# Comprehensive Analysis of Machine Learning-Based Network Intrusion Detection Systems: Design, Implementation, and Evaluation

## 1. Executive Summary

The modern digital ecosystem is characterized by an unprecedented interconnectivity of systems, ranging from enterprise cloud infrastructures to localized Internet of Things (IoT) networks. This expansion has fundamentally altered the cybersecurity landscape, expanding the attack surface and enabling malicious actors to deploy increasingly sophisticated, adaptive, and automated threats. The traditional paradigm of network defense—predicated largely on signature-based Intrusion Detection Systems (IDS)—is facing a crisis of efficacy. These legacy systems, which rely on static databases of known threat patterns, are inherently reactive; they require a "Patient Zero" to identify a new threat and a subsequent window of exposure to update signature definitions. In an era of zero-day exploits, polymorphic malware, and rapid-fire automated botnets, this latency is unacceptable.

This report presents an exhaustive research analysis on the design and development of an intelligent, Machine Learning (ML)-based Network Intrusion Detection System (NIDS). The primary objective of this project is to transcend the limitations of signature-based detection by deploying a data-driven framework capable of learning the behavioral characteristics of network traffic. By treating network security as a classification and anomaly detection problem, the proposed system aims to autonomously identify both known attacks (via Supervised Learning) and novel, unseen threats (via Unsupervised Learning).

The research focuses on a hybrid architecture that synergizes the strengths of three specific algorithms: **Random Forest (RF)** and **Support Vector Machines (SVM)** for robust classification of established attack vectors, and **Isolation Forest (iForest)** for the detection of zero-day anomalies.1 The analysis delves deep into the theoretical underpinnings of these models, their implementation using the Python ecosystem (Scikit-learn, Pandas), and their validation against industry-standard benchmark datasets: NSL-KDD, UNSW-NB15, and CIC-IDS2017.3

A critical component of this report is the rigorous examination of data engineering challenges, specifically the "Curse of Dimensionality" inherent in packet flow data and the severe Class Imbalance problem, where malicious traffic represents a minute fraction of overall network activity. The report details the application of advanced preprocessing techniques, including Synthetic Minority Over-sampling Technique (SMOTE) and Exhaustive Feature Selection (EFS), to optimize model performance.3 Furthermore, the evaluation framework prioritizes **Recall**—the probability of detecting an actual attack—recognizing that in the domain of cybersecurity, a False Negative (missed attack) carries a catastrophic risk profile compared to a False Positive (false alarm).6

By synthesizing empirical evidence, mathematical theory, and architectural best practices, this document establishes a comprehensive blueprint for a next-generation IDS that is adaptive, resilient, and capable of operating in the high-stakes environment of modern network security.

## 2. Problem Domain: The Crisis in Network Defense

### 2.1 The Evolution of the Threat Landscape

The trajectory of cyber threats over the last two decades has been defined by a shift from "Vandalism" to "Warfare." Early cyber attacks were often static, noisy, and focused on disruption. Modern threats, such as Advanced Persistent Threats (APTs), are stealthy, persistent, and financially or politically motivated. Attackers leverage automation to scan for vulnerabilities at planetary scale, and use Artificial Intelligence (AI) to obfuscate their code, changing its digital fingerprint while retaining its malicious payload (polymorphism).

The most dangerous category of threats facing modern networks is the **Zero-Day Attack**. These are exploits that target vulnerabilities unknown to the software vendor or the security community. Because no patch exists, and no signature has been created, traditional defenses are blind to them. The "window of vulnerability" for zero-days can extend for months or even years. To detect these, a defense system cannot rely on *what an attack looks like* (signature); it must recognize *what an attack does* (behavior).7

### 2.2 Limitations of Legacy Signature-Based IDS

Legacy IDS platforms, such as Snort or Suricata in their default configurations, operate on a rule-based engine. They perform deep packet inspection (DPI) to match packet contents against a repository of known signatures.

* **The Specificity Trap:** Signatures are brittle. A trivial modification to an exploit code—such as adding a "No Operation" (NOP) sled or changing a variable name—can alter the file hash or byte sequence enough to bypass the signature.
* **The Generalization Failure:** A rule created to detect "WannaCry Ransomware" cannot detect a new variant like "NotPetya" unless the rule is incredibly generic, which then leads to high false positives.
* **Maintenance Latency:** The protection offered by signature-based systems is only as good as the last update. In the gap between a new threat emerging and the signature update being applied, the network is defenseless.

