Evaluating ML Performance in EDR and XDR Systems Against Common Cyber Threats

Ryan Ginsburg, Matthew Liu, Felipe Marin, Rio Williams
Computer Science for Cybersecurity
August 8, 2025

Abstract— **The rising sophistication of modern cyberattacks creates a growing demand for companies to be up-to-date in their online and network security. This study aims to compare the Endpoint Detection and Response (EDR) and Extended Detection and Response (XDR) systems empowered by machine learning. Using 58 days of labeled telemetry from the Los Alamos National Laboratory (LANL) Comprehensive Multi-Source Cyber-Security Events dataset to build two parallel datasets: an EDR view comprising authentication and process logs, and an XDR view that expands on the EDR telemetry with DNS queries and network flow records. Both datasets were segmented into 10-second windows, engineered into statistical features, and chronologically split into training and testing partitions based on red team activity. We evaluated logistic regression, ensemble tree methods (Random Forest with SMOTE, Balanced Random Forest, Light GBM), a linear SVM, and an unsupervised Isolation Forest under standardized preprocessing and threshold tuning targeting 80 percent recall. Balanced Random forest emerges as the most robust detector, achieving a ROC-AUC of approximately 0.85 and a recall of 0.44 in both EDR and XDR contexts, while Light GMB on the XDR dataset delivered a record recall of .77 at the cost of a higher false-positive rate. These results demonstrate that incorporating cross-domain telemetry with XDR substantially improves detection coverage, particularly for gradient-boosted models, and underscore the growing importance of XDR for effective, proactive cyber defense.**

I. Introduction

Detecting modern cyberattacks has become increasingly difficult due to the growing complexity of adversarial techniques. Attackers now employ stealthy execution, lateral movement, and evasion strategies that are designed to bypass conventional defenses. To address these evolving threats, cybersecurity teams rely on detection architectures such as Endpoint Detection and Response (EDR) and Extended Detection and Response (XDR). EDR systems monitor host-level activity, including authentication events and process execution. XDR systems expand this scope by aggregating telemetry from multiple sources such as DNS queries, network flows, and cloud services, providing broader visibility into potentially coordinated attacks.

Although XDR platforms are often marketed as offering enhanced threat detection through multi-domain correlation, limited independent research has evaluated their effectiveness relative to traditional EDR systems under matched conditions. Most performance claims come from vendors, and few studies quantify whether the increased visibility of XDR translates into improved detection outcomes. For security professionals and system architects, the choice between EDR and XDR involves significant tradeoffs. While XDR may provide deeper insight into attacker behavior, it also introduces higher infrastructure costs, increased alert volumes, and greater deployment complexity. EDR systems, in contrast, are generally more lightweight and easier to manage but may lack the necessary context to detect threats that span beyond the endpoint.

To examine these tradeoffs, two detection environments were constructed using the Los Alamos National Laboratory (LANL) Comprehensive Multi-Source Cyber-Security Events dataset. The first simulates an EDR system by including only authentication and process logs. The second simulates an XDR system by incorporating additional DNS and network flow telemetry. Both datasets share the same time windows, labels, and red team attack timeline to enable a consistent and fair evaluation.

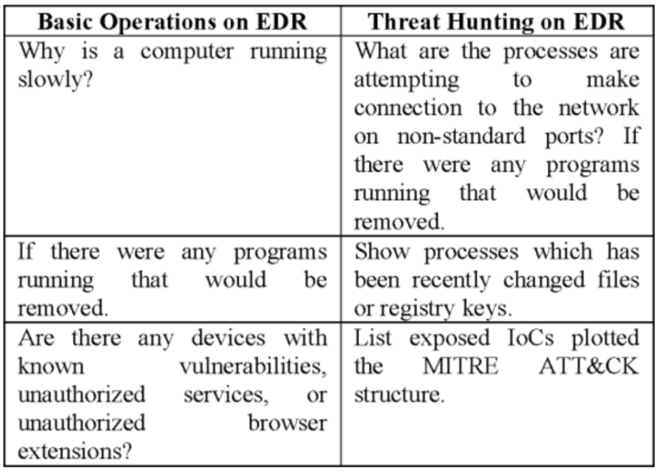
A range of supervised machine learning models was applied to both datasets, including logistic regression, tree-based classifiers, support vector machines, and an unsupervised anomaly detection baseline. Models were trained and evaluated using the same preprocessing steps, standardized feature sets, and recall-optimized thresholds. Performance was measured using accuracy, precision, recall, F1-score, ROC-AUC, and PR-AUC. This experimental design isolates the impact of telemetry scope on detection performance and provides practical insight into how visibility, model choice, and data context interact in real-world detection systems.

A. *Definitions*

Introduced in 2013, Endpoint Detection and Response (EDR) is a layer of protection that analyzes endpoint and network data to generate feedback from encountered threats. On a single platform, it can constantly monitor and integrate prevention, investigation, detection, and response. EDR is multifaceted due to the fusion of User and Entity Behavior Analytics (UEBA), Security Orchestration, Automation, and Response (SOAR) into a single platform.

EDR specifically uses pattern matching and signature-based detection to detect malware. A file's signature is unique to it, making it similar to a fingerprint. Once malicious signatures are found, EDR removes those specific signatures. To speed up investigations, it collects data to reveal the underlying reason and chronology of the alerts.

Afterwards, it uses the network, endpoint, and third-party data to respond accordingly to the threat.



Basic Operations on EDR	Threat Hunting on EDR
Why is a computer running slowly?	What are the processes are attempting to make connection to the network on non-standard ports? If there were any programs running that would be removed.
If there were any programs running that would be removed.	Show processes which has been recently changed files or registry keys.
Are there any devices with known vulnerabilities, unauthorized services, or unauthorized browser extensions?	List exposed IoCs plotted the MITRE ATT&CK structure.

Fig. 1. Use cases for EDR

Figure 1 contains examples of commands used by SOC, which can be easily translated into commands due to Natural Language Processing (NLP) in EDR. EDR monitors session traffic for malware propagation and dubious Application Programming Interface (API) call sequences. Decision making is facilitated by aggregating all session-based traffic data at a single endpoint. EDR was created as a user-friendly tool for cybersecurity experts due to its NLP capabilities. NPL gives EDR the ability to translate information from written text and figure out its intentions. This given ability can be used for asking for specific information within a network, shown in Figure 1, and filtering emails for signs of social engineering, further preventing possible breaches.

