

Explainable Tree-Based Machine Learning for Event-Level Lateral Movement Detection in Active Directory

Antonius Djajakusuma^{1*)}, Theresia Herlina²⁾

^{1,2)}Department of Information Technology, Pradita University, Indonesia

¹⁾antonius.djakusuma@student.pradita.ac.id, ²⁾theresia.herlina@pradita.ac.id

Submitted : Sep 13, 2026 | Accepted : Oct 9, 2026 | Published : Oct 13, 2026

Abstract: Windows Active Directory governs authentication across enterprise networks, making it a primary target for lateral movement techniques such as Pass-the-Hash and Golden Ticket, which disguise themselves as legitimate administrative activity within high-volume Sysmon logs. Prior work reported near-perfect accuracy for tree-based machine learning on this task but relied on global feature-importance methods that cannot explain individual predictions, leaving the source of that accuracy unexamined. This study aims to compare four tree-based algorithms and to determine which technical indicators actually drive their decisions. The method comprises leakage-aware feature selection, stratified splitting, partial SMOTE oversampling applied to training data only, evaluation through Macro F1-Score, Matthews Correlation Coefficient, ROC AUC and per-event inference time, and instance-level interpretation using SHAP and LIME, supported by an ablation study across four feature sets. On 35 behavioral features Extra Trees achieved the highest Macro F1 of 0.9370, and every algorithm exceeded the throughput required for event-level stream processing. The ablation study produced the central result: on the full feature set the model reached a near-perfect Macro F1 of 0.9998, yet this stemmed largely from memorizing attack-tool identities rather than learning behavior, because precision on the credential-theft class collapsed to 0.7343 in the ablation setting once binary-identity features were removed, corresponding to 0.7345 for the same feature set in the main 200-estimator run. It is concluded that Explainable AI serves not only as a transparency mechanism but as a diagnostic instrument capable of exposing shortcut learning behind seemingly excellent detection performance.

Keywords: Active Directory; Explainable AI; Lateral Movement; Machine Learning; Sysmon

INTRODUCTION

Windows Active Directory is the principal identity management system in Windows-based enterprise networks, providing a centralized database for user authentication and authorization. Compromising a domain controller grants an attacker control over every organizational asset connected to the directory (Smiliotopoulos, Kambourakis, & Barbatsalou, 2023). This characteristic makes Active Directory a primary target in advanced cyber attacks.

A critical phase in such attacks is lateral movement, in which the attacker expands privileges and moves between systems after breaching the initial perimeter. This movement exploits weaknesses in the Kerberos and Server Message Block protocols through techniques including Pass-the-Hash, Pass-the-Ticket, Golden Ticket, and Silver Ticket, allowing the attacker to obtain administrator privileges without passing through standard password validation (Bai et al., 2021; Smiliotopoulos et al., 2023). Detection is difficult because attackers disguise their activity to resemble legitimate administrative behavior (Smiliotopoulos, Kambourakis, & Kolias, 2024). Pass-the-Hash reuses password hash values stolen from memory, while Golden Ticket forges a Kerberos ticket-granting ticket using the directory service account key. Both generate authentication traffic that is syntactically valid, so static rules cannot separate them from normal activity. Without early detection, an attacker may retain hidden access for a long period, and the interval between initial compromise and detection determines the magnitude of the damage.

Security visibility in this environment depends on log quality. Windows System Monitor, commonly known as Sysmon, records process creation and network connections in real time, yielding deeper visibility than standard

*name of corresponding author

This is an Creative Commons License This work is licensed under a Creative Commons Attribution-NonCommercial 4.0 International License.

Windows logs. The volume, however, is very high: the LMD-2023 dataset contains more than 1.75 million log samples from a single testbed. This motivates machine learning approaches that automatically separate threat anomalies from normal activity. Yet evaluations of log-based anomaly detection often fail to represent real conditions, because reported accuracy proves inconsistent once training data selection, class distribution, and early detection capability are considered (Le & Zhang, 2022). Advanced models, including tree-based algorithms, are moreover black boxes whose internal decision process cannot be inspected directly (Barredo Arrieta et al., 2020), leaving analysts unable to verify whether a prediction rests on operationally relevant indicators or on meaningless correlations (Gaspar, Silva, & Silva, 2024).

This study addresses three questions. First, how the four tree-based algorithms compare in detecting lateral movement on LMD-2023. Second, how consistent their performance differences are and how large the practical meaning of those differences is. Third, which technical features actually drive predictions according to SHAP and LIME. Detection in this study is defined operationally at the event level: each Sysmon event is classified independently, without waiting for session aggregation or a time window. This design permits an alert on the first anomalous event observed, but the present work measures only classification at that granularity. It does not measure time-to-detection, the position of the first detected event within an attack sequence, or the delay from attack onset, so no claim of earliness within the attack chain is made here. (Jin, Lu, Qin, Cheng, & Mao, 2020; Milajerdi, Gjomemo, Eshete, Sekar, & Venkatakrishnan, 2019)

The contribution is threefold. This study presents a lateral movement detection framework combining tree-based algorithms with local interpretation, provides a direct comparison against published benchmark performance while revealing the source of that performance through an ablation study, and reports per-event inference time as an indicator of real-time deployment feasibility.

