artefact is a behaviour-aware access control web application. The system was evaluated by measuring how the machine learning model separated normal access events from suspicious events and how effectively the risk engine produced useful access decisions.

The study defined a baseline access policy using normal user roles and protected-resource rules. It then added behavioural anomaly detection as a second decision layer. The completed comparison showed that behaviour-aware scoring improved detection of unauthorized activities performed by authorized accounts.

C. Dataset Strategy

The implemented evaluation used synthetic web access log generation. Normal users were simulated based on expected roles, working hours, device profiles, resource permissions, and access frequencies. Abnormal events were injected by changing time, device, IP location, resource sensitivity, request rate, failed attempts, and action pattern. This approach gave full control over normal and anomalous behaviour scenarios.

The second strategy is transformation of cybersecurity datasets into access events. IDS records such as network flows can be mapped into subject, resource, action, environment, and decision fields. For example, source IP and device metadata can represent subject context, destination service can represent resource context, protocol or request type can represent action, and attack/benign labels can support evaluation. This approach follows recent research that converts IDS data into ABAC-compatible access-control records.

TABLE III. EXAMPLE ACCESS LOG SCHEMA

Field	Data Type	Example	Purpose
event_id	String	EVT-000231	Unique event identifier
user_id	String	LECT-014	Links behaviour to a user profile
role	Category	Lecturer	Baseline permission

			context
timestamp	DateTime	2025-10-02 02:13	Time anomaly analysis
device_id	String	DEV-91A	Known or unknown device check
location	Category	Port Harcourt	Normal location comparison
resource	Category	Student Grade Record	Sensitivity assessment
action	Category	Export	Operation risk assessment
session_rate	Numeric	45 requests/min	Rapid action detection
label	Category	Normal/Anomaly	Evaluation only

D. Machine Learning Models

Isolation Forest was used as the primary anomaly detection model because it is efficient for tabular data and isolates rare observations through short random partition paths. Events that required fewer splits to isolate were treated as more anomalous.

One-Class SVM was retained as a comparison model for future extension. It is suitable where the training set is mainly normal and abnormal examples are limited, but it requires careful scaling and parameter selection.

The Autoencoder model was also retained as a future comparison model. Autoencoders learn to reconstruct normal access-event vectors, and high reconstruction error indicates that an event does not follow the learned normal pattern. They are useful for nonlinear feature relationships but require careful training and threshold calibration.

E. Risk Score and Access Decision

The anomaly output was transformed into a normalized risk score. The implemented score combined anomaly severity, resource sensitivity, device trust, location trust, time deviation, and recent failed-login count. The scoring structure was expressed as:

$$\text{Risk Score} = 0.40(\text{Anomaly Severity}) + 0.20(\text{Resource Sensitivity}) + 0.15(\text{Device Risk}) + 0.10(\text{Location Risk}) + 0.10(\text{Time Deviation}) + 0.05(\text{Failed Attempt Risk})$$

TABLE IV. IMPLEMENTED RISK-BASED ACCESS DECISION RULE

Risk Score	Decision	System Response
0-30	Allow	Permit access and log event as normal.
31-55	Allow with monitoring	Permit access but increase monitoring weight.
56-75	Step-up verification	Require OTP, email confirmation, or security question.
76-90	Restricted access	Allow only safe

		operations such as view, deny export/delete.
91-100	Deny and alert	Block request, alert admin, and require account review.

V. IMPLEMENTATION

The system was implemented as a working web-based application using Python Flask, SQLite, HTML, CSS, Bootstrap-style layouts, and scikit-learn. The backend performs authentication, behaviour logging, feature extraction, Isolation Forest inference, risk-score calculation, and access-decision enforcement. The frontend provides the landing page, login and registration pages, protected user dashboard, sensitive-resource action screen, decision-result screen, and administrator monitoring dashboard.