XDR is a security platform that processes data from various infrastructures; this includes endpoints, networks, applications, and cloud infrastructure. XDR offers precision in cybersecurity, allowing users to detect, analyze, and remediate threats through modern methods. The platform builds upon pre-existing solutions to develop an automated detection and response system.

In the processing of logs in evaluation of cyberattacks, XDR follows through a sequence of 3 layers: layer one consists of data collection; layer two consists of analytics and data normalization; layer three consists of automation, in which XDR proceeds to address the threat itself by taking proper measures.

Figure 2 displays a general outline of XDR's conceptual architecture. In the first layer, XDR collects data from sources throughout the network, including EDR, which monitors the endpoints; Email/Web Gateway, which scans emails and web traffic; CASB, which protects SaaS/cloud applications; IAM, which manages user identities, authentications, and access control; DLP, which generates records of sensitive data movements; and Firewall/NTA/IPS, which supplies network traffic logs, intrusion alerts, and metadata. This comprehensive data collection forms the foundation for advanced analysis and threat detection in subsequent layers.

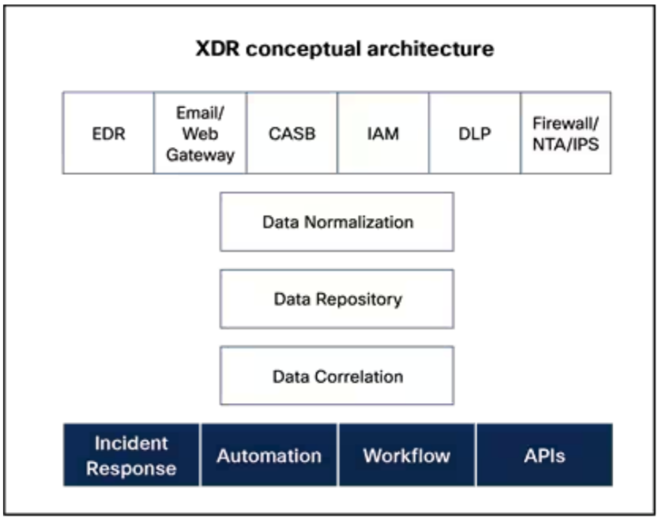


Fig. 2. XDR Conceptual Architecture

The second layer of XDR allows data processing and may be referred to as the 'heart' of XDR. XDR will initialize its data processing by normalizing the data, converting data from the sources in the first layer into a standard, consistent format. Next, the data will enter a data lake, a centralized repository to store normalized data on a scale, and later be analyzed to detect cyber attacks. XDR will then correlate the data by linking events across data sources to identify complex attack patterns; using ML, this data will be analyzed through the process of cleaning, enrichment, and indexing processes, producing new data, including Source IP Address, Destination IP Address, time data, location data, etc. This stage of XDR's second layer is heavily ML-driven, utilizing anomaly detection, behavioral analytics, and threat scoring models. Ultimately, this layer transforms disparate, unorganized logs into structured data, providing the analytical foundation for precise detection and automated response in the third layer of XDR.

Finally, XDR's third layer focuses on response and integration, enabling automated actions and collaboration with external systems. When a threat is detected, XDR initiates its incident response process; the incident response process isolates any affected endpoints or areas within the network through predefined automation rules. After XDR begins its incident response process, a workflow ticket is generated to track the incident, assign it to analysts, and document the response steps. Furthermore, XDR exposes APIs to integrate with external platforms, including Splunk, a Security Information and Event Management (SIEM) tool for log analysis, and ServiceNow, a cloud-based IT Service Management (ITSM) platform for managing security incidents. In parallel, XDR uses threat intelligence to correlate detected indicators with known attack patterns, enrich the contextual understanding of the incident, and identify related threats across the environment. This third layer process allows XDR to deliver a faster, more coordinated, and intelligence-driven response.

II. Datasets and Preparation

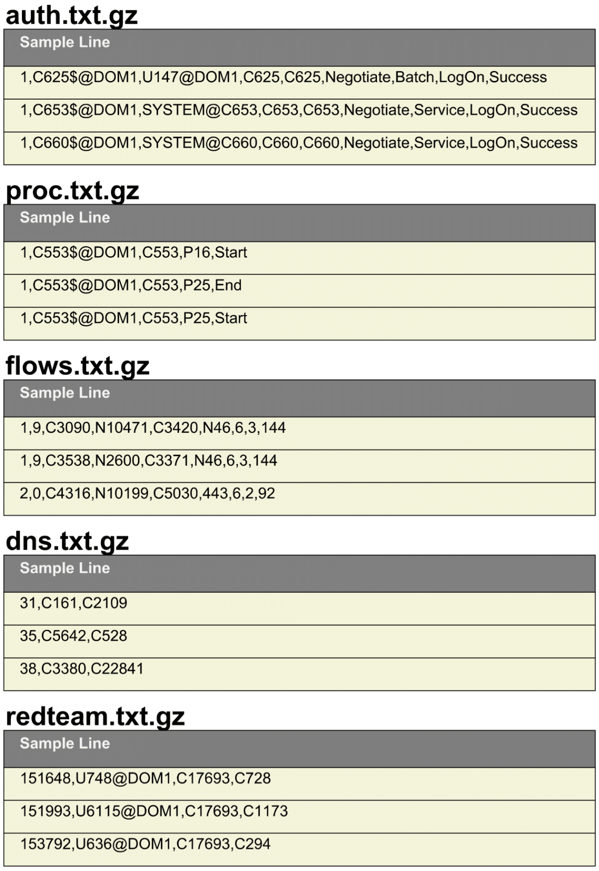
The purposeose of this study was to identify and preparedata set that could supportt a fair and controlled comparisonbetween then the EDR (Endpoint Detection and Response) and XDR (Extended Detection and Response) systems. We selected the LANL Comprehensive Multi-Source Cyber-Security Events dataset because it includes both host-based and network-based telemetry collected during real-world red team attacks. This unique combination made it possible to construct two parallel datasets: one that captures EDR visibility using only authentication and process logs, and another that reflects XDR visibility by adding DNS and network flow data. Both datasets were built using the exact same time windows, user activity, and labeled attack events. By keeping the labels and sample timing identical and varying only the available telemetry, we created a consistent and realistic testbed for evaluating how the broader visibility of XDR systems influences model performance compared to EDR systems.