LITERATURE REVIEW

Research on lateral movement detection has taken several directions. The benchmark study by Smiliotopoulos et al. (2023) detected lateral movement in Sysmon logs using six tree-based algorithms and reported Extra Trees as the best performer, with a ROC AUC of 99.84% and an F1-Score of 99.41%. Feature interpretation in that study, however, relied only on logistic regression coefficients and Principal Component Analysis, both global in nature and unable to explain predictions for individual samples. A follow-up study (Smiliotopoulos & Kambourakis, 2026) extended the evaluation by comparing resampling techniques on the same dataset, but likewise did not apply local interpretation.

Other approaches use graph analysis and authentication logs. Graph modeling of Active Directory infrastructure has been shown to delay the time to domain administrator compromise through hardening of critical nodes (Herranz-Oliveros, Tejedor-Romero, Gimenez-Guzman, & Cruz-Piris, 2024), and detection based on the Remote Desktop protocol has been studied using machine learning (Bai et al., 2021). Sequence models have been applied to Windows audit logs for malware detection (Ring, Schlör, Wunderlich, Landes, & Hotho, 2021), while insider threat detection has been examined using conventional classifiers (Bin Sarhan & Altwaijry, 2023). A systematic survey grouped the literature into endpoint detection and response, graph-based, and machine learning approaches, noting that interpretability remains rarely discussed (Smiliotopoulos et al., 2024).

Explainable AI has advanced in adjacent domains. Combining SHAP and LIME improves the transparency of multi-layer perceptron models (Gaspar et al., 2024) and Random Forest models (Wali, Farrukh, & Khan, 2025), and transparent pipelines that surface feature attributions to analysts have been proposed (Mallampati & Seetha, 2024). A two-stage framework with SHAP-based explanations has been applied to low-frequency attacks (Mahmoud, Youssef, & Abdel-Hamid, 2025), and black-box Explainable AI frameworks for network intrusion detection have been evaluated systematically (Arreche, Guntur, Roberts, & Abdallah, 2024). In the Internet of Things domain, Explainable AI has been integrated with federated learning (Kalakoti, Nömm, & Bahsi, 2025), evaluated on data streams (Alabbadi & Bajaber, 2025), and embedded in detection frameworks (Bin Hulayyil, Li, & Saxena, 2025).

A gap therefore remains. Existing work on Sysmon-based lateral movement achieves high accuracy but supplies neither local interpretation nor inference cost, even though deployment feasibility on real-time log streams depends on the latter. Equally important, no study has asked whether the reported accuracy reflects genuine behavioral learning or reliance on identifying attributes present in the data, which is the question this study sets out to answer.

METHOD

The research was conducted in six sequential stages: data collection, feature selection, data splitting, preprocessing with class balancing, training with evaluation and statistical validation, and interpretation. All experiments were executed in Python on Google Colaboratory. Baseline modeling used Scikit-Learn (Pedregosa et al., 2011), ensemble training used the XGBoost and CatBoost libraries, class balancing used Imbalanced-learn,

and interpretation used the SHAP and LIME libraries. The random seed was fixed at 42 so that results are reproducible.

The dataset is LMD-2023, a collection of Sysmon logs constructed specifically for lateral movement research (Smiliotopoulos et al., 2023). A single dataset was used deliberately, reflecting the scarcity of alternatives: public Sysmon corpora that combine labelled lateral movement activity, sufficient event volume, and a realistic Active Directory topology remain rare, and most public intrusion detection datasets capture network flows rather than host-level telemetry and therefore cannot support the behavioral features examined here. Using the same dataset as the benchmark also permits direct comparison. The published dataset contains 1,752,890 samples with 93 raw features; after removal of malformed records, 1,752,836 records remained and were used throughout, and all percentages are computed against this analyzed total. The data source is Sysmon logs collected on domain member hosts rather than event logs on the domain controller; Active Directory serves as both environment and target. The class distribution appears in Table 1.

Table 1. Structure of the LMD-2023 Dataset

Label	Class	Sample Count	Description
0	Normal	1,611,619 (91.94%)	Sysmon traffic from normal network activity
1	EoRS	110,710 (6.32%)	Exploitation of remote services such as EternalBlue and BlueKeep
2	EoHT	30,507 (1.74%)	Exploitation of hashing techniques such as Pass-the-Hash and Golden Ticket

Note: counts are exact for the 1,752,836 analyzed records and sum to that total; all percentages in this manuscript are computed against it. The published dataset reports 1,752,890 samples, the difference of 54 records being malformed rows removed before analysis. The corresponding test-partition counts appear in the support column of Table 3.