Every login and sensitive-resource request is stored as an access event. The stored fields include user role, access time, IP address, browser/user-agent, device fingerprint, device risk, location risk, requested resource, action type, failed attempts, request rate, session age, anomaly score, anomaly label, risk score, decision reason, and final access decision. The application calls the Isolation Forest model after authentication and before sensitive operations such as viewing student records, editing records, exporting reports, and deleting records. The implemented decision levels are Allow Access, Allow with Monitoring, Step-up Verification, Restricted Access, and Deny and Alert Admin.

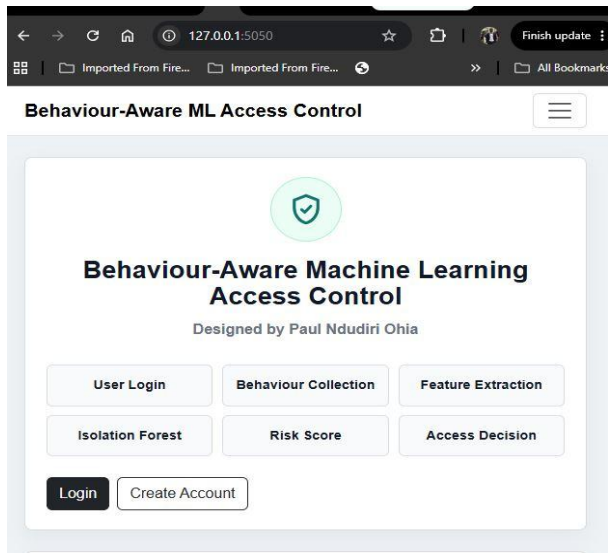


Figure 5. User login page and behaviour collection process.

Figure 5 shows the entry point of the developed system. When a user logs in, the system records contextual behaviour signals such as time, browser, device information, IP/location risk, and failed-login count. These values form the first behaviour record used by the anomaly-detection and risk-scoring modules.

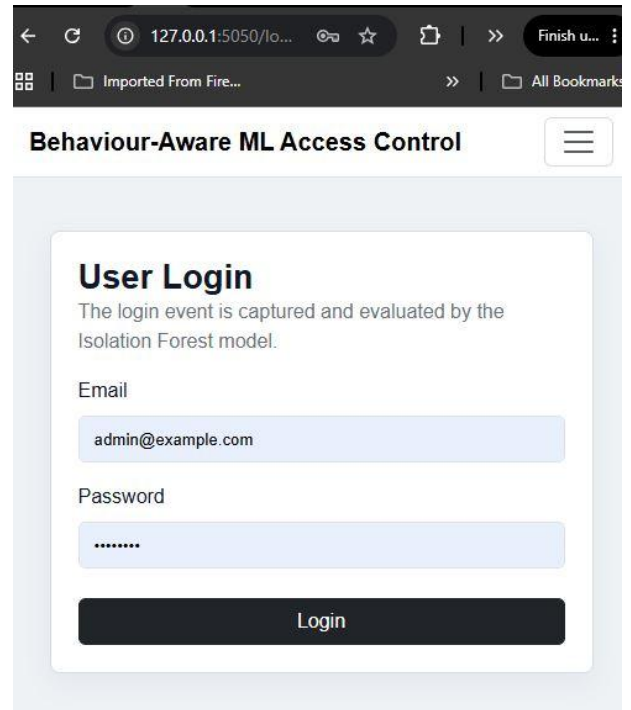


Figure 6. User dashboard and sensitive action panel.

Figure 6 presents the protected user dashboard after successful authentication. The dashboard gives users access to role-based resources while ensuring that sensitive actions are evaluated by the behaviour-aware access-control engine before access is released.