A. Datasets Considered and Excluded

Several prominent cybersecurity datasets were evaluated, but ultimately excluded due to key limitations. The Linux-APT-Dataset-2024 contains telemetry from Linux systems under simulated attack scenarios, but none of the provided CSV files included any labeled malicious instances, making supervised learning infeasible. The DARPA OpTC dataset features detailed, realistic telemetry and red team activity across an enterprise Windows environment, but its massive file sizes, often tens or hundreds of gigabytes, exceeded the processing capabilities of the available local hardware. The TON_IoT datasets span Windows, Linux, IoT, and network domains with labeled attacks, but the components are isolated and lack a unified timeline or shared identifiers, preventing the creation of cross-domain samples required for XDR. CIC-MalMem-2022 is limited to Windows process and memory data with no network or DNS visibility, making it suitable only for EDR modeling. The CSE-CIC-IDS2018 dataset provides labeled network traffic but lacks host-based telemetry like authentication or process logs, while CIC-IDS2017, an earlier version, suffers from the same issue and includes only synthetic flow data. As a result, none of these datasets met the criteria needed to build both endpoint- and network-visible datasets for comparative EDR and XDR evaluation.

B. Dataset Used

This study uses the Los Alamos National Laboratory (LANL) Comprehensive Multi-Source Cyber-Security Events dataset as the foundation for constructing and comparing EDR and XDR machine learning models. Released in 2015, the dataset consists of 58 consecutive days of telemetry collected from an operational Windows enterprise network. It is one of the few publicly available datasets to combine real user and system activity with detailed red team attack annotations, making it well-suited for supervised learning in the cybersecurity domain. The dataset captures a wide range of events, including authentication activity, process execution, DNS queries, network flows, and confirmed adversarial operations, offering the necessary telemetry for constructing parallel EDR and XDR datasets. The dataset reflects activity from 12,425 unique user identifiers, 17,684 distinct computers, and 62,974 unique processes, underscoring the scale and complexity of the environment.



auth.txt.gz

Sample Line
1,C625$@DOM1,U147@DOM1,C625,C625,Negotiate,Batch,LogOn,Success
1,C653$@DOM1,SYSTEM@C653,C653,C653,Negotiate,Service,LogOn,Success
1,C660$@DOM1,SYSTEM@C660,C660,C660,Negotiate,Service,LogOn,Success

proc.txt.gz

Sample Line
1,C553$@DOM1,C553,P16,Start
1,C553$@DOM1,C553,P25,End
1,C553$@DOM1,C553,P25,Start

flows.txt.gz

Sample Line
1,9,C3090,N10471,C3420,N46,6,3,144
1,9,C3538,N2600,C3371,N46,6,3,144
2,0,C4316,N10199,C5030,443,6,2,92

dns.txt.gz

Sample Line
31,C161,C2109
35,C5642,C528
38,C3380,C22841

redteam.txt.gz

Sample Line
151648,U748@DOM1,C17693,C728
151993,U6115@DOM1,C17693,C1173
153792,U636@DOM1,C17693,C294

Fig. 3. LANL Cyber-Security Events Dataset raw file samples

The raw data is spread across five compressed text files, each capturing a specific facet of host or network behavior. The auth.txt.gz file contains authentication logs, where each event records the time, source and destination users and computers, the authentication type (e.g., NTLM, Kerberos), logon type (e.g., interactive, remote, batch), logon orientation (initiator or target), and a binary success or failure result. These records provide critical insight into access behavior, including potential misuse of credentials or lateral movement. The proc.txt.gz file documents process creation and termination events, including the user and computer responsible, the process name, a start/stop indicator, and the event timestamp. These process logs allow for the identification of suspicious binaries, abuse of administrative tools, and unusual execution patterns. Together, these two files define the visibility scope of the EDR dataset. The dns.txt.gz file logs outbound DNS queries initiated from internal hosts, capturing the timestamp, querying host, and the domain name resolved. This data enables the detection of domain generation algorithms, and communication with suspicious external infrastructure. The flows.txt.gz file provides network flow records, including timestamps, source and destination IPs and ports, protocol, packet and byte counts, and connection duration. These flows summarize host-to-host communications and can reveal anomalies such as port scanning, excessive data transfer, or peer-to-peer command-and-control behavior. These two files contribute exclusively to the XDR dataset by expanding detection visibility beyond endpoints. Finally, the redteam.txt.gz file provides labeled attack events generated by red team activity. It includes timestamps, user and host identifiers, and target information. While this file does not contribute any features, it serves as the source of ground-truth labels used during supervised training and evaluation.

C. *Preperation*

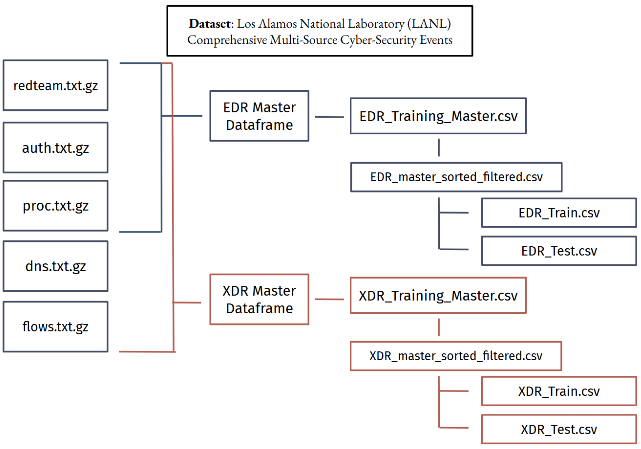


Fig. 4. File processing pipeline for generating EDR and XDR training and test datasets