Raw Sysmon logs contain many unique identifier columns such as Globally Unique Identifiers, hash values, and event record numbers. These carry no behavioral information, and encoding them as categories invites the model to memorize identifiers rather than learn attack patterns, producing spurious accuracy that does not generalize. Feature selection therefore proceeded in three stages. The first removed explicit identifiers including ProcessGuid, EventRecordID, and raw timestamps, along with categorical columns of very high cardinality that behave as identifiers. The second removed Sysmon collector artifacts such as Configuration, ConfigurationFileHash, and RuleName, which describe the collection configuration rather than attacker behavior. The third protected behavioral features such as GrantedAccess and CallTrace from automatic filtering, since certain process access values indicate credential memory reading. This constitutes a deliberate deviation from the benchmark, which reduced the raw features to fifteen through Principal Component Analysis; here features were retained on label-leakage criteria instead. The flow of features through these stages runs as follows. Of the 93 raw columns, 37 were discarded by the three screening stages, leaving 56, to which the six derived features described below were added, producing the 62 features of Scenario A. Within those 62, two groups were isolated for the ablation study: 14 binary-identity features, which name the executable involved through file name, product name, publisher, and related attributes, and 13 environment-identity features, which name the host, user, or domain in which the event occurred. Removing the 13 environment-identity features yields the 49 features of Scenario B, removing the 14 binary-identity features yields the 48 of Scenario C, and removing both groups leaves the 35 behavioral and protocol features of Scenario D, which is the main analysis path.

Six derived features were constructed from command-line and process image attributes to capture behavior rather than identity: command-line length, Shannon entropy, argument count, an encoded-command indicator, a lateral movement keyword indicator, and a living-off-the-land binary indicator. Such markers cannot be evaded by renaming a file, making them conceptually more robust to disguise than identity features.

Data were split in stratified fashion into 80% training and 20% test partitions, preserving class proportions in both. All preprocessing transformations were learned from the training data alone and then applied to the test data, preventing test statistics from leaking into training. Preprocessing covered missing value handling through median and mode, categorical encoding, Z-score standardization, and precision reduction for memory efficiency; standardization was retained because LIME builds local linear models requiring a uniform feature scale. The Synthetic Minority Over-sampling Technique was applied to the EoRS and EoHT classes on the training data only, leaving the test distribution intact so that evaluation reflects real conditions. Balancing was partial, raising minority classes to 30% of the majority rather than equalizing them, because full equalization exceeds the memory capacity of the environment and generates synthetic samples far exceeding the original EoHT samples, risking distortion of the decision boundary.

Because the dataset is imbalanced, standard accuracy is unrepresentative, so four metrics were used: Macro F1-Score, which weights each class equally; the Matthews Correlation Coefficient, which measures classification

quality reliably on imbalanced data; ROC AUC, which measures class discrimination; and per-event inference time, which supports the operational definition of event-level detection. Four tree-based ensembles were compared. Random Forest builds many trees on bootstrap samples and aggregates their votes (Breiman, 2001). Extra Trees extends this by selecting split thresholds at random, reducing variance and shortening training (Geurts, Ernst, & Wehenkel, 2006). XGBoost builds trees sequentially through regularized gradient boosting (Chen & Guestrin, 2016). CatBoost applies ordered boosting with symmetric trees to reduce prediction shift (Prokhorenkova, Gusev, Vorobev, Dorogush, & Gulin, 2018). All four were trained with 200 estimators and the same seed. No hyperparameter search was performed, so the algorithms were compared under an equivalent budget rather than individually optimized settings; because boosting is generally more sensitive to learning rate and tree depth than bagging, the comparison reflects equivalent rather than optimal configurations.

Statistical comparison used two procedures. The Friedman test, with the Nemenyi post-hoc test applied when the p-value fell below 0.05, was run on Macro F1 scores from stratified five-fold cross-validation of the training data as an indicative check on whether performance differences are consistent rather than incidental (Demšar, 2006). Folds drawn from one dataset are not fully independent, so those p-values are optimistic and are reported as indicative only. The confirmatory comparison therefore used the McNemar test on paired predictions over the held-out test set, the procedure appropriate to a single dataset. For each pair of algorithms the discordant counts were formed from the events that one model classified correctly and the other did not, the chi-square statistic was computed with the Edwards continuity correction, and the six resulting p-values were adjusted by the Holm-Bonferroni procedure. Effect size was reported in two forms: the odds ratio of the discordant counts and the difference in overall accuracy with a 95% confidence interval. Interpretation used two methods. SHAP distributes feature contributions using Shapley values from game theory (Lundberg & Lee, 2017), with tree-specific development enabling efficient local explanations (Lundberg et al., 2020), while LIME builds a simple linear model around a single data point to explain the decision on that sample (Ribeiro, Singh, & Guestrin, 2016). Both were applied to the best-performing model at global and individual levels. Agreement between the two methods was quantified rather than inspected on a single case. Ninety test events were drawn by stratified sampling, fifteen from each combination of true class and prediction outcome, so that correctly and incorrectly classified events of all three classes are represented equally. For each event the ten highest ranked features of each method were compared through the size of their intersection, the Jaccard index, and the Spearman rank correlation across the features ranked by both. The reference distribution supplied to LIME was a random subsample of 100,000 training events taken before oversampling.