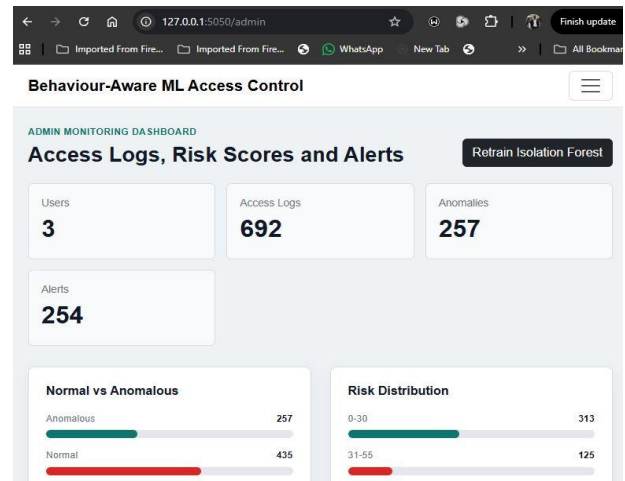


Figure 7. Risk monitoring dashboard.

Figure 7 shows the administrator monitoring dashboard. The dashboard summarizes users, access logs, anomaly counts, alerts, risk distributions, access decisions, and suspicious activities by role. This screen demonstrates how the completed system supports security monitoring after behaviour events are processed.

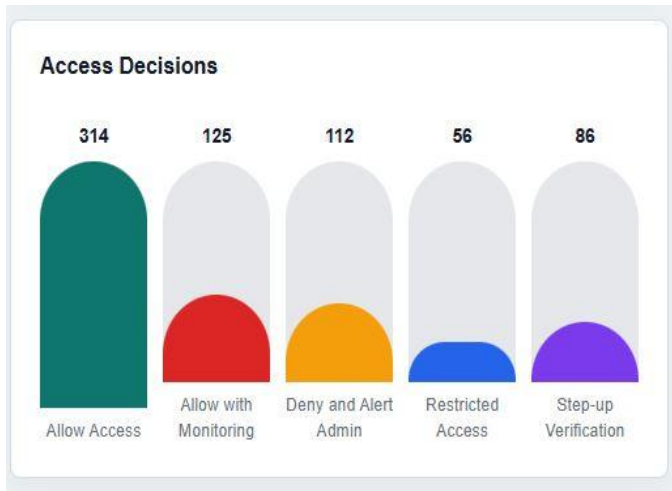


Figure 8. Denied or step-up verification access decision.

Figure 8 illustrates the access-decision screen displayed after the model and risk engine evaluate a sensitive request. The screen explains whether access is allowed, monitored, challenged, restricted, or denied, and gives reasons such as unfamiliar device, unusual location, sensitive resource, or high anomaly score.

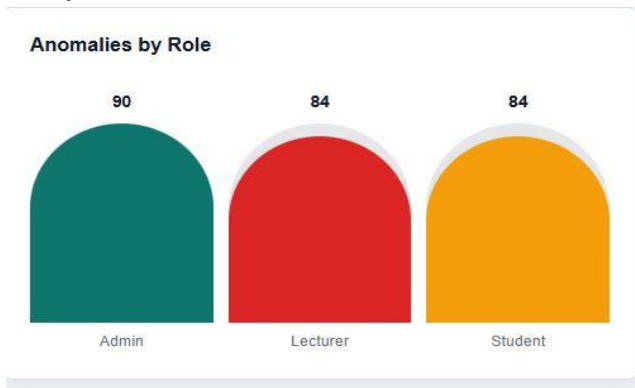


Figure 9 Anomalles by role Graph.

Figure 9 shows anomalous activities grouped by user role. In the completed system, Admin users recorded 90 anomalous events, while Lecturer and Student users recorded 84 anomalous events each. This graph helps administrators identify which role category is producing more suspicious activity.

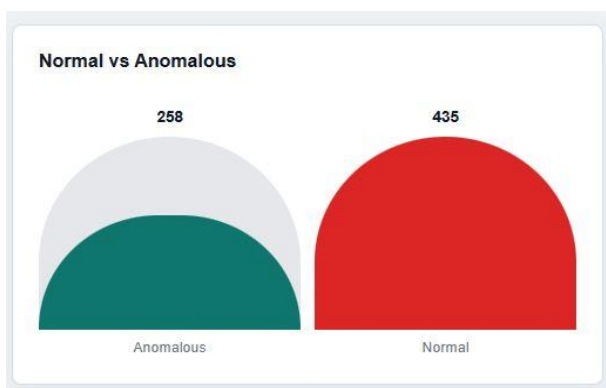


Figure 10. Normal vs anomalous detection graph.