The Problem Statement of this project addresses these precise failures. The goal is to build a system that *learns* the statistical properties of "Normal" vs. "Malicious" traffic, allowing it to generalize to new, unseen variations of attacks without human intervention.1

### 2.3 The Machine Learning Paradigm Shift

Machine Learning introduces a probabilistic approach to intrusion detection. Instead of a binary decision based on a database match, ML models evaluate the statistical probability that a given flow of network packets belongs to a malicious class. This project adopts a **Hybrid Machine Learning Strategy**, acknowledging that no single algorithm is a "silver bullet" for the diversity of network threats.1

* **Supervised Learning (The Classifier):** This component acts as the expert analyst. It is trained on vast quantities of labeled historical data (e.g., "This pattern of SYN packets is a DoS attack"). It excels at identifying known threats with high precision.
* **Unsupervised Learning (The Anomaly Detector):** This component acts as the sentinel. It is not told what an attack looks like; instead, it is taught what "normal" looks like. It flags anything that deviates significantly from this baseline, making it the primary defense against zero-day incursions.7

## 3. Theoretical Framework of Selected Algorithms

The selection of Random Forest, Support Vector Machine (SVM), and Isolation Forest is not arbitrary; it represents a strategic coverage of the classification spectrum. RF handles the complexity and noise of high-dimensional data; SVM handles complex decision boundaries in lower dimensions; and Isolation Forest handles the outlier detection necessary for zero-day threats.

### 3.1 Random Forest: The Ensemble Workhorse

Random Forest (RF) is an ensemble learning method that operates by constructing a multitude of decision trees at training time. It is a "Bagging" (Bootstrap Aggregating) technique, but with a critical modification that makes it superior for network data.

#### 3.1.1 Mechanism of Action

In a standard decision tree, the algorithm searches the entire feature space for the best split at each node. In contrast, Random Forest introduces randomness in two ways:

1. **Bootstrapping:** Each tree is trained on a random sample of the data drawn with replacement. This means some trees see specific attack instances while others do not, promoting diversity.
2. **Feature Subsampling:** At each split in the tree, the algorithm considers only a random subset of features (e.g., $\sqrt{n\\_features}$).

This second point is vital for NIDS. Network data often contains a few very strong features (e.g., Destination Port) that would dominate the decision making in every tree if standard bagging were used. By forcing trees to use other features (e.g., Flow Duration or Packet Variance), RF decorrelates the trees, reducing the variance of the model and making it highly robust to noise and overfitting.11

#### 3.1.2 Mathematical Basis: Entropy and Information Gain

The trees in the forest grow by splitting data to maximize "purity." This is typically measured using Gini Impurity or Entropy.

For a node $t$ with class probabilities $p(i|t)$, the Gini Impurity $I\_G(t)$ is:

$$I\_G(t) = 1 - \sum\_{i=1}^{C} p(i|t)^2$$

The algorithm selects the split that maximizes the Information Gain, which is the decrease in impurity:

$$Gain = I\_G(parent) - \sum\_{k} \frac{N\_k}{N} I\_G(child\_k)$$

In the context of NIDS, this mathematical process effectively ranks network features by their ability to discriminate between "Normal" and "Attack" traffic, providing intrinsic feature selection capabilities.11

### 3.2 Support Vector Machine (SVM): The Margin Maximizer

SVM is a supervised learning algorithm that is particularly powerful for binary classification tasks where the data is not linearly separable in its original space.

#### 3.2.1 Mechanism of Action

The objective of SVM is to find the hyperplane that maximizes the margin—the distance between the hyperplane and the nearest data points of any class (the support vectors).

$$\text{maximize } \frac{2}{||w||} \text{ subject to } y\_i(w \cdot x\_i + b) \ge 1$$

This optimization problem is solved using Lagrange multipliers.

#### 3.2.2 The Kernel Trick in NIDS

Network traffic data is complex and rarely linearly separable (e.g., a "Normal" HTTP request and a "Malicious" HTTP request might look very similar in 2D space). SVM overcomes this using the Kernel Trick, typically the Radial Basis Function (RBF) kernel:

$$K(x\_i, x\_j) = \exp(-\gamma ||x\_i - x\_j||^2)$$

This function maps the input vectors into an infinite-dimensional feature space where a linear separator can be found.

* **Relevance:** SVMs are highly effective at finding the optimal decision boundary in these high-dimensional spaces, provided the dataset size is manageable. They are robust against overfitting in high-dimensional space, which is common with feature-rich datasets like CIC-IDS2017.3 However, their training time complexity ($O(n^3)$) makes them slower than RF for massive datasets.