RESULT

The four algorithms were trained on the 35 behavioral features and evaluated on test data retaining the original class distribution. Table 2 reports overall performance together with inference throughput.

Table 2. Performance and Inference Throughput on 35 Behavioral Features

Algorithm	Macro F1	MCC	ROC AUC	Training (s)	Throughput (events/s)
Random Forest	0.9368	0.9507	0.9977	218.3	76,079
Extra Trees	0.9370	0.9509	0.9973	266.8	31,686
XGBoost	0.8980	0.9138	0.9981	176.3	71,927
CatBoost	0.9222	0.9347	0.9980	303.2	304,316

Extra Trees obtained the highest Macro F1 at 0.9370, followed very closely by Random Forest at 0.9368, while the boosting algorithms scored lower at 0.8980 and 0.9222. Both boosting algorithms nevertheless achieved marginally higher ROC AUC, indicating that the two metrics capture different aspects: ROC AUC measures probability ranking, whereas Macro F1 measures final decision quality at the standard threshold with equal class weight. The Matthews Correlation Coefficient reinforces the same ordering, highest for Extra Trees and Random Forest at 0.9509 and 0.9507 and lowest for XGBoost at 0.9138. Throughput varied by an order of magnitude, from 31,686 events per second for Extra Trees to 304,316 for CatBoost. Per-class metrics, with emphasis on the EoHT minority class, appear in Table 3.

Table 3. Per-Class Metrics of the Four Algorithms

Algorithm / Class	Precision	Recall	F1-Score	Support
Random Forest – Normal	0.9986	0.9936	0.9961	322,325
Random Forest – EoRS	0.9922	0.9885	0.9904	22,142
Random Forest – EoHT	0.7337	0.9395	0.8240	6,101
Extra Trees – EoHT	0.7345	0.9395	0.8245	6,101
XGBoost – EoHT	0.5802	0.9359	0.7163	6,101
CatBoost – EoHT	0.6814	0.9376	0.7892	6,101

Note: Normal and EoRS rows are shown for Random Forest only; for the other algorithms these classes yielded values within 0.005 of those reported, so their rows are omitted for brevity.

A consistent pattern emerges. Precision on the Normal and EoRS classes is very high across all algorithms, whereas precision on EoHT is far lower, reaching only 0.7345 for Extra Trees and falling to 0.5802 for XGBoost. Recall on EoHT nevertheless remains high everywhere, between 0.9359 and 0.9395. The models locate almost all EoHT events, but a substantial share of their alerts are false positives.

Global feature importance was estimated for the Extra Trees model using TreeSHAP on a random subsample of 500 test instances rather than the full test set, so the ranking in Figure 1 is a subsample-based estimate of global importance.

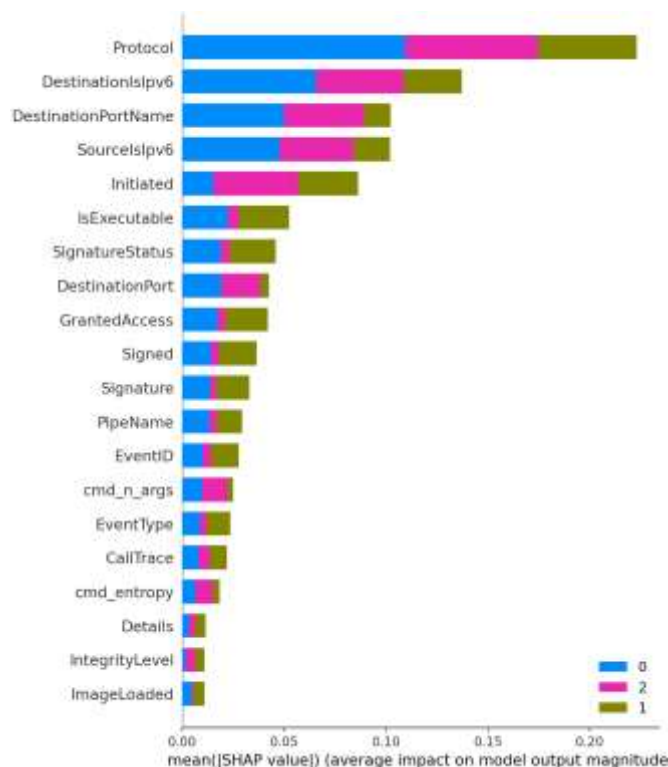


Fig. 1 Global feature importance according to SHAP for the Extra Trees model

The largest contribution comes from Protocol, followed by DestinationIsIpv6, DestinationPortName, SourceIsIpv6, and Initiated, all describing network connection properties and consistent with lateral movement as inter-system communication over particular protocols and ports. Lower-ranked features describe process and file behavior, namely IsExecutable, SignatureStatus, GrantedAccess, and PipeName. The presence of GrantedAccess aligns with domain knowledge, since certain access values indicate credential memory reading, and PipeName is likewise meaningful because named pipes are common in lateral movement. Before leakage-aware filtering was applied, the highest contributing features had instead been ConfigurationFileHash, Configuration, and RuleName, all artifacts of the collector configuration rather than indicators of attack behavior.