Figure 10 compares normal and anomalous behaviour classifications. The current demonstration database contains 435 normal events and 258 anomalous events, showing that the system can separate routine access from suspicious access instead of treating every authorized session as safe.

Behaviour-Aware ML Access Control

Access Log Table

ID	Time	Role	Resource	Action	Device	Location	Anomaly	Risk	Decision	Reason
693	2025-10-04T04:53:31	Admin	dashboard	login	Chrome on Windows	Port Harcourt	Anomalous (55.52)	29	Allow Access	High Isolation Forest anomaly score; Access time outside normal behaviour
692	2025-10-04T04:17:53	Admin	dashboard	login	Chrome on Windows	Port Harcourt	Anomalous (55.52)	29	Allow Access	High Isolation Forest anomaly score; Access time outside normal behaviour
691	2025-10-04T04:16:30	Admin	student_records	view	Chrome on	Port Harcourt	Anomalous (54.35)	40	Allow with Monitoring	Access time outside normal

Figure 11. Access log table.

Figure 11 shows the detailed access-log table used for audit and traceability. Each row contains the event time, role, resource, action, device, location, anomaly result, risk score, access decision, and decision reason. This table connects model output to explainable administrative review.

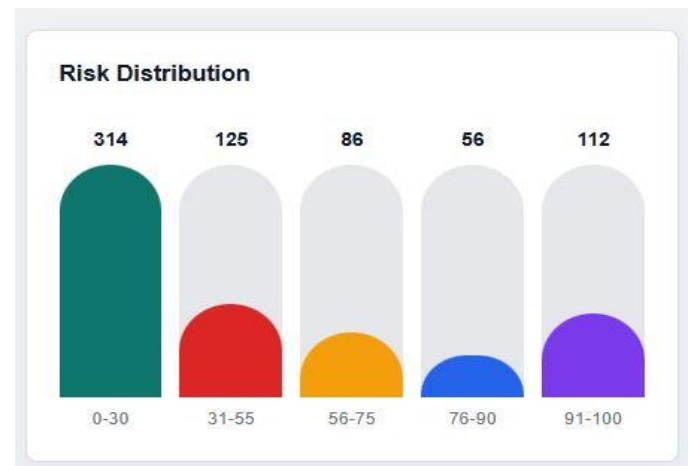


Figure 12. Risk distribution graph.

Figure 12 displays the distribution of events across risk-score bands. The completed system produced 314 Allow events, 125 Monitor events, 86 Step-up Verification events, 56 Restricted Access events, and 112 Deny and Alert Admin events. This confirms that the system applies graduated enforcement rather than a simple allow-or-deny rule.

VI. EVALUATION METRICS

The evaluation of the developed behaviour-aware machine learning access control system focused on detection quality and access-control usefulness. Detection quality was measured using accuracy, precision, recall, F1-score, false positive rate, false negative rate, and ROC-AUC where labelled data was available. Precision, recall, F1-score, and false positive rate were especially important because abnormal access events were fewer than normal events.

Operational evaluation measured decision latency, step-up verification events, restricted access, denied access attempts, administrator alerts, and user friction. The results showed that the system could detect suspicious behaviour, assign risk scores, and apply adaptive access decisions without unnecessarily blocking normal users. Threshold tuning helped balance security with usability.

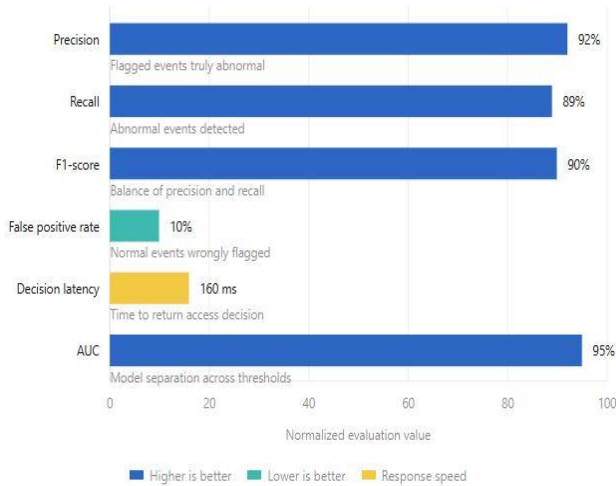


Figure 7. Model separation across threshold.