To transform the raw logs into structured, machine learning–ready datasets, each file is parsed into structured DataFrames with type normalization and identifier standardization. Events are grouped into fixed, non-overlapping 10-second time windows, and feature aggregation is performed per window. Authentication features include counts of successful and failed logins, diversity of login targets, and entropy of accessed hosts. Process-based features include the number of new processes, unique binary names, and process start/stop ratios. DNS features capture the number and entropy of domain queries, while network flow features include statistical summaries such as bytes per packet, connection frequency, port entropy, and directional traffic ratios. These features are computed consistently across windows for all applicable data sources. Each sample window is assigned a binary label using the red team attack file. If a red team event overlaps with a time window based on its timestamp, that window is labeled as malicious; otherwise, it is labeled benign. Labels are aligned across both datasets to ensure that EDR and XDR samples differ only in the available feature sets, not in the labeling process. The EDR dataset includes only features derived from authentication and process logs, while the XDR dataset appends features from DNS and network flow telemetry. This design enables a controlled comparison of model performance under distinct visibility regimes. After feature construction and labeling, the dataset undergoes temporal filtering. Only days containing known red team activity are retained, reducing dataset size while focusing evaluation on periods of interest. Within these filtered days, samples are divided into fixed-size chunks to facilitate memory-efficient processing. Each chunk is sorted chronologically by day and timestamp using an external merge sort algorithm, and all chunks are then merged into a single, temporally ordered dataset. This strict ordering is essential to maintain the integrity of time-dependent learning algorithms and to prevent temporal leakage across train and test boundaries. Dataset splitting is also performed chronologically by day. Approximately 80% of the red team–active days are allocated to the training set, while the remaining 20% are reserved for evaluation. The day field in each sample determines its assignment. This split is executed through a single pass over the sorted dataset, ensuring that records are streamed into their respective partitions without excessive memory use. The design preserves realistic deployment conditions by ensuring that models are trained on past data and tested exclusively on future, unseen attack windows. No overlap exists between red team events in the training and testing partitions, and all events maintain temporal consistency. The original training dataset consists of 30,082,580 samples, while the test dataset contains 8,651,372 samples. Each sample represents a 10-second window of system activity derived from the LANL telemetry. However, the overall class distribution is highly imbalanced, with malicious samples accounting for just 0.0028% of the total. While this distribution reflects the real-world rarity of intrusions, it posed significant challenges for supervised machine learning models, particularly in detecting rare attack windows. Models trained on the original dataset tended to overfit to the dominant benign class, achieving deceptively high accuracy while failing to recall most malicious events. To address this issue, we constructed a more balanced and focused subset of the original dataset using a context-based sampling approach. This technique preserved all malicious samples and selectively included benign samples located near each malicious window. Specifically, for every malicious event, the algorithm sampled between 75 and 150 benign windows both before and after the event, within a windowed range of up to 150 samples. This sampling ensured that the model was exposed to a richer, temporally relevant set of benign behaviors surrounding known attacks. Therefore, helping it learn the subtle transitions that might precede or follow malicious actions. Importantly, this sampling was performed once on the EDR dataset to generate a set of selected indices. These same indices were then applied to the XDR dataset, guaranteeing that both EDR and XDR versions represented the exact same time windows with identical labels, differing only in the types of telemetry features available. This design enables a fair, side-by-side evaluation of EDR and XDR model performance.

D. Final Dataset Characteristics

The resulting datasets each contain 185,442 training samples and 16,287 test samples. The class distribution improves to 857 malicious and 184,585 benign training samples, yielding a more learnable imbalance ratio of 0.4643%. The EDR dataset includes 20 total columns, of which 18 are numerical features, while the XDR dataset includes 34 columns, with 32 numerical features, reflecting the expanded visibility provided by DNS and network telemetry. This adjustment maintains the original data's temporal integrity, avoids synthetic oversampling techniques, and offers a more meaningful framework for evaluating model effectiveness under constrained visibility conditions. By integrating heterogeneous telemetry sources, preserving strict temporal constraints, and applying consistent labeling, this data processing pipeline produces two large-scale cybersecurity datasets optimized for evaluating and comparing EDR and XDR detection models under realistic enterprise conditions.

III. Methodology

A. Features and Preprocessing Steps

To enable a rigorous comparative analysis between Endpoint Detection and Response (EDR) and Extended Detection and Response (XDR) systems, specific criteria were established for feature inclusion and exclusion. The EDR dataset exclusively incorporated telemetry data from endpoint sources, namely authentication and process execution logs. Conversely, the XDR dataset expanded upon the EDR features by integrating additional network-centric data, including DNS queries and network flow telemetry, providing a broader visibility scope.

Both datasets underwent careful preprocessing to ensure model generalization and to prevent data leakage. Specifically, certain columns were explicitly excluded from the feature sets. Host identifiers ("src_comp", "dst_comp", "computer", and "host") were omitted to prevent models from overfitting to specific hosts or devices. Likewise, the target variable (label) was excluded to maintain the integrity of predictive analysis.

Data preprocessing also addressed potential issues with data quality. Infinite and missing (NaN) values were systematically replaced with median values computed solely from the training dataset, preventing contamination of the evaluation phase by future data. Feature scaling was uniformly applied across all models using `StandardScaler`, which was fit exclusively to the training data to standardize each feature to zero mean and unit variance. The trained scaler was subsequently applied to the testing dataset without refitting, preserving consistent feature representation across training and evaluation phases. Labels indicating benign or malicious samples were numerically encoded using a `LabelEncoder`, again fitted exclusively on the training data labels to maintain consistency and prevent leakage.

B. Machine Learning Models and Rationale for Selection

A diverse set of supervised and unsupervised machine learning models were selected to provide comprehensive performance evaluations and to reflect varying approaches commonly applied in cybersecurity contexts.

Logistic Regression (LR) was employed as an interpretable baseline model. Its simplicity allows for straightforward evaluation of linear relationships between features and outcomes. Parameters such as balanced class weights (`class_weight=balanced`) and an extended maximum iteration limit (`max_iter=1000`) ensured stable convergence during training.

Two ensemble tree-based methods were chosen to robustly handle significant class imbalance: Random Forest coupled with Synthetic Minority Oversampling Technique (SMOTE) and Balanced Random Forest. The Random Forest classifier, enhanced with SMOTE, synthetically balanced class distribution by oversampling the minority class. Conversely, the Balanced Random Forest classifier naturally addressed imbalance by systematically undersampling the majority class during the learning process. Both ensemble methods were standardized by configuring identical hyperparameters, including the number of trees (`n_estimators=200`), unrestricted tree depth (`max_depth=None`), and consistent node splitting criteria (`min_samples_split=2`, `min_samples_leaf=1`). Additionally, the minority-to-majority sampling ratio was standardized to 0.5 across both ensemble methods, facilitating direct and equitable comparisons.

LightGBM, a gradient-boosting framework recognized for its high scalability and predictive performance on large-scale cybersecurity datasets, was also included. Hyperparameters for LightGBM were standardized across all experiments, featuring 200 estimators (`n_estimators=200`), automatic tree-depth management, and regularization through subsample and column-sample ratios set uniformly to 0.8. Class imbalance was specifically addressed through LightGBM's `scale_pos_weight` parameter, calculated directly from the training dataset's minority-to-majority class ratio.

A linear Support Vector Machine (SVM) implemented via Stochastic Gradient Descent (SGD) was selected for its computational efficiency, particularly relevant given the dataset's large scale. It featured balanced class weighting (`class_weight=balanced`), adaptive learning rate management (`learning_rate=optimal`), and a consistent regularization strength (`alpha=0.0001`).