The ablation study compared four feature set scenarios while holding model, data, and procedure constant. Because four retraining runs were required, it used 100 estimators rather than 200; scenario D under this setting reproduced the Macro F1 of the main experiment, indicating that the reduction did not affect the conclusions. Results appear in Table 4.

Table 4. Ablation Study across Four Feature Set Scenarios (Extra Trees)

Scenario	Features	Macro F1	Recall EoHT	Precision EoHT
A. All features	62	0.9998	0.9995	0.9998
B. Without environment identity	49	0.9944	0.9992	0.9794
C. Without binary identity	48	0.9560	0.9780	0.7853
D. Behavior only (main path)	35	0.9370	0.9395	0.7343

Note: the ablation used 100 estimators to keep four retraining runs tractable, whereas Tables 2 and 3 report the main 200-estimator run. Scenario D therefore corresponds to the main model, and the marginal difference in EoHT precision (0.7343 here versus 0.7345 in Table 3) reflects only this configuration difference, not a different experiment. The feature counts follow the selection described in the Methods: Scenario A retains all 62 selected features, Scenario B removes the 13 environment-identity features, Scenario C removes the 14 binary-identity features, and Scenario D removes both groups, leaving 35.

Performance declines progressively as feature groups are removed. Scenario A, retaining all features including identity attributes, reached a Macro F1 of 0.9998 and EoHT precision of 0.9998, whereas scenario D, restricted to behavioral features, reached EoHT precision of 0.7343, a gap of 26.6 percentage points. The sharpest single drop occurs between scenarios B and C, where binary-identity features including file names, product names, and publisher names are removed and EoHT precision falls from 0.9794 to 0.7853, a decline of 19.4 percentage points. Recall on the EoRS class, by contrast, remained high in every scenario.

Activation levels and contributions of the six derived command-line features appear in Table 5, computed against the 1,752,836 analyzed records.

Table 5. Activation and Importance of Derived Command-Line Features

Feature	Active Events	Percentage	Model Importance
cmd_length	1,752,836	100.0%	0.0066
cmd_entropy	27,393	1.56%	0.0352
cmd_n_args	12,577	0.72%	0.0243
is_lolbin	589	0.034%	0.00025
cmd_encoded	88	0.005%	0.00003
cmd_lm_action	77	0.004%	0.00022

Statistical features contribute measurably while security-semantic features contribute almost nothing. Entropy, argument count, and length obtained meaningful importance, whereas the encoded-command and lateral movement keyword indicators were active on only 88 and 77 events respectively. The command-line column in this dataset is largely empty, averaging about 2.6 characters, because most events are network connections and process accesses that carry no command-line attribute; only process creation events do.

Local interpretation was applied to a single correctly classified EoHT event using both methods, shown in Figure 2 and Figure 3.

**Fig. 2 Local SHAP explanation for a single EoHT event**

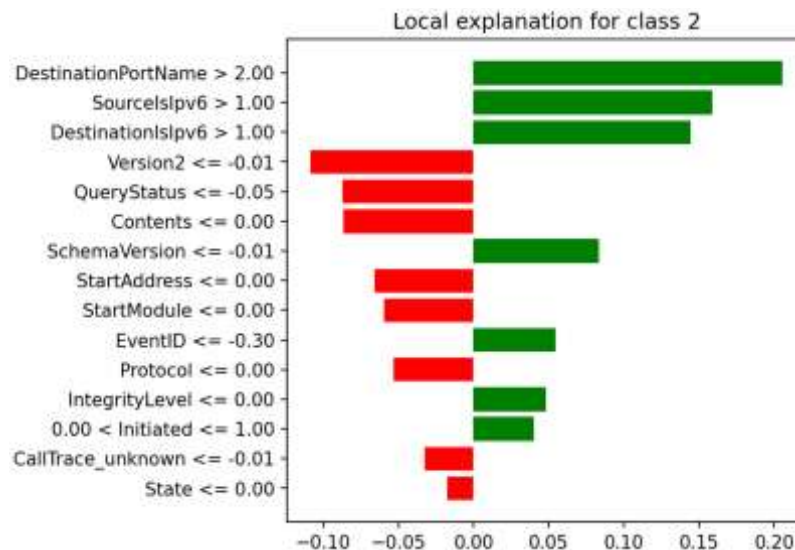


Fig. 3 Local LIME explanation for the same event

Both methods agree on the dominant drivers of this event, with DestinationPortName, SourceIsIpsv6, and DestinationIsIpsv6 appearing as the principal contributors in each, while only three of their ten highest ranked features coincide. A single event cannot establish how far the two methods agree in general, so the comparison was repeated over the stratified sample of ninety events. Across that sample the mean overlap among the ten highest ranked features was 4.89 with a standard deviation of 1.19, an agreement level of 48.9%; the mean Jaccard index was 0.33 and the mean Spearman rank correlation 0.38 with a standard deviation of 0.15. The two methods selected the same single most important feature in 35.6% of events and shared on average 1.41 of their three highest ranked features. Agreement varied with class and with correctness, reaching 5.93 of ten on correctly classified EoRS events and falling to 3.33 of ten on misclassified EoHT events, with correctly classified events exceeding misclassified ones within every class.