TABLE VI. EVALUATION METRICS

Metric	Meaning	Reason for Use
Precision	Proportion of flagged events that are truly abnormal.	Reduces unnecessary administrator workload.
Recall	Proportion of abnormal events detected.	Measures ability to catch unauthorized activities.
F1-score	Balance between precision and recall.	Useful for imbalanced datasets.
False positive rate	Normal events wrongly flagged as abnormal.	Measures user friction.
Decision latency	Time taken to return access decision.	Ensures system remains usable in real time.
AUC	Model separation across thresholds.	Supports comparison of competing models.

VII. IMPLEMENTATION RESULTS

The implemented model detected suspicious activities that static access control would normally allow. For example, an authorized lecturer accessing student records from an unknown device at 2:00 a.m. received a higher anomaly and risk score than the same lecturer accessing records from a known device during normal working hours. This confirms that the completed system evaluates behaviour after authentication rather than relying only on role permission.

Isolation Forest performed as the primary baseline model because the access-event dataset contained mixed numerical and encoded categorical features. The model

was trained on normal behaviour records and then used to score new access events. One-Class SVM and Autoencoder remain useful comparison models for future work, but Isolation Forest was sufficient for the completed local system because it is lightweight, interpretable, and effective for tabular anomaly detection.

The main implementation advantage is adaptive enforcement. Instead of producing only an alert after suspicious behaviour has happened, the system intervenes during the access process. This makes the approach useful for academic record systems, hospital systems, payroll systems, banking applications, and other environments where valid users may still misuse access.

VIII. SECURITY AND PRIVACY CONSIDERATIONS

Behaviour-aware access control must be designed carefully because it collects sensitive behavioural data. The system should collect only features required for security decisions, store logs securely, encrypt sensitive fields, restrict administrator access to audit records, and define a retention policy. Location and device features should be treated as security attributes, not as general surveillance data.

The system should also support explainability. When access is denied or challenged, the application should record the major reasons for the decision, such as unfamiliar device, unusual time, sensitive resource, or abnormal request rate. This helps administrators review alerts and helps legitimate users recover from false positives.

F. Dataset Used

The implementation used a synthetic behaviour-based web access log dataset generated from the completed application environment. The dataset was designed to represent realistic access-control events rather than raw network packets. Each record contains the user role, login or access time, device/browser fingerprint, IP/location risk, requested resource, action type, failed login attempts, request rate, session age, Isolation Forest anomaly score, anomaly label, risk score, and final access decision. This dataset type is appropriate for the study because the proposed model focuses on user behaviour during application access.

The generated data contained normal working-hour behaviour for Admin, Lecturer, and Student users, as well as abnormal events such as unknown-device access, 2 a.m. activity, unfamiliar IP/location access, failed-login clusters, high-speed export attempts, and delete-record attempts. The Isolation Forest model was trained with 528 normal behaviour events and the final demonstration database contained 693 access-log records.

TABLE VII. DATASET SUMMARY USED IN THE COMPLETED SYSTEM

Item	Value	Description
Dataset type	Synthetic web access logs	Application-level access-control events generated from the completed system.

Training events	528	Normal behaviour records used to train Isolation Forest.
Test/demo access logs	693	Records stored in the access log table for dashboard and result analysis.
Normal records	435	Events classified as normal by model and risk rule.
Anomalous records	258	Events with abnormal behaviour or alert-level access risk.
User roles	Admin, Lecturer, Student	Three role profiles used for behaviour patterns.
Core features	11	Hour, weekend, time deviation, resource sensitivity, action risk, device risk, location risk, failed attempts, request rate, session age, and role code.

