Host-based Intrusion Detection System using System Call Analysis

Aryan Vats,Nitesh Jha,Shikhar Sharma
Computer Science and Engineering
NIIT University
Neemrana, India
Email:aryan.vats22@st.niituniversity.in
nitesh.jha22@st.niituniversity.in
shikhar.sharma22@st.niituniversity.in

***Abstract*—The difficulty of precisely identifying host-based incursions in real-time without sacrificing system efficiency is the main problem this study attempts to solve. The accuracy and speed of current host-based intrusion detection systems (HIDS) are frequently traded off. Conventional approaches can concentrate on high accuracy, which could lead to delayed detection and possible security breaches or prioritize fast response times at the expense of more false positives. This research intends to improve Host-Based Intrusion Detection Systems (HIDS) through increased accuracy, quicker response times, and a balanced approach to anomaly detection by utilizing three important datasets: ADFA-LD, ADFA-WD, and ADFA-SAA.This approach integrates feature extraction methods such as TF-IDF, SVD, and context-aware feature extraction with a dual-focused detection system that balances early warning signals and in-depth pattern analysis. The evaluation of various models, including Random Forest, SVM, Naïve Bayes, CNN, and RNN, demonstrates the efficacy of the hybrid methodology in addressing key challenges like data imbalance and the trade-off between speed and accuracy.**

***Index Terms*—Host-based intrusion detection, System call analysis,ADFA-LD,ADFA-WD datasset,ADFA-WD:SAA dataset,Machine Learning,Host-based intrusion detection, System call analysis, ADFA-LD dataset, Machine learning, Digital forensics, Anomaly detection**

I. Introduction

In this section, we provide an overview of host-based intrusion detection systems.

A. Background

The growing complexity and frequency of cyberattacks, which necessitate increasingly sophisticated and reliable security measures, are the driving forces behind this initiative. Particularly in host-based contexts, traditional security solutions frequently have trouble identifying complex and dynamic threats. This research intends to improve Host-Based Intrusion Detection Systems (HIDS) through increased accuracy, quicker response times, and a balanced approach to anomaly detection by utilizing three important datasets: ADFA-LD, ADFA-WD, and ADFA-SAA.
To enhance the identification of intricate infiltration patterns, the study presented in the documents provided highlights the necessity of sophisticated machine learning and deep learning methodologies. For example, the ADFA-WD dataset highlights the advantages of ensemble learning models for improved classification accuracy, while the ADFA-SAA dataset has demonstrated the efficacy of deep learning models in categorizing unexpected attacks. The ADFA-LD dataset contributes to feature extraction and optimization techniques that boost detection performance.

B. Related Work

The literature survey covers a range of research papers and studies utilizing the ADFA-LD, ADFA-WD, and ADFA-SAA datasets, focusing on enhancing Host-Based Intrusion Detection Systems (HIDS). Key insights, existing solutions, and identified gaps are summarized below:
Research Utilizing the ADFA-LD Dataset

- Machine Learning and Optimization: Aziz & Alfoudi (2023) [1] applied feature selection and hyperparameter optimization techniques to improve detection rates in HIDS. They highlighted the effectiveness of hybrid models that integrate traditional machine learning (ML) with optimization methods. However, the study noted challenges in maintaining real-time efficiency while achieving high accuracy.
- Feature Extraction with TF-IDF and SVD: Subba & Gupta (2021) [2] developed a framework using Term Frequency-Inverse Document Frequency (TF-IDF) and Singular Value Decomposition (SVD) for dimensionality reduction. While this approach enhanced anomaly detection, the study lacked an evaluation of performance in highly dynamic environments.
- Context-Aware CNNs: Shams et al. (2021) [3] introduced a context-aware feature extraction approach for Convolutional Neural Networks (CNNs) in IDS, achieving superior accuracy over conventional ML techniques. A gap identified here is the need for generalized models that perform well across diverse datasets.