The indicative Friedman test produced a statistic of 13.08 with a p-value of 0.0045 across five cross-validation folds, whose mean Macro F1 scores were 0.980805 for Random Forest, 0.980832 for Extra Trees, 0.970691 for XGBoost, and 0.971225 for CatBoost, with standard deviations below 0.0006 in every case. This figure is reported for continuity with the cross-validation procedure and is not relied upon as evidence of significance. The confirmatory McNemar test was then applied to paired predictions over all 350,568 test events, with results in Table 6.

Table 6. Pairwise McNemar Test on Paired Test-Set Predictions

Comparison (A vs B)	n01 / n10	χ^2	p (Holm)	Δ Accuracy (95% CI)	Odds Ratio
Extra Trees vs Random Forest	17 / 29	2.63	0.1048	0.000034 (-0.000004, 0.000072)	1.71
Extra Trees vs CatBoost	164 / 1,101	692.57	< 0.001	0.002673 (0.002474, 0.002872)	6.71
Extra Trees vs XGBoost	118 / 2,358	2,024.69	< 0.001	0.006390 (0.006111, 0.006668)	19.98
Random Forest vs CatBoost	172 / 1,097	672.79	< 0.001	0.002639 (0.002439, 0.002838)	6.38
Random Forest vs XGBoost	127 / 2,355	1,998.20	< 0.001	0.006355 (0.006077, 0.006634)	18.54
CatBoost vs XGBoost	391 / 1,694	813.05	< 0.001	0.003717 (0.003462, 0.003972)	4.33

Note: n01 counts the test events that model A classified incorrectly and model B correctly, and n10 the reverse; the two together are the discordant events on which the test is based. The chi-square statistic uses the Edwards continuity correction, and the six p-values are adjusted by the Holm-Bonferroni procedure. The odds ratio is n10 divided by n01 and exceeds 1 when model A is the stronger of the pair. All comparisons use the same 350,568 test events.

A grouping pattern emerges. Extra Trees and Random Forest cannot be separated: they disagree on only 46 of 350,568 events, the accuracy difference is 0.000034 with a confidence interval spanning zero, and the adjusted

p-value of 0.1048 does not reach significance. Every comparison crossing from the bagging group to the boosting group is significant at p below 0.001, with odds ratios placing the bagging model between six and twenty times more likely to be correct on the events where the pair disagrees, and CatBoost separates significantly from XGBoost in turn. The indicative and the confirmatory procedures therefore agree on the ordering, but only the McNemar test supports a claim of significance, and its effect sizes show that the separation, although statistically certain at this sample size, is small in absolute terms: the largest accuracy gap between any two algorithms is 0.0064.

DISCUSSIONS

The ablation results carry the principal implication of this study. Near-perfect performance on the full feature set does not measure the ability to recognize attack behavior; much of it stems from memorizing attack tool identities, which does not generalize because an attacker can evade detection simply by renaming a tool binary. The benchmark reported an F1-Score of 99.41% and ROC AUC of 99.84% (Smiliotopoulos et al., 2023), values falling within the range of scenarios A and B here. This study does not conclude that the reference value is erroneous, but rather that its interpretation requires caution, since the dataset admits a shortcut that a model will exploit if permitted. The divergence between the two attack classes supports this reading: EoRS involves exploitation of network services and so leaves traces in connection properties, whereas EoHT involves credential theft and is recognized in this dataset more through tool identity than through behavior. Explainable AI proved essential to uncovering this, functioning as a diagnostic instrument rather than a transparency mechanism alone. This differs from prior applications, where interpretation was used chiefly to build analyst trust (Gaspar et al., 2024; Mahmoud et al., 2025; Wali et al., 2025).

Interpretation reliability deserves comment. Measured over ninety stratified events rather than one, agreement between SHAP and LIME proved partial rather than high: the two methods shared roughly half of their ten highest ranked features, agreed on the single leading feature in about a third of events, and reached a mean rank correlation of 0.38. This is consistent with their differing foundations, since SHAP computes globally consistent Shapley contributions while LIME approximates a linear model in a local neighborhood, so divergence among secondary features is expected. Two observations carry practical weight. Agreement is lowest precisely where the model is wrong, falling to 3.33 of ten shared features on misclassified EoHT events, so disagreement between the two explanations is itself a signal that a prediction deserves review. And because agreement on the leading feature is far from universal, an explanation drawn from one method alone should not be treated as the account of a decision, which argues for using more than one method, echoing evaluations of Explainable AI frameworks in other intrusion detection settings (Arreche et al., 2024).

The statistical picture must be read against its own limits. Across cross-validation folds the absolute Macro F1 gap between best and worst algorithm is only 0.0101, and on the held-out test set it is 0.0390, both small at the aggregate level. Per-class analysis tells a different story: EoHT precision is 0.7345 for Extra Trees against 0.5802 for XGBoost, a gap of 15 percentage points on the class that matters most for detection. Aggregate metrics can therefore conceal differences precisely where they are operationally decisive.