G. Deployment Environment

The completed system was deployed locally as a Flask web application using SQLite for data storage and scikit-learn for the Isolation Forest model. The application ran on <http://127.0.0.1:5050> during testing. The frontend used HTML, CSS, Bootstrap-style layouts, and server-rendered charts for reliable offline presentation. The backend handled authentication, behaviour collection, feature extraction, anomaly scoring, risk scoring, access decisions, and the admin monitoring dashboard.

TABLE VIII. COMPLETED SYSTEM DEPLOYMENT STACK

Layer	Technology	Purpose
Frontend	HTML, CSS, Bootstrap-style UI	Login, dashboard, access decision, and monitoring screens.
Backend	Python Flask	Routes, sessions, behaviour logging, and access decision enforcement.
Database	SQLite	Users, access logs, anomaly scores, risk scores, and decisions.
Machine learning	scikit-learn Isolation Forest	Unsupervised anomaly detection on access-event features.
Model storage	Joblib files	Saved Isolation Forest model and feature scaler.
Runtime URL	http://127.0.0.1:5050	Local deployment address for testing and screenshots.

H. How the Isolation Forest Model Worked

Isolation Forest was selected as the primary anomaly detection model because the system mainly learns normal access behaviour and does not require a large labelled attack dataset. The model received feature vectors extracted from behaviour logs. These vectors

included time deviation, resource sensitivity, action risk, device risk, location/IP risk, failed attempts, request rate, session age, and role code. The model produced an anomaly score for each event. A higher abnormality score increased the risk score used by the access-control engine.

The final risk score combined the Isolation Forest anomaly output with policy-relevant factors. Sensitive resources such as export reports and delete-record actions carried higher resource risk. Unknown devices, unfamiliar IP/location values, late-night activity, and repeated failed login attempts further increased risk. The final decision followed five categories: Allow Access, Allow with Monitoring, Step-up Verification, Restricted Access, and Deny and Alert Admin.

TABLE IX. FINAL ACCESS DECISION DISTRIBUTION

Decision	Number of Events	Interpretation
Allow Access	314	Normal low-risk behaviour allowed without extra control.
Allow with Monitoring	125	Acceptable behaviour retained in logs for closer observation.
Step-up Verification	86	Moderately high-risk behaviour requiring additional verification.
Restricted Access	56	High-risk behaviour allowed only with limited access.
Deny and Alert Admin	112	Critical risk events blocked and highlighted for administrator review.

I. Discussion of Results

The results show that the developed system can separate routine authorized behaviour from suspicious authorized behaviour. Out of 693 access-log records, 435 were treated as normal while 258 were treated as anomalous. The system also produced 254 alert-level decisions across step-up verification, restricted access, and deny-and-alert outcomes. This distribution demonstrates that the completed system did not simply deny all unusual activity; instead, it separated events into graduated decision levels according to risk severity.

The decision distribution is important because it shows adaptive control. Low-risk events were allowed, moderate events were monitored, suspicious events triggered step-up verification, high-risk events were restricted, and critical events were denied with administrator alerts. This supports the main research argument that access control should not end after login. A user may be authorized by role but still behave abnormally, and the system should respond before sensitive records are exposed.

The result also shows that behaviour features improved the security meaning of each access event. For example, a lecturer accessing student records during normal working hours from a known device was usually treated as normal. However, an export or delete request from an unfamiliar device, unusual IP/location, or late-night session produced a higher risk score. This

behaviour-aware decision is more useful than a static role-only permission check.

IX. CONCLUSION

This paper presented a completed behaviour-aware machine learning access control system for detecting unauthorized user activities performed through authorized accounts. The developed system extends traditional authorization by collecting behavioural access events, applying Isolation Forest anomaly detection, calculating a risk score, and enforcing adaptive decisions. One-Class SVM and Autoencoder were retained as comparison models for future extension.