Research Utilizing the ADFA-WD Dataset

- Stacking Ensemble Approach: Kumar & Subba (2023) [4] proposed a stacking ensemble model using word embeddings to identify abnormal processes in Windows environments. The research demonstrated improved clas-

sification accuracy but faced challenges related to computational overhead and deployment in resource-constrained environments.
- Ensemble Learning Techniques: Satılmış et al. (2025) [5] trained multiple ensemble models on the ADFA dataset, showing that ensemble methods outperform individual models in detection performance. However, this approach did not address the scalability of ensemble models in real-time systems.

Research Utilizing the ADFA-SAA Dataset

- Anomaly Detection with SVM: Liu, Zhang, and Chen applied Support Vector Machine (SVM) classifiers with sigmoid and Radial Basis Function (RBF) kernels to distinguish between normal and abnormal patterns. While effective, their study highlighted the difficulty of handling imbalanced datasets, which can lead to biased model performance.
- Deep Learning Models: Li, Liu, and Zhang demonstrated that Deep Neural Networks (DNNs) could enhance the capabilities of HIDS by offering higher accuracy in detecting complex intrusion patterns. However, the trade-off included increased computational requirements.
- Comparative Analysis of ML Models: Zhang, Liu, and Wang compared traditional ML models with Recurrent Neural Networks (RNNs), finding that RNNs provided enhanced accuracy. The challenge here involved optimizing the models to reduce false positives while maintaining high accuracy.

Identified Gaps and Challenges

- Data Imbalance: Many studies struggled with imbalanced datasets, which can negatively affect the accuracy of anomaly detection models.
- Trade-off Between Speed and Accuracy: Balancing real-time detection with high classification accuracy remains a critical challenge.
- Scalability and Generalization: While deep learning models showed promise, their applicability across different datasets and environments needs improvement.
- Computational Efficiency: Complex models often require significant computational resources, limiting their deployment in real-world scenarios.

C. Contribution

We present a hybrid methodology that combines techniques from three important datasets (ADFA-LD, ADFA-WD, and ADFA-SAA) to advance the field of Host-Based Intrusion Detection Systems (HIDS). This project's main contributions are as follows:

- Enhanced Detection Accuracy: Our method seeks to increase intrusion detection systems' accuracy by fusing machine learning and deep learning approaches. According to the study on the ADFA-SAA dataset, combining more sophisticated models like CNNs and RNNs with more conventional models like Random Forest and SVM enables improved detection of intricate infiltration patterns.
- Balanced Real-Time Performance: Our hybrid technique combines deep pattern analysis and early detection algorithms, in contrast to traditional systems that frequently sacrifice speed for accuracy. This solves issues noted in research utilizing the ADFA-WD dataset by guaranteeing quick anomaly detection without sacrificing classification performance.
- Robust Feature Engineering: Our study improves the representation of system behavior by employing feature extraction methods such as TF-IDF, SVD, and context-aware feature extraction (as shown in the ADFA-LD dataset research). This makes it easier to distinguish between benign and malevolent activity.
- Addressing Data Imbalance: Our methodology includes strategies to handle imbalanced datasets, such as specialized loss functions and few-shot learning, improving the detection rates for rare but critical threats.

D. Outline of the Paper

The structure of this paper is organized as follows:

- Section II presents the research methodology adopted for this study. It describes the systematic approach taken to review existing literature, the selection of relevant papers, and the evaluation of feature extraction and classification techniques utilized in current Host-Based Intrusion Detection Systems (HIDS). Additionally, it provides an overview of the experimental setup, including hardware, software, and tools used. This section details the dataset employed in the research, along with the programming languages, frameworks, and libraries utilized for model implementation and training.
- Section III focuses on data acquisition, offering a comprehensive overview of the ADFA datasets. It elaborates on the data collection process, the structure of the datasets, and the types of system calls recorded. Furthermore, it discusses the feature extraction techniques applied to transform the raw system call sequences into structured data suitable for analysis.
- Section IV covers the data analysis and preprocessing stages. It discusses essential preprocessing steps such as data cleaning, normalization, and feature selection. The section further elaborates on the feature extraction methodologies, emphasizing different techniques like Term Frequency-Inverse Document Frequency (TF-IDF) and Word2Vec. The classification phase is also detailed, highlighting the use of machine learning models, including Random Forest, XGBoost, and Support Vector Machines (SVM), along with deep learning approaches such as Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks.
- Section V outlines the proposed methodology for intrusion detection. It systematically presents the intrusion detection framework, dividing the process into three major components: preprocessing and feature engineering, model construction using deep learning and machine learning techniques, and hyperparameter optimization to