Inference measurements reveal a trade-off with practical consequences. Extra Trees is the most accurate but the slowest, while CatBoost is marginally less accurate yet nearly ten times faster, a difference arising from their prediction mechanisms, since Extra Trees aggregates many randomized trees while CatBoost uses compact symmetric trees. A medium-scale enterprise network generates roughly 1,000 to 10,000 Sysmon events per second, and even the slowest model here exceeds that upper bound threefold, so the operational definition of event-level detection is satisfied and model choice may be driven by deployment priorities rather than by feasibility. These findings complement the benchmark, which did not report inference cost.

Two design choices bound the scope of these conclusions. First, the stratified 80:20 split assigns individual events at random, so events originating from the same host, session, attack campaign, or tool may appear in both partitions. Performance reported here therefore reflects generalization to unseen events within a known environment rather than to unseen hosts or novel attack campaigns. This caveat compounds the central finding: a model that memorizes tool identity will be further flattered by a split that places sibling events on both sides of it, and a group-wise split by host, session, tool, or campaign is required to separate the two effects. Second, the four algorithms were compared under an equal budget of 200 estimators without hyperparameter search, yet the decisive parameters differ in kind across these families, covering depth, learning rate, subsampling, and regularization. The ranking reported here is therefore a statement about these four configurations, not about the algorithms in general, and a tuned comparison could reorder them. Both limitations bear directly on the first research question and are the priority for subsequent work.

CONCLUSION

This study compared four tree-based algorithms for event-level lateral movement detection in Sysmon logs within an Active Directory environment and integrated SHAP and LIME for instance-level interpretation. On 35

behavioral features Extra Trees and Random Forest performed best with a Macro F1 near 0.937, the McNemar test on paired test-set predictions placed the bagging group significantly above the boosting group while finding no significant separation between Extra Trees and Random Forest, and global interpretation showed that the behavior-based model grounds its decisions in network connection properties and process access indicators that analysts can examine. The central result came from the ablation study, which demonstrated that near-perfect performance of 0.9998 on the full feature set stemmed largely from memorizing attack tool identities, since EoHT precision fell to 0.7343 once binary-identity features were removed. Explainable AI thus served a dual role, improving transparency while exposing the true source of performance. All algorithms exceeded the throughput required for event-level stream processing.

A practical recommendation follows from the command-line diagnostics. Where Sysmon captures only connection and process-access events, as in this dataset, security-semantic markers activate on fewer than 2% of events and their extraction cost is not repaid, so practitioners should rely on connection and process-access indicators instead. Whether this recommendation reverses in environments that log process creation events with full argument capture remains an untested hypothesis: such markers are conceptually robust to tool renaming, but the present dataset provides no evidence on their activation rate or utility under that logging configuration, and testing it requires a corpus with complete process attributes. The decision should rest on the measured activation rate in the practitioner's own log corpus rather than on conceptual appeal.

Several limitations qualify these findings. The work rests on a single public dataset and has not been validated elsewhere. Results derive from a single stratified split reported without confidence intervals, although fold scores indicate low variance. The Friedman test was applied to folds from one dataset rather than to independent datasets and is reported as indicative only, with the McNemar test on paired test-set predictions carrying the confirmatory role; precision on the EoHT class remains near 0.73 on the best model. Future work should pursue cross-dataset validation, per-algorithm hyperparameter tuning, datasets with more complete process attributes, decision threshold tuning or cost-sensitive learning to raise minority class precision, verification on production systems, and extension of the ablation study to all four algorithms.

ACKNOWLEDGMENT

The authors thank the Department of Information Technology, Pradita University, for the academic support provided during this research.

REFERENCES

- Alabbadi, A., & Bajaber, F. (2025). An intrusion detection system over the IoT data streams using eXplainable Artificial Intelligence (XAI). *Sensors*, 25(3), 847. <https://doi.org/10.3390/s25030847>
- Arreche, O., Guntur, T. R., Roberts, J. W., & Abdallah, M. (2024). E-XAI: Evaluating black-box explainable AI frameworks for network intrusion detection. *IEEE Access*, 12, 23954–23988. <https://doi.org/10.1109/ACCESS.2024.3364526>
- Bai, T., Bian, H., Salahuddin, M. A., Abou Daya, A., Limam, N., & Boutaba, R. (2021). RDP-based lateral movement detection using machine learning. *Computer Communications*, 165, 9–19. <https://doi.org/10.1016/j.comcom.2020.10.013>
- Barredo Arrieta, A., Díaz-Rodríguez, N., Del Ser, J., Bennetot, A., Tabik, S., Barbado, A., ... Herrera, F. (2020). Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI. *Information Fusion*, 58, 82–115. <https://doi.org/10.1016/j.inffus.2019.12.012>
- Bin Hulayyil, S., Li, S., & Saxena, N. (2025). Explainable AI-based intrusion detection in IoT systems. *Internet of Things*, 31, 101589. <https://doi.org/10.1016/j.iot.2025.101589>
- Bin Sarhan, B., & Altwaijry, N. (2023). Insider threat detection using machine learning approach. *Applied Sciences*, 13(1), 259. <https://doi.org/10.3390/app13010259>
- Breiman, L. (2001). Random forests. *Machine Learning*, 45(1), 5–32. <https://doi.org/10.1023/A:1010933404324>
- Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. <https://doi.org/10.1145/2939672.2939785>
- Demšar, J. (2006). Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7, 1–30.
- Gaspar, D., Silva, P., & Silva, C. (2024). Explainable AI for intrusion detection systems: LIME and SHAP applicability on multi-layer perceptron. *IEEE Access*, 12, 30164–30175. <https://doi.org/10.1109/ACCESS.2024.3368987>
- Geurts, P., Ernst, D., & Wehenkel, L. (2006). Extremely randomized trees. *Machine Learning*, 63(1), 3–42. <https://doi.org/10.1007/s10994-006-6226-1>
- Herranz-Oliveros, D., Tejedor-Romero, M., Gimenez-Guzman, J. M., & Cruz-Piris, L. (2024). Unsupervised learning for lateral-movement-based threat mitigation in Active Directory attack graphs. *Electronics*, 13(19),