### 3.3 Isolation Forest: The Anomaly Hunter

Isolation Forest (iForest) represents a fundamental departure from distance-based or density-based anomaly detection (like LOF or DBSCAN). Instead of building a model of "Normal" data, it explicitly isolates anomalies.

#### 3.3.1 Mechanism of Action

The premise of iForest is that anomalies are "few and different."

1. **Random Partitioning:** The algorithm builds an ensemble of "Isolation Trees" (iTrees). To build a tree, it recursively selects a random feature and a random split value between the max and min of that feature.
2. **Path Lengths:** Because anomalies are statistically distinct (located in sparse regions of the feature space), they require fewer random splits to be isolated. Normal points, being clustered together, require more splits to separate.
3. **Anomaly Score:** The path length $h(x)$ from the root to the terminating node is averaged across all trees. A shorter path length indicates a higher likelihood of being an anomaly.

#### 3.3.2 The Anomaly Score Equation

The anomaly score $s(x, n)$ is defined as:

$$s(x, n) = 2^{-\frac{E(h(x))}{c(n)}}$$

Where $c(n)$ is the average path length of an unsuccessful search in a Binary Search Tree (the normalization factor).

* If $s$ is close to 1, the instance is definitely an anomaly (Attack).
* If $s$ is close to 0, the instance is normal.
* If $s$ is near 0.5, the instance is indistinguishable from normal traffic.

This linear time complexity ($O(n)$) makes iForest uniquely suited for high-volume network traffic analysis compared to computationally expensive nearest-neighbor methods.13

## 4. Data Ecosystem: Anatomy of Network Datasets

The validity of any ML-NIDS research hinges on the quality of the data used for training. This project utilizes three distinct benchmarks, each representing a different era and complexity of network traffic.

### 4.1 NSL-KDD: The Academic Benchmark

The KDD Cup 99 dataset was the de facto standard for decades but was flawed due to massive data redundancy (78% duplicates in training, 75% in testing). This redundancy caused classifiers to bias heavily toward frequent records (like normal traffic or DoS attacks) and ignore rare attacks.16

**NSL-KDD** was created to solve this. It is a distilled version of KDD99 with redundant records removed.

* **Features:** 41 features, including basic features (Duration, Protocol, Service), content features (Login status, Root access), and traffic features (Count, Serror rate).
* **Utility:** While the traffic patterns (Telnet, FTP exploits) are dated, NSL-KDD remains a vital benchmark for algorithmic comparison because it is computationally lightweight and well-understood. It allows for rapid prototyping of the RF and SVM models before scaling to larger datasets.3

### 4.2 UNSW-NB15: The Modern Hybrid

Developed by the Cyber Range Lab of UNSW, this dataset addresses the lack of modern attack vectors in KDD. It combines real normal traffic with synthetic attack behaviors generated by the IXIA PerfectStorm tool.18

* **Complexity:** It contains 49 features and, crucially, a wider variety of attack families (9 categories): Fuzzers, Analysis, Backdoors, DoS, Exploits, Generic, Reconnaissance, Shellcode, and Worms.
* **Features:** It introduces sophisticated flow-based features like smean (Source mean packet size), dmean (Destination mean packet size), and tcprtt (TCP Round Trip Time), which are highly indicative of connection quality and potential flooding attacks.20
* **Challenge:** The dataset is highly imbalanced. "Generic" and "Normal" classes dominate, while "Worms" and "Shellcode" are incredibly rare. This makes it an ideal testbed for evaluating the effectiveness of SMOTE and class-weighting strategies.18

### 4.3 CIC-IDS2017: The State-of-the-Art

This dataset represents the most current and realistic approximation of a modern network environment. It was generated by the Canadian Institute for Cybersecurity (CIC).

* **Environment:** It captures traffic from a complete network topology including Modems, Firewalls, Switches, and diverse OSs (Windows, Ubuntu, Mac OS). It includes full packet payloads, not just headers.
* **Attacks:** It includes modern threats like **Heartbleed**, **Botnet**, **DDoS**, **Web Attacks (SQL Injection, XSS)**, and **Infiltration**.
* **Features:** Over 80 features extracted using CICFlowMeter, including statistical measures of flow inter-arrival times (IAT), packet lengths, and flag counts.21
* **Scale:** With millions of records, this dataset tests the scalability of the NIDS. It is the primary dataset used in this project to validate the system's performance against "real-world" conditions.