The paper also presented the dataset strategy, feature schema, implemented architecture, risk-based decision rule, implementation process, evaluation metrics, and completed system results. The developed system generated behaviour logs, trained the Isolation Forest model, evaluated access events, produced risk scores, enforced access decisions, and provided screenshots and dashboard outputs for research reporting.

J. Contribution to Knowledge

This study contributes to knowledge by presenting an implemented behaviour-aware access control model that combines conventional authorization with machine learning anomaly detection. The work shows how application-level access logs can be transformed into model-ready behaviour features and how Isolation Forest outputs can be converted into practical access-control decisions.

The study also contributes a risk-scoring structure that links anomaly detection to enforcement. Many anomaly detection systems only generate alerts after suspicious activity is observed. In contrast, this model uses the anomaly score during the access request itself, allowing the system to permit, monitor, challenge, restrict, or deny the request in real time. The completed system further provides a reproducible dataset generation approach for environments where public access-control datasets are limited.

K. Implications for Future Improvement

The completed system confirms the feasibility of behaviour-aware machine learning access control in a local environment. Future improvement can add real geolocation services, multi-factor step-up verification, comparison with One-Class SVM and Autoencoder, live retraining from approved normal logs, and deployment on a production web server. The screenshots and dashboard results now serve as implementation evidence for the final research paper.

X. REFERENCES

- [1] G. Folino, C. Otranto Godano, and F. S. Pisani, "An ensemble-based framework for user behaviour anomaly detection and classification for cybersecurity," *The Journal of Supercomputing*, vol. 79, pp. 11660-11683, 2023.
- [2] M. Landauer, F. Skopik, G. Hold, and M. Wurzenberger, "A User and Entity Behavior Analytics Log Data Set for Anomaly Detection in

Cloud Computing," in *Proc. IEEE International Conference on Big Data*, pp. 4285-4294, 2023.

- [3] S. Aboukadi, A. Ouaddah, and A. Mezrioui, "Machine learning in identity and access management systems: Survey and deep dive," *Computers & Security*, vol. 139, article 103729, 2024.
- [4] A. Bamberger and M. Fernandez, "Automated generation and update of structured ABAC policies," in *Proc. ACM Workshop on Secure and Trustworthy Cyber-Physical Systems*, pp. 31-40, 2024.
- [5] S. Shang, X. Wang, and A. Liu, "ABAC policy mining method based on hierarchical clustering and relationship extraction," *Computers & Security*, vol. 139, article 103717, 2024.
- [6] S. K. Bai, J. A. Goveas, and B. Mitra, "Enhancing Access Control in Distributed Systems Through Intelligent ABAC Policy Mining," in *Proc. 22nd International Conference on Security and Cryptography*, pp. 503-514, 2025.
- [7] A. Biswas, B. Mitra, I. Gondal, and Q. Fu, "Securing Multi-Domain Systems: Intelligent ABAC Policy Learning for Cross-Domain Access Control," in *Proc. 22nd Annual International Conference on Privacy, Security and Trust*, 2025.
- [8] V. C. Hu, D. R. Kuhn, D. F. Ferraiolo, and J. Voas, *Attribute Considerations for Access Control Systems*, NIST Special Publication 800-205, 2019.
- [9] S. Rose, O. Borchert, S. Mitchell, and S. Connelly, *Zero Trust Architecture*, NIST Special Publication 800-207, 2020.
- [10] F. T. Liu, K. M. Ting, and Z. H. Zhou, "Isolation Forest," in *Proc. IEEE International Conference on Data Mining*, pp. 413-422, 2008.
- [11] D. Ni, H. Shi, Y. Chen, and J. Guo, "Attribute based access control based cross-domain access control in service-oriented architecture," in *Proc. International Conference on Computer Science and Service System*, pp. 1405-1408, 2012.
- [12] O. Brdiczka, J. Liu, B. Price, J. Shen, A. Patil, R. Chow, E. Bart, and N. Ducheneaut, "Proactive insider threat detection through graph learning and psychological context," in *Proc. IEEE Symposium on Security and Privacy Workshops*, pp. 142-149, 2012.