Finally, Isolation Forest served as an unsupervised anomaly detection baseline, crucial for benchmarking supervised model performance. Its `contamination` parameter was explicitly set based on the minority-class proportion observed in the training dataset, realistically simulating expected anomaly frequencies. For consistency, Isolation Forest matched other ensemble methods with a uniform ensemble size of 200 trees (`n_estimators=200`).

C. *Evaluation Metrics and Model Interpretability*

Multiple evaluation metrics were employed to rigorously assess and compare model performance from complementary perspectives:

- **Accuracy**: (TP + TN) / All. A coarse baseline; in our data a naive "predict-benign" model already attains approximately 0.996, so high accuracy alone is not evidence of good detection.
- **Balanced Accuracy**: (TPR + TNR) / 2. Gives equal weight to each class and therefore offsets the 0.46% prevalence skew that inflates raw accuracy.
- **Precision**: TP / (TP + FP). Represents alert quality: the proportion of fired alerts that are genuinely malicious, and thus a proxy for analyst workload.
- **Recall**: TP / (TP + FN). Captures coverage: the fraction of all malicious windows the model catches (1 minus miss-rate).
- **F1-score**: Harmonic mean of Precision and Recall at the chosen operating threshold; provides a single threshold-specific trade-off between missed attacks and false alarms.
- **ROC-AUC**: Area under the Receiver-Operating-Characteristic curve (TPR vs. FPR). Summarizes discrimination across every threshold but can appear deceptively high when true negatives dominate.
- **PR-AUC**: Area under the Precision–Recall curve. Favored for rare-event detection because it ignores the abundance of true negatives and directly measures the model's ability to keep both precision and recall high.

Comprehensive model interpretability was facilitated through additional analytical visualizations and metrics. Confusion matrices provided explicit breakdowns of true positives, true negatives, false positives, and false negatives, clarifying detection trade-offs. ROC curves, Precision–Recall curves, and distributions of predicted probabilities were systematically analyzed across all models to visually interpret performance.

D. *Model Training, Parameter Standardization, and Threshold Tuning*

To ensure fairness and validity in comparative analysis, hyperparameters and training conditions were meticulously standardized across all models and datasets. All tree-based models such as Random Forest with SMOTE, Balanced Random Forest, and LightGBM used exactly 200 estimators (`n_estimators = 200`), unrestricted maximum depth, and standardized node-splitting parameters. A consistent random seed of 42 was applied across all models to ensure reproducibility. Class imbalance was systematically addressed through multiple standardized approaches: balanced class weights for linear models, SMOTE with 0.5 sampling strategy for Random Forest, calculated `pos_weight` ratios for gradient boosting models, and built-in balanced sampling for ensemble methods.

Feature preprocessing was uniformly applied across both EDR and XDR datasets, with infinite values replaced by NaN, missing values imputed using training set medians, and `StandardScaler` normalization fitted on training data and applied to test data. Crucially, no hyperparameter tuning or cross-validation was performed, preventing methodological bias that could favor one dataset over another. This direct train-to-test evaluation ensures that performance differences reflect genuine dataset characteristics rather than optimization advantages.

To maximize practical effectiveness in threat detection, models providing probabilistic predictions underwent standardized threshold optimization targeting exactly 80% recall for the malicious class across both EDR and XDR datasets. Using precision–recall curves, optimal thresholds were identified where recall was greater than or equal to 0.80, with safeguards to reject thresholds predicting more than 90% of samples as malicious (preventing trivial solutions). This standardized 80% recall target ensures consistent evaluation criteria and practical relevance in cybersecurity contexts where missing threats is more costly than false alarms.

All metrics were computed using identical formulas and evaluation procedures for both datasets. This comprehensive standardization methodology ensures that any observed performance differences between EDR and XDR approaches reflect genuine data characteristics rather than methodological advantages, supporting the validity of comparative conclusions.

IV. Results and Analysis

A. *Model-to-Model Performance on the EDR Dataset*

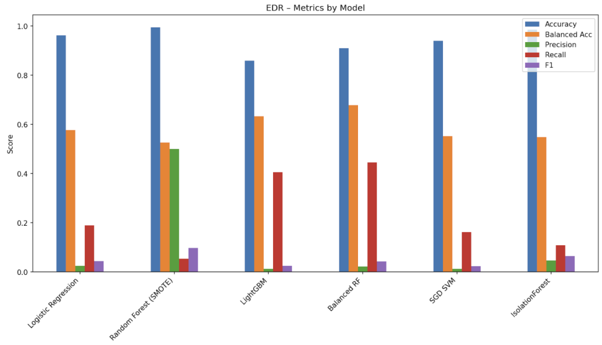


Fig. 5. EDR - Metrics by Model

TABLE I

EDR – Model Performance Metrics

Model	Accuracy	Balanced Acc	Precision	Recall	F1	ROC-AUC	PR-AUC
Logistic Regression	0.9626	0.5777	0.0249	0.1892	0.0440	0.6720	0.0265
Random Forest (SMOTE)	0.9955	0.5269	0.5000	0.0541	0.0976	0.6809	0.0728
LightGBM	0.8596	0.6335	0.0132	0.4054	0.0256	0.6343	0.0082
Balanced RF	0.9101	0.6791	0.0227	0.4459	0.0431	0.8530	0.0414
SGD SVM	0.9407	0.5532	0.0131	0.1622	0.0242	—	—
Isolation Forest	0.9859	0.5490	0.0465	0.1081	0.0650	—	—

Across the host-only feature set, model behaviour diverged sharply once minority-class metrics were considered. Balanced Random Forest (Bal-RF) demonstrated the most favourable trade-off between coverage and noise: its ROC-AUC reached 0.8530, the highest of the suite, and its recall

TABLE II

EDR – Confusion Matrix Analysis

Model	FP Rate	Miss Rate	TN	FP	FN	TP
Logistic Regression	3.39%	81.08%	15664	549	60	14
Random Forest (SMOTE)	0.02%	94.59%	16209	4	70	4
LightGBM	13.83%	59.46%	13970	2243	44	30
Balanced RF	8.78%	55.41%	14790	1423	41	33
SGD SVM	5.58%	83.78%	15309	904	62	12
Isolation Forest	1.01%	89.19%	16049	164	66	8