### Table 1: Comparative Analysis of Datasets

3

| **Feature** | **NSL-KDD** | **UNSW-NB15** | **CIC-IDS2017** |
| --- | --- | --- | --- |
| **Year** | 2009 | 2015 | 2017 |
| **Feature Count** | 41 | 49 | 80+ |
| **Attack Types** | 4 Categories (DoS, Probe, R2L, U2R) | 9 Families (Fuzzers, Worms, etc.) | 7+ Categories (Heartbleed, Botnet, Web, etc.) |
| **Traffic Type** | Simulated (Legacy) | Hybrid (Real/Synthetic) | Real/Emulated (Modern) |
| **Key Challenge** | Outdated patterns | Class Imbalance | High Dimensionality & Volume |

## 5. Data Engineering and Preprocessing Methodology

Raw network traffic data is "dirty"—it contains missing values, categorical strings, and vastly differing scales. Rigorous preprocessing is required to transform this raw signal into a structured format suitable for machine learning.

### 5.1 Data Cleaning and Imputation

Network logs often contain NaN (Not a Number) or Infinity values. This can happen due to packet capture errors or calculations involving division by zero (e.g., Flow Rate when duration is zero).

* **Strategy:** For the Flow Duration or Packet Rate features, infinite values are replaced with the maximum finite value in the column. Rows with NaN in critical identifier columns are dropped. For other numerical features, missing values are imputed using the **Median** strategy, which is more robust to outliers than the Mean.22

### 5.2 Categorical Encoding: The Curse of Dimensionality

Models like SVM and Neural Networks require numerical inputs. Network data contains categorical features like Protocol (TCP, UDP), Service (HTTP, FTP), and Flags.

* **Label Encoding:** Assigns a unique integer to each category (e.g., TCP=1, UDP=2). This preserves dimensionality but introduces an ordinal relationship (2 > 1) that does not exist in reality. This is acceptable for **Random Forest**, which handles categorical splits natively, but harmful for SVM.23
* **One-Hot Encoding:** Creates a new binary column for each category (e.g., Proto\_TCP, Proto\_UDP). This is mathematically correct for SVM but explodes the feature space (The "Curse of Dimensionality"). If the Service feature has 100 unique values, the dataset grows by 100 columns.
* **Project Decision:** We utilize **Label Encoding** for the Random Forest and Isolation Forest models to maintain computational efficiency. For SVM, we perform One-Hot Encoding but apply dimensionality reduction (PCA) subsequently to manage the feature space size.24

### 5.3 Feature Scaling: Standardization

Features in network datasets have radically different scales. Flow Duration might range from 0 to 100,000,000 (microseconds), while Packet Count might range from 1 to 1,000.

* **The Impact:** In distance-based algorithms like SVM and K-Means, the feature with the larger range will dominate the distance calculation (Euclidean distance), rendering the smaller feature irrelevant.
* Standardization (Z-Score): We apply the StandardScaler from Scikit-learn, which transforms the data such that it has a mean of 0 and a standard deviation of 1:  
    
  $$z = \frac{x - \mu}{\sigma}$$  
    
  This ensures that all features contribute equally to the model's decision boundary. While Random Forest is scale-invariant, standardization is critical for the SVM and Isolation Forest components.25

### 5.4 Addressing Class Imbalance: SMOTE

Class imbalance is the defining characteristic of NIDS data. In a real network, malicious traffic might be 0.1% of the total. In UNSW-NB15, "Worms" are a tiny fraction of the dataset.

* **The Risk:** A model trained on 99% benign traffic achieves 99% accuracy by predicting "Benign" for everything. It has high Accuracy but 0% Recall for attacks.
* SMOTE (Synthetic Minority Over-sampling Technique): We employ SMOTE to balance the training data. Unlike random oversampling (which duplicates rows and causes overfitting), SMOTE creates new synthetic instances. It selects a minority sample, finds its $k$ nearest neighbors in the feature space, and interpolates a new point along the vector between the sample and a neighbor.  
    
  $$x\_{new} = x\_i + \lambda \times (x\_{neighbor} - x\_i)$$  
    
  where $\lambda$ is a random number between 0 and 1.5
* **Implementation Note:** SMOTE is applied **only to the training set** within the cross-validation loop. Applying it to the test set would cause data leakage and invalidate the evaluation.5

## 6. System Architecture and Workflow

The proposed NIDS is not a monolithic model but a pipeline of interconnected components.

### 6.1 Feature Selection: Reducing the Noise

The CIC-IDS2017 dataset has over 80 features. Many are redundant (e.g., Total Length of Fwd Packets vs. Subflow Fwd Bytes). Using all features increases training time and the risk of overfitting.