3944. <https://doi.org/10.3390/electronics13193944>
- Jin, D., Lu, Y., Qin, J., Cheng, Z., & Mao, Z. (2020). SwiftIDS: Real-time intrusion detection system based on LightGBM and parallel intrusion detection mechanism. *Computers & Security*, 97, 101984. <https://doi.org/10.1016/j.cose.2020.101984>
- Kalakoti, R., Nömm, S., & Bahsi, H. (2025). Federated learning of explainable AI (FedXAI) for deep learning-based intrusion detection in IoT networks. *Computer Networks*, 270, 111479. <https://doi.org/10.1016/j.comnet.2025.111479>
- Le, V.-H., & Zhang, H. (2022). Log-based anomaly detection with deep learning: How far are we? *Proceedings of the 44th International Conference on Software Engineering*, 1356–1367. <https://doi.org/10.1145/3510003.3510155>
- Lundberg, S. M., Erion, G., Chen, H., DeGrave, A., Prutkin, J. M., Nair, B., ... Lee, S.-I. (2020). From local explanations to global understanding with explainable AI for trees. *Nature Machine Intelligence*, 2(1), 56–67. <https://doi.org/10.1038/s42256-019-0138-9>
- Lundberg, S. M., & Lee, S.-I. (2017). A unified approach to interpreting model predictions. *Advances in Neural Information Processing Systems*, 30, 4765–4774.
- Mahmoud, M. M., Youssef, Y. O., & Abdel-Hamid, A. A. (2025). XI2S-IDS: An explainable intelligent 2-stage intrusion detection system. *Future Internet*, 17(1), 25. <https://doi.org/10.3390/fi17010025>
- Mallampati, S. B., & Seetha, H. (2024). Enhancing intrusion detection with explainable AI: A transparent approach to network security. *Cybernetics and Information Technologies*, 24(1), 98–117. <https://doi.org/10.2478/cait-2024-0006>
- Milajerdi, S. M., Gjomemo, R., Eshete, B., Sekar, R., & Venkatakrishnan, V. N. (2019). HOLMES: Real-time APT detection through correlation of suspicious information flows. *Proceedings of the IEEE Symposium on Security and Privacy*, 1137–1152. <https://doi.org/10.1109/SP.2019.00026>
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12, 2825–2830.
- Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: Unbiased boosting with categorical features. *Advances in Neural Information Processing Systems*, 31, 6638–6648.
- Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). Why should I trust you? Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. <https://doi.org/10.1145/2939672.2939778>
- Ring, M., Schlör, D., Wunderlich, S., Landes, D., & Hotho, A. (2021). Malware detection on Windows audit logs using LSTMs. *Computers & Security*, 109, 102389. <https://doi.org/10.1016/j.cose.2021.102389>
- Smiliotopoulos, C., & Kambourakis, G. (2026). Machine learning for lateral movement detection using Sysmon logs: An empirical comparison of imbalanced and resampled data. *International Journal of Information Security*, 25(1), 38. <https://doi.org/10.1007/s10207-025-01182-1>
- Smiliotopoulos, C., Kambourakis, G., & Barbatsalou, K. (2023). On the detection of lateral movement through supervised machine learning and an open-source tool to create turnkey datasets from Sysmon logs. *International Journal of Information Security*, 22(6), 1893–1919. <https://doi.org/10.1007/s10207-023-00725-8>
- Smiliotopoulos, C., Kambourakis, G., & Kolias, C. (2024). Detecting lateral movement: A systematic survey. *Heliyon*, 10(4), e26317. <https://doi.org/10.1016/j.heliyon.2024.e26317>
- Wali, S., Farrukh, Y. A., & Khan, I. (2025). Explainable AI and Random Forest based reliable intrusion detection system. *Computers & Security*, 157, 104542. <https://doi.org/10.1016/j.cose.2025.104542>