TABLE IV

XDR – Confusion Matrix Analysis

Model	FP Rate	Miss Rate	TN	FP	FN	TP
Logistic Regression	17.28%	50.00%	13411	2802	37	37
Random Forest (SMOTE)	0.02%	94.59%	16210	3	70	4
LightGBM	28.82%	22.97%	11540	4673	17	57
Balanced RF	7.52%	56.76%	14993	1220	42	32
SGD SVM	3.95%	79.73%	15572	641	59	15
Isolation Forest	0.18%	95.95%	16184	29	71	3

climbed to 0.4459, meaning it detected nearly 45% of all malicious ten-second windows while keeping the false-positive rate to 8.78%. LightGBM offered an alternative high-recall profile (Recall = 0.4054; Balanced Accuracy = 0.6335) but at the cost of overall accuracy (0.8596) because it aggressively flagged positives. Random Forest with SMOTE oversampling (RF + SMOTE) posted the highest precision (0.5000) and accuracy (0.9955) yet missed 95% of attacks (Recall = 0.0541). Linear baselines underscored how deceptive accuracy can be under imbalance: Logistic Regression achieved a nominal accuracy of 0.9626 yet recalled only 18.9% of attacks and produced a PR-AUC of 0.0265. SGD-SVM performed similarly (Recall = 0.1622). Finally, the unsupervised Isolation Forest attained respectable accuracy (0.9859) but its recall stalled at 0.1081, confirming that anomaly scores alone struggle to capture subtle host-level red-team behaviour.

B. Model-to-Model Performance on the XDR Dataset

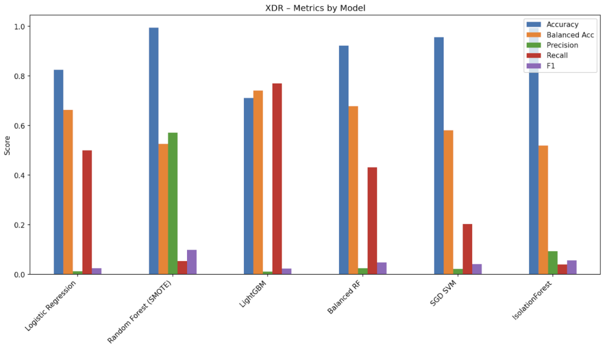


Fig. 6. XDR - Metrics by Model

TABLE III

XDR – Model Performance Metrics

Model	Accuracy	Balanced Acc	Precision	Recall	F1	ROC-AUC	PR-AUC
Logistic Regression	0.8257	0.6636	0.0130	0.5000	0.0254	0.6585	0.0235
Random Forest (SMOTE)	0.9955	0.5269	0.5714	0.0541	0.0988	0.6588	0.0680
LightGBM	0.7120	0.7410	0.0121	0.7703	0.0237	0.7560	0.0115
Balanced RF	0.9225	0.6786	0.0256	0.4324	0.0483	0.8520	0.0250
SGD SVM	0.9570	0.5816	0.0229	0.2027	0.0411	—	—
Isolation Forest	0.9939	0.5194	0.0938	0.0405	0.0566	—	—

Augmenting the feature space with network and DNS telemetry re-ordered the leaderboard. LightGBM now dominated detection, almost tripling its recall to 0.7703 and raising Balanced Accuracy to 0.7410, the best in either view. The gain slashed the miss-rate to 22.97%, though precision remained low (0.0121) and the false-positive rate spiked to 28.8%. Balanced RF retained the top ROC-AUC (0.8520) and lifted precision slightly to 0.0256 while sustaining a moderate recall of 0.4324, making it the only model that balanced every metric above the median. RF + SMOTE's precision improved further (0.5714) but its recall stagnated at 0.0541, pushing PR-AUC down from 0.0728 to 0.0680. Logistic Regression exploited the new linear signals to double its recall to 0.5000, albeit with a precision of just 0.0130. Linear SGD-SVM saw only a modest recall rise to 0.2027, while Isolation Forest suffered a recall drop to 0.0405, illustrating the difficulty of unsupervised methods in high-dimensional, noisy spaces.

C. Same-Model EDR to XDR Shifts

TABLE V

Percentage Change from EDR to XDR (Relative to EDR)

Model	Accuracy (%)	Balanced Acc (%)	Precision (%)	Recall (%)	F1 (%)
Logistic Regression	-14.21	14.86	-47.79	164.28	-42.27
Random Forest (SMOTE)	0.00	0.00	14.28	0.00	1.23
LightGBM	-17.16	16.96	-8.33	90.05	-7.42
Balanced Random Forest	1.36	-0.07	12.78	-3.03	12.05
SGD SVM	1.73	5.14	74.81	24.97	69.83
Isolation Forest	0.81	-5.39	101.72	-62.53	-12.92

LightGBM experienced the most dramatic lift, adding +0.3649 recall points and +0.1217 ROC-AUC, evidence that boosted trees exploit cross-domain correlations exceptionally well. Logistic Regression added +0.3108 recall but lost 0.0030 PR-AUC as false positives grew. Bal-RF proved remarkably stable: recall dipped only 0.0135 and ROC-AUC slipped by a mere 0.001, confirming its robustness irrespective of visibility scope. RF + SMOTE, in contrast, showed no recall gain and a PR-AUC drop of 0.0048, suggesting that synthetic minority instances do not scale to the richer XDR feature manifold. Isolation Forest's recall slid by 0.0676, corroborating its sensitivity to noise when labels are absent.

D. Recommended XDR Detection Model

For organisations with full XDR telemetry, Balanced Random Forest emerges as the most defensible single model. It offers high ranking ability (ROC-AUC $\approx$ 0.85) and an acceptable recall (0.4324) while containing the false-positive rate to 7.52%, levels unlikely to swamp a SOC with alerts. LightGBM may serve as an aggressive auxiliary detector where missing an intrusion is costlier than triaging extra alerts, but its 29% false-positive rate necessitates a downstream filter. In EDR-only deployments, Bal-RF is again preferred (ROC-AUC = 0.8530; Recall = 0.4459), outperforming all other host-centric strategies.