* **Random Forest Feature Importance:** We utilize the Random Forest model itself for feature selection. By calculating the average decrease in Gini Impurity across all trees for each feature, we rank them by predictive power.
* **Top Features:** Research consistently shows that a subset of 15-20 features yields optimal results. Key features include Destination Port, Flow Duration, Total Fwd Packets, Packet Length Mean, and Flow IAT (Inter-Arrival Time) Mean. We employ **Recursive Feature Elimination (RFE)** to iteratively strip the least important features until the optimal subset remains.3

### 6.2 The Hybrid Detection Engine

The core innovation is the hybrid "Cascade" architecture.

1. **Tier 1: Supervised Classification (Random Forest):**
   * The incoming traffic vector (after preprocessing and feature selection) is fed into the Random Forest classifier.
   * The RF model, trained on labeled data (SMOTE-balanced), outputs a classification: "Normal" or "Attack Class X" (e.g., DoS).
   * **Decision:** If the RF predicts an attack with high probability ($>0.5$), an alert is raised immediately. RF is chosen here for its high precision and multi-class capability.
2. **Tier 2: Unsupervised Anomaly Detection (Isolation Forest):**
   * If Tier 1 classifies the traffic as "Normal," it is passed to the Isolation Forest.
   * The iForest calculates an Anomaly Score based on path length.
   * **Decision:** If the Anomaly Score exceeds a specific threshold (e.g., -0.5 offset or raw score > 0.7), the traffic is flagged as a "Zero-Day Anomaly."
   * **Logic:** Tier 1 catches known threats. Tier 2 catches threats that look "statistically weird" but didn't match the known attack patterns in Tier 1. This significantly reduces the False Positive rate of the iForest, as it only processes traffic that has already "passed" the first filter.9

### 6.3 Tools and Implementation Stack

The system is implemented in **Python**, the lingua franca of data science.

* **Pandas:** Used for high-performance data manipulation, reading massive CSVs (chunking strategies for CIC-IDS2017), and handling time-series indexing.
* **NumPy:** Provides the underlying vector/matrix operations for Scikit-learn.
* **Scikit-learn:** The primary ML library. We use:
  + RandomForestClassifier for the supervised tier.
  + SVC (Support Vector Classifier) for comparative benchmarking.
  + IsolationForest for the anomaly tier.
  + GridSearchCV for hyperparameter tuning.
  + Pipeline to chain preprocessing (Scaler) and modeling steps, preventing data leakage.26
* **Matplotlib / Seaborn:** Used for generating Confusion Matrices, ROC Curves, and Feature Importance plots.

## 7. Performance Evaluation Framework

In cybersecurity, standard metrics like "Accuracy" are deceptive. We utilize a suite of security-focused metrics.

### 7.1 The Confusion Matrix and Derived Metrics

* **True Positive (TP):** Attack correctly detected as Attack.
* **True Negative (TN):** Normal correctly identified as Normal.
* **False Positive (FP):** Normal incorrectly flagged as Attack (False Alarm).
* **False Negative (FN):** Attack incorrectly identified as Normal (Missed Breach).

From these, we derive:

* **Precision:** $TP / (TP + FP)$. High precision means fewer false alarms.
* **Recall (Sensitivity):** $TP / (TP + FN)$. High recall means fewer missed attacks. **This is the primary focus.**
* **F1-Score:** $2 \cdot \frac{Precision \cdot Recall}{Precision + Recall}$. The harmonic mean.

### 7.2 The Precision-Recall Trade-off and Alert Fatigue

There is an inverse relationship between Precision and Recall. Lowering the decision threshold (e.g., flagging anything with >30% attack probability) increases Recall (we catch more attacks) but drastically lowers Precision (we get more false alarms).

* **Alert Fatigue:** In a Security Operations Center (SOC), if an IDS generates 1,000 alerts and 950 are false positives, analysts will ignore the system. This "Alert Fatigue" is a critical operational failure mode.
* **Strategy:** We do not simply maximize Recall. We optimize the **F-beta Score** (specifically F2, which weighs Recall higher than Precision) to find a threshold that catches the maximum number of attacks while keeping False Positives within a manageable range for human analysts.6

### 7.3 ROC-AUC vs. Precision-Recall Curves

* **ROC-AUC:** Plots True Positive Rate vs. False Positive Rate. While standard, it can be overly optimistic on imbalanced datasets.
* **Precision-Recall (PR) Curve:** We focus on the PR Curve/Area Under Curve (PR-AUC). On highly imbalanced data (like UNSW-NB15), the PR curve gives a truer picture of the model's ability to classify the minority class (attacks) correctly without being swamped by the massive number of True Negatives.32

## 8. Empirical Analysis and Results

The analysis of the research snippets provides a clear picture of the system's performance.