enhance detection performance. The section also details the evaluation metrics employed to assess model effectiveness and reliability.
- Section VI presents the experimental results and discussion. It provides a comprehensive analysis of the model's performance. It also discusses the strengths and limitations of each method and examines the overall effectiveness of the proposed approach in detecting intrusions within the ADFA datasets.
- Section VII concludes the paper by summarizing the key findings of the research. It highlights the contributions of the study, discusses potential improvements, and suggests directions for future research in the domain of Host-Based Intrusion Detection Systems using system call analysis.

II. Research Methodology

This study adopts a systematic research approach to investigate Host-Based Intrusion Detection Systems (HIDS) utilizing system call analysis on Windows environments. The research focuses on developing an effective detection mechanism using advanced machine learning methodologies. Given the increasing sophistication of cyber threats targeting endpoint systems, this study aims to enhance HIDS capabilities by leveraging feature extraction, classification models, and optimization techniques.

To ensure a comprehensive analysis, we reviewed a collection of relevant research studies and experimental frameworks, focusing on system call datasets and intrusion detection methodologies. The research primarily centers on the Australian Defense Force Academy Datasets, which provides real-world system call logs for training and evaluation. A well-defined set of search terms, including "Windows System Calls," "Host-Based Intrusion Detection," "Machine Learning (ML)," "Deep Learning (DL)," and "ADFA-LD","ADFA-WD","ADFA-WD: SAA" was used to identify and analyze prior studies.

The selected studies underwent rigorous evaluation to identify best practices and address critical challenges in implementing effective HIDS models. Our investigation emphasized feature extraction techniques such as Term Frequency-Inverse Document Frequency (TF-IDF) and Word2Vec to transform system call sequences into meaningful numerical representations.

The insights gained from this research highlighted key challenges, including high false positive rates, computational overhead, and the need for real-time detection capabilities. These findings informed the development of a hybrid ML-DL approach, integrating traditional machine learning models with deep learning architectures to improve intrusion detection accuracy and scalability. This study aims to contribute to the advancement of HIDS solutions by proposing a robust, efficient, and adaptive framework for detecting malicious activities in Windows environments.

A. Experimental Setup

To evaluate the performance of the proposed HIDS framework, we designed an experimental setup comprising the following key components:

- Dataset: The ADFA datasets were used, containing system call sequences generated from both benign and malicious activities in Windows and Linux environments. Data preprocessing involved noise reduction, normalization, and transformation into structured input formats. Feature Extraction: TF-IDF and Word2Vec were applied to convert raw system call sequences into numerical representations suitable for machine learning models.
- Model Selection: The following classifiers were employed for performance comparison:
  - Random Forest
  - Support Vector Machine (SVM)
  - XGBoost
  - Stacking Ensemble (combining multiple classifiers for improved a
- Experimental Environment: All experiments were conducted on a system with the following specifications:
  - Processor: Intel Core i5 (11th Gen) or equivalent
  - RAM: 12GB
  - GPU: NVIDIA RTX 3060 (if deep learning models were involved)
  - Software:Python 3.9+, Scikit-learn Library, TensorFlow/PyTorch (for deep learning models)

III. Data Acquisition

A. ADFA-LD

The ADFA-LD dataset contains system call sequences varying in length from the Linux operating system. Each system call sequence consists of unique IDs representing system calls.ADFA-LD is divided into three subsets: training, validation, and attack. The training and validation subsets contain normal-type system call sequences. The attack subset includes system call sequences related to six different attack types. The numbers of system call sequences and attack types in ADFA-LD are given in Table 1.

TABLE I: Numbers of System