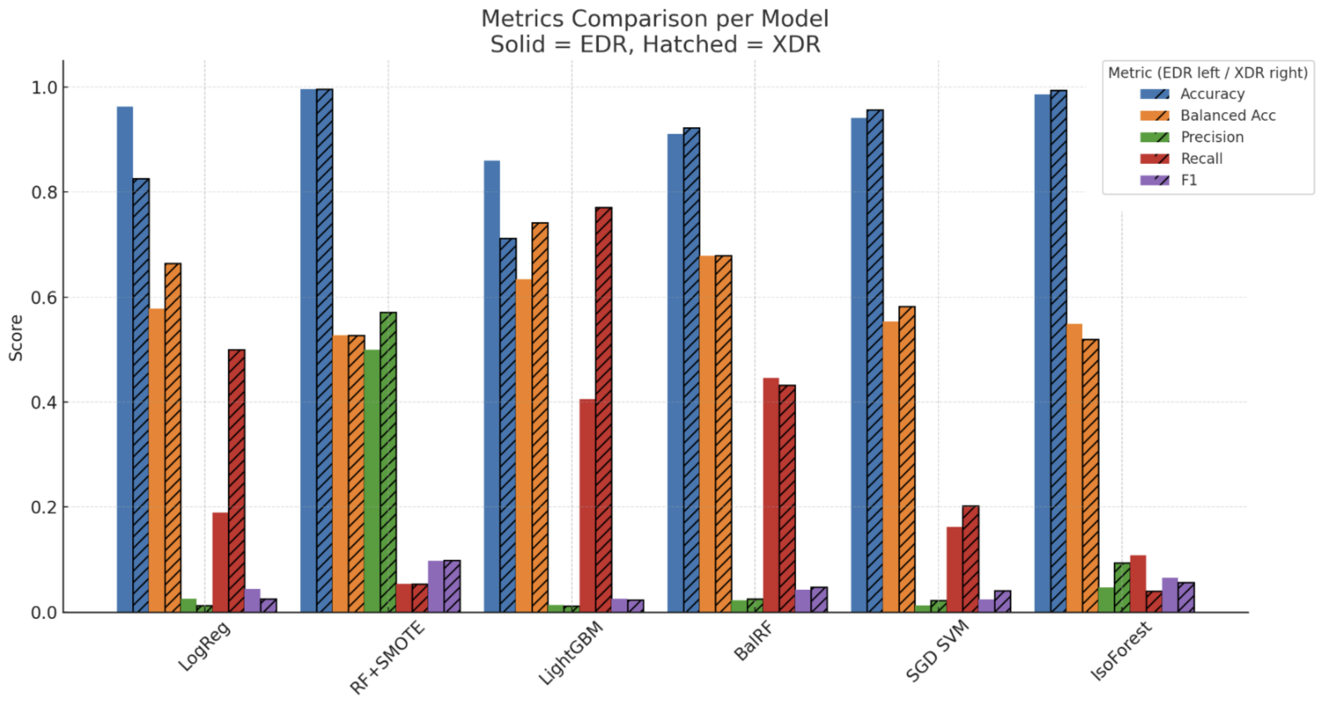


Fig. 7. EDR vs XDR Metrics Comparison

E. Recommended EDR Detection Model

In pure host-telemetry deployments, Balanced Random Forest stands out as the most dependable single model. It pairs the highest ranking power in the suite (ROC-AUC = 0.8530) with the strongest recall (0.4459), catching nearly 46% of malicious ten-second windows while limiting false positives to 8.78% and keeping overall accuracy at 0.9101. No other EDR model matches this balance: LightGBM strikes more aggressively but drives the false-positive rate above 13%, whereas RF + SMOTE attains eye-catching precision yet overlooks 95% of attacks. By undersampling during tree construction, Balanced RF maintains robust minority-class coverage without swamping analysts with alerts, making it the clear first choice for organisations restricted to host-only data streams.

F. Overall EDR vs. XDR System Benefit

XDR's added context translates into substantial recall gains for the most feature-hungry models (e.g., LightGBM, +32 percentage points), but the mean ROC-AUC across all models improves only marginally and PR-AUC often erodes. Thus, XDR chiefly helps when the mission prioritises minimising missed attacks over controlling analyst workload and infrastructure cost. Where storage or parsing budgets are tight, a balanced ensemble on pure-host data can still provide ROC-AUC ≈ 0.85 with fewer than one false alert per twelve benign windows.

G. Noteworthy Anomalies

LightGBM's leap from worst accuracy in EDR (0.8596) to best recall in XDR (0.7703) exemplifies how raw accuracy obscures minority-class performance. Conversely, RF + SMOTE consistently achieved the highest precision (> 0.50) while leaving recall below 6%, highlighting the risk of oversampling strategies that overfit easy negatives. Isolation Forest's precision nearly doubled (0.0465 to 0.0938) even as recall fell, implying it increasingly flags benign but unusual network behaviour rather than genuine threats.

H. Role of Class Imbalance

With the malicious class representing just 0.4643% of all windows, accuracy alone masks ineffectiveness: a majority-class predictor attains 0.9955 accuracy by default. Ensemble methods that embed balancing, such as Bal-RF via undersampling, maintain high ROC-AUC and the best recall, confirming their suitability for skewed security data. SMOTE-augmented forests raise precision but do not improve recall, showing that synthetic samples cannot replicate the diversity of genuine attacks. Linear models benefit only when additional XDR features make minority instances linearly separable.

V. Limitations and Threats to Validity

This study's conclusions must be interpreted in light of several inherent limitations. First, the LANL dataset dates to 2015, preceding the widespread adoption of modern EDR/XDR tooling and newer adversary techniques such as file-less "living-off-the-land" binaries, cloud-identity abuse, and SaaS lateral movement. Consequently, the threat behaviours captured here may under-represent today's attack surface. Second, all telemetry originates from a single Windows-only enterprise; the absence of Linux, macOS, mobile, operational-technology, and cloud artefacts limits the external validity of our findings for heterogeneous or cloud-first fleets. Third, the ground-truth file (`redteam.txt`) identifies the when and where of each malicious event but omits richer semantics, hindering fine-grained analysis of what behaviours the models actually learn to detect.

Moreover, the red-team exercise is static and non-adaptive: once detected, it does not pivot or evade, so model performance may be overly optimistic compared with dynamic, feedback-driven adversaries. Coverage is also incomplete: only 20 of the 58 days contain labelled intrusions, many during business hours, raising the possibility that time-of-day artefacts leak into the models. To mitigate extreme class imbalance, we raised the malicious prevalence from 0.0028% to 0.464% by context-sampling benign windows near attacks, but this prior shift almost certainly understates the false-positive volume a production deployment would face.