### 8.1 Algorithm Benchmarking

* **Random Forest:** Consistently outperforms other supervised models. On CIC-IDS2017, RF achieves **99.5% Accuracy** and **99.9% Recall** for known attacks, superior to SVM (which struggles with the dataset size) and standard Decision Trees.3 Its ensemble nature makes it resilient to the noise in the flow duration features.
* **SVM:** While accurate (99.3% on NSL-KDD), its training time is prohibitive for real-time retraining on large datasets. It serves well as a "second opinion" classifier on smaller, highly specific feature subsets but is less viable as the primary engine for high-bandwidth links.3
* **Isolation Forest:** The unsupervised component demonstrates variable success rates depending on the nature of the zero-day attack. It achieves detection rates of **30-62%** on simulated zero-day attacks that the supervised models miss entirely. While this seems low compared to supervised recall, it represents a 30-62% improvement over zero visibility.7

### 8.2 Impact of SMOTE

The application of SMOTE on the UNSW-NB15 dataset yields a significant improvement in the detection of rare classes. Without SMOTE, detection of "Worms" and "Backdoors" is near zero. With SMOTE, Recall for these classes improves significantly, pushing the overall weighted F1-score from ~89% to **95.1%**.36

### 8.3 Hybrid System Efficacy

The cascaded hybrid model (RF + iForest) demonstrates the best operational profile. By using RF to filter known traffic, the Isolation Forest is exposed to less noise, which improves its anomaly detection precision. This architecture successfully balances the need for high Recall on known threats with the capability to flag novel deviations.1

## 9. Challenges, Limitations, and Future Outlook

### 9.1 Adversarial Machine Learning

A sophisticated attacker, knowing the system uses ML, can craft "Adversarial Examples." By adding imperceptible perturbations to the traffic (e.g., slightly altering packet timing within allowed variances), they can force the model to misclassify a malicious flow as normal.

* **Future Scope:** Research into **Adversarial Training** (injecting adversarial examples into the training set) is required to harden the model against these evasion techniques.37

### 9.2 Concept Drift

Network traffic changes over time (e.g., a new streaming service launches, altering "normal" bandwidth profiles). A model trained on 2017 data (CIC-IDS2017) may flag legitimate 2026 traffic as anomalous.

* **Future Scope:** Implementation of **Online Learning** or periodic retraining pipelines is necessary to handle Concept Drift. The system must continuously ingest verified logs to update its definition of "Normal".2

### 9.3 Explainable AI (XAI)

A "Black Box" model that says "Attack" without reason is hard to trust.

* **Future Scope:** Integration of SHAP (SHapley Additive exPlanations) values into the alert dashboard. This would tell the analyst *why* the RF model flagged a flow (e.g., "Flagged because Destination Port = 4444 and Flow Duration < 1ms"), bridging the gap between ML capability and human decision-making.37

## 10. Conclusion

This research confirms that a Machine Learning-Based Network Intrusion Detection System significantly enhances the defensive capabilities of modern networks. By moving beyond static signatures to behavioral learning, the proposed system addresses the critical vulnerability of zero-day exploits. The **Hybrid Architecture**, leveraging the precision of **Random Forest** for known threats and the outlier detection of **Isolation Forest** for unknown anomalies, offers a robust "Defense in Depth."

However, the success of such a system relies heavily on rigorous Data Engineering—specifically the use of **SMOTE** to handle class imbalance and **Feature Selection** to manage dimensionality. Furthermore, the operational deployment must carefully tune the **Precision-Recall Trade-off** to avoid Alert Fatigue. As cyber threats continue to evolve towards automation and AI-driven adaptation, this data-driven, adaptive security paradigm is not merely an improvement but a necessity for the integrity of digital infrastructure.

### Table 2: Technical Summary of System Components

| **Component** | **Technology / Algorithm** | **Role** | **Key Configuration / Insight** |
| --- | --- | --- | --- |
| **Preprocessing** | Pandas, Scikit-learn | Data Cleaning | Impute missing values with Median; Drop infinite values. |
| **Encoding** | Label Encoding | Categorical to Numeric | Used for RF/iForest to avoid dimensionality explosion. |
| **Scaling** | StandardScaler | Normalization | Critical for iForest distance calculations ($z$-score). |
| **Imbalance** | SMOTE | Data Balancing | Generates synthetic minority samples; applied to Train set only. |
| **Feature Selection** | Random Forest Importance | Dim. Reduction | Top 20 features selected (e.g., Flow Duration, Packet Length). |
| **Tier 1 Model** | Random Forest Classifier | Known Threat Detection | n\_estimators=100, criterion='gini'. High Precision. |
| **Tier 2 Model** | Isolation Forest | Zero-Day Detection | contamination='auto'. Flags statistical outliers. |
| **Metric Focus** | Recall (Sensitivity) | Evaluation | Optimized to minimize False Negatives (Missed Attacks). |

**Note on Citations:** Citations are integrated inline using the format to support claims, statistical data, and theoretical concepts derived from the provided research snippets.

#### Works cited

1. Hybrid Machine Learning Models for Intrusion Detection: Combining Supervised and Unsupervised Techniques | Power System Technology, accessed on January 17, 2026, <https://powertechjournal.com/index.php/journal/article/view/2352>
2. Hybrid Machine Learning Models for Intrusion Detection: Combining Supervised and Unsupervised Techniques - Power System Technology, accessed on January 17, 2026, <https://powertechjournal.com/index.php/journal/article/download/2352/1686/4497>
3. Intrusion detection system based on machine learning using least ..., accessed on January 17, 2026, <https://pmc.ncbi.nlm.nih.gov/articles/PMC11978955/>
4. Improving Intrusion Detection with Hybrid Deep Learning Models: A Study on CIC-IDS2017, UNSW-NB15, and KDD - Journal of Information Systems Engineering and Management, accessed on January 17, 2026, <https://jisem-journal.com/index.php/journal/article/download/1665/653/2705>
5. Balancing Strategies in Machine Learning: Comparing SMOTE, Undersampling, and Class Weights in a… | by Gonzalo Surribas Sayago | Medium, accessed on January 17, 2026, <https://medium.com/@surribasg/balancing-strategies-in-machine-learning-comparing-smote-undersampling-and-class-weights-in-a-31d37106953a>
6. Trade-offs false alarm vs recall (probability of detection). - ResearchGate, accessed on January 17, 2026, <https://www.researchgate.net/figure/Trade-offs-false-alarm-vs-recall-probability-of-detection_fig1_325727106>
7. Enhancing Zero-Day Attack Detection in IoT Networks via Isolation Forest and Ensemble Tree Models - ELECTRICA, accessed on January 17, 2026, <https://electricajournal.org/index.php/pub/article/download/1243/1231>
8. Zero-Day Attack Detection System Using Autoencoders and Isolation Forest: An Unsupervised Machine Learning Approach - ResearchGate, accessed on January 17, 2026, <https://www.researchgate.net/publication/397497602_Zero-Day_Attack_Detection_System_Using_Autoencoders_and_Isolation_Forest_An_Unsupervised_Machine_Learning_Approach>
9. Network intrusion detection using hybrid approach, accessed on January 17, 2026, <https://journalwjarr.com/sites/default/files/fulltext_pdf/WJARR-2025-0367.pdf>
10. (PDF) Evaluating the Performance of SVM, Isolation Forest, and DBSCAN for Anomaly Detection - ResearchGate, accessed on January 17, 2026, <https://www.researchgate.net/publication/388323510_Evaluating_the_Performance_of_SVM_Isolation_Forest_and_DBSCAN_for_Anomaly_Detection>
11. Isolation Forest vs Random Forest | by Amit Yadav - Medium, accessed on January 17, 2026, <https://medium.com/@amit25173/isolation-forest-vs-random-forest-6e916623b8e5>
12. Random Forest Classification in Python With Scikit-Learn: Step-by-Step Guide (with Code Examples) | DataCamp, accessed on January 17, 2026, <https://www.datacamp.com/tutorial/random-forests-classifier-python>
13. What is Isolation Forest - GeeksforGeeks, accessed on January 17, 2026, <https://www.geeksforgeeks.org/machine-learning/what-is-isolation-forest/>
14. Anomaly detection algorithms in Scikit-Learn - Nicolas Goix, accessed on January 17, 2026, <https://ngoix.github.io/nicolas_goix_osi_presentation.pdf>
15. Anomaly Detection in Python with Isolation Forest - DigitalOcean, accessed on January 17, 2026, <https://www.digitalocean.com/community/tutorials/anomaly-detection-isolation-forest>
16. RANDOM FOREST CLASSIFICATION OF NSL- KDD DATASET USING HYBRID FEATURE SELECTION MODEL - Jetir.Org, accessed on January 17, 2026, <https://www.jetir.org/papers/JETIR1812462.pdf>
17. Feature Selection for Intrusion Detection Using Random Forest - SciRP.org, accessed on January 17, 2026, <https://www.scirp.org/journal/paperinformation?paperid=65359>
18. NETWORK INTRUSION DETECTION USING THE UNSW-NB15 ..., accessed on January 17, 2026, <https://aircconline.com/csit/papers/vol15/csit151707.pdf>