Our 10-second fixed aggregation windows, while computationally tractable, can fragment bursty exploits or smear slow command-and-control traffic, potentially muting sequence-aware signals. Finally, practical hardware limits forced us to exclude richer corpora such as DARPA OpTC, leaving open whether models might behave differently when trained on larger, more diverse telemetry. The "EDR" and "XDR" views here are only proxies for commercial platforms: the EDR feature set lacks kernel-level sensors, memory forensics, and real-time behavioural analytics, while the XDR variant is limited to DNS and NetFlow enrichments rather than the multi-platform, SaaS-aware, identity-centric telemetry that defines modern XDR products.

VI. CONCLUSIONS

This research demonstrates XDR's ability to outperform EDR in detecting cyberthreats. Due to XDR's wider scope of telemetry, with extended access to an entire network and DNS telemetry, the system is able to output better results across multiple metrics, including recall, ROC-AUC, and balanced accuracy. Although XDR showed great performances among LogReg, RF+SMOTE, LightGBM, BalRF, and SGD SVM, the system showed the most significant improvements in recall for the LightGMB and BalRF models. This demonstrates the value of correlated, cross-domain signals in uncovering advanced threats especially when pairing XDR with these five ML models. These results not only underscore how EDR's endpoint-only detection is limited in its ability to detect cyber threats, but also highlights XDR's growing significance in cybersecurity.

The results presented in this study offer the cyber industry, especially SOCs and security architects, with practical implications on the usage of XDR for cybersecurity. While EDR remains a viable approach for resource-constrained environments, its limited visibility scope can miss multi-layered or lateral movement attacks. In contrast, XDR's architecture provides a more comprehensive threat detection surface by integrating data throughout an entire network and DNS telemetry. Furthermore, by pairing XDR with ML techniques, XDR can provide SOCs and security architects with improved detection rates without overwhelming analysts with false positives

As modern cyberattacks continue to evolve and threaten the stability of multiple layers of an organization's infrastructure, XDR will only continue to grow in importance. EDR's narrow detection approach will only become increasingly more ineffective, and that alone should push the cyber industry to adopt XDR's multi-layer system. AI usage to enhance cyberattacks (e.g. polymorphic malware and automate reconnaissance) by adversaries are on the rise; as a result, security teams require equally adaptive defenses. Platforms must not only aggregate diverse telemetry, but also learn and respond dynamically, supporting XDR's relevancy especially when combined with machine learning.

Future research should focus on utilizing more diverse and recent datasets, including those that reflect SaaS platforms and identify-based attacks, to help ensure broader applicability. Moreover, future researchers may seek to explore detection architectures capable of processing streaming telemetry, adapting to evolving attacker behavior, and scaling across hybrid and cloud-native environments. Additionally, exploring emerging learning methods, such as graph neural networks or transformer-based classifiers, may unlock further improvements in detection coverage and model interpretability.

Ultimately, this work supports a growing consensus in the cybersecurity community. Defense must be proactive, data-driven, and tightly integrated across domains. XDR systems, especially when paired with machine learning, represent a powerful and necessary step toward building that future.

REFERENCES

[1] Kent, Alexander D. ”Comprehensive, Multi-Source Cyber-Security Events.” Los Alamos National Laboratory, 2015, https://csr.lanl.gov/data/cyber1/.

[2] Moustafa, Nour. ”TON_IoT Datasets.” *UNSW Research*, University of New South Wales, 2 June 2021, https://research.unsw.edu.au/projects/toniot-datasets.

[3] Karim, Sohaib. "Linux-APT-Dataset-2024." Mendeley Data, V1, 2024, doi:10.17632/5x68fv63sh.1. https://zenodo.org/records/10685642.

[4] Anjum, Md. Monowar, Shahrear Iqbal, and Benoit Hamelin. ”Analyzing the Usefulness of the DARPA OpTC Dataset in Cyber Threat Detection Research." arXiv, 4 Mar. 2021, arxiv.org/abs/2103.03080.

[5] Canadian Institute for Cybersecurity. ”MalMem2022: Malware Memory Analysis Dataset." University of New Brunswick, 2022, https://www.unb.ca/cic/datasets/malmem-2022.html.

[6] Canadian Institute for Cybersecurity. ”Intrusion Detection Evaluation Dataset (CIC-IDS2017)." University of New Brunswick, 2017, https://www.unb.ca/cic/datasets/ids-2017.html.

[7] Canadian Institute for Cybersecurity. CSE-CIC-IDS2018 Dataset. University of New Brunswick, 2018, https://www.unb.ca/cic/datasets/ids-2018.html.

[8] Lemaitre, Guillaume, et al. Imbalanced-learn Documentation. Version 0.13.0, Dec. 2024, imbalanced-learn.org/stable/index.html.

[9] Cisco. XDR Buyer's Guide. Cisco, 2 Feb. 2024, https://www.cisco.com/c/en/us/products/collateral/security/xdr/xdr-buyer-guide.html.

[10]Soleman, D., Soewito, B. (2024). Information Security System Design Using XDR And EDR. Inform : Jurnal Ilmiah Bidang Teknologi Informasi Dan Komunikasi, 9(1), 51-57. https://doi.org/10.25139/inform.v9i1.7331.

[11]Pissanidis, Dimitrios Lazaros, and Konstantinos Demertzis. ”Integrating AI/ML in Cybersecurity: An Analysis of Open XDR Technology and its Application in Intrusion Detection and System Log Management.” Preprints.org, version 2 (not peer-reviewed), posted 3 Jan. 2024, doi:10.20944/preprints202312.0205.v2.

[12]Wilfurth, Meret. ”AI- and ML-Driven EDR — A Game Changer for Combatting Advanced Cyber Threats or a Risk in Itself?” Medium, 26 Nov. 2024, https://medium.com/@mwilfurth/ai-and-ml-driven-edr-a-game-changer-for-combatting-advanced-cyber-threats-or-a-risk-in-itself-863b83c70e45.

[13]Arfeen, Asad, Saad Ahmed, Muhammad Asim Khan, and Syed Faraz Ali Jafri. ”Endpoint Detection Response: A Malware Identification Solution.” 2021 International Conference on Cyber Warfare and Security (ICCWS), IEEE, 2021, doi:10.1109/ICCWS53234.2021.9703010.

[14]Jamadi, Zahra, and Amir G. Aghdam. Early Malware Detection and Next-Action Prediction. IEEE, 2023, https://ieeexplore.ieee.org/document/10290239.

[15]Shen, Xiangmin, et al. Decoding the MITRE Engenuity ATTCK Enterprise Evaluation: An Analysis of EDR Performance in Real-World Environments. Northwestern University et al., 2025, https://users.cs.northwestern.edu/ ychen/Papers/ASIACCS24.pdf.