19. The UNSW-NB15 dataset - Research Data Australia, accessed on January 17, 2026, <https://researchdata.edu.au/the-unsw-nb15-dataset/1957529>
20. The features of UNSW-NB15 dataset. | Download Scientific Diagram - ResearchGate, accessed on January 17, 2026, <https://www.researchgate.net/figure/The-features-of-UNSW-NB15-dataset_tbl1_338504507>
21. Intrusion Detection Evaluation Dataset (CIC-IDS2017) - University of New Brunswick, accessed on January 17, 2026, <https://www.unb.ca/cic/datasets/ids-2017.html>
22. Multi-class Network Intrusion Detection with Class Imbalance via LSTM & SMOTE - arXiv, accessed on January 17, 2026, <https://arxiv.org/pdf/2310.01850>
23. One Hot Encoding vs Label Encoding | by Amit Yadav | Biased-Algorithms - Medium, accessed on January 17, 2026, <https://medium.com/biased-algorithms/one-hot-encoding-vs-label-encoding-28aee12b3984>
24. One Hot Encoding vs Label Encoding - GeeksforGeeks, accessed on January 17, 2026, <https://www.geeksforgeeks.org/machine-learning/one-hot-encoding-vs-label-encoding/>
25. Resampling Imbalanced Network Intrusion Datasets to Identify Rare Attacks - MDPI, accessed on January 17, 2026, <https://www.mdpi.com/1999-5903/15/4/130>
26. Building A Scikit Learn Classification Pipeline - Kaggle, accessed on January 17, 2026, <https://www.kaggle.com/code/gautham11/building-a-scikit-learn-classification-pipeline>
27. Feature selection using random forest for the NSL-KDD dataset - ResearchGate, accessed on January 17, 2026, <https://www.researchgate.net/figure/Feature-selection-using-random-forest-for-the-NSL-KDD-dataset_tbl2_366761220>
28. Top 15 features by Random Forest importance -NSL-KDD - ResearchGate, accessed on January 17, 2026, <https://www.researchgate.net/figure/Top-15-features-by-Random-Forest-importance-NSL-KDD_fig1_396778583>
29. Hybrid Intrusion Detection System Based on Combination of Random Forest and Autoencoder - MDPI, accessed on January 17, 2026, <https://www.mdpi.com/2073-8994/15/3/568>
30. A Simple Example of Pipeline in Machine Learning with Scikit-learn | Towards Data Science, accessed on January 17, 2026, <https://towardsdatascience.com/a-simple-example-of-pipeline-in-machine-learning-with-scikit-learn-e726ffbb6976/>
31. You Don't Have to Choose Between Alert Overload and Missing Real-World Attacks, accessed on January 17, 2026, <https://www.dropzone.ai/blog/you-dont-have-to-choose-between-alert-overload-and-missing-real-world-attacks>
32. Precision-Recall — scikit-learn 1.8.0 documentation, accessed on January 17, 2026, <https://scikit-learn.org/stable/auto_examples/model_selection/plot_precision_recall.html>
33. How to use scikit-learn's TunedThresholdClassifierCV for Threshold Optimization?, accessed on January 17, 2026, <https://www.geeksforgeeks.org/machine-learning/how-to-use-scikit-learns-tunedthresholdclassifiercv-for-threshold-optimization/>
34. Machine learning-based network intrusion detection for big and imbalanced data using oversampling, stacking feature embedding and feature extraction - arXiv, accessed on January 17, 2026, <https://arxiv.org/html/2401.12262v1>
35. Enhancing Zero-Day Attack Detection in IoT Networks via Isolation Forest and Ensemble Tree Models - R Discovery, accessed on January 17, 2026, <https://discovery.researcher.life/article/enhancing-zero-day-attack-detection-in-iot-networks-via-isolation-forest-and-ensemble-tree-models/54ad3888a9ef3439aec34b7b8d0e7777>
36. Network intrusion detection using oversampling technique and machine learning algorithms - PMC - NIH, accessed on January 17, 2026, <https://pmc.ncbi.nlm.nih.gov/articles/PMC8771780/>
37. Robust IoT security using isolation forest and one class SVM algorithms - PMC - NIH, accessed on January 17, 2026, <https://pmc.ncbi.nlm.nih.gov/articles/PMC12540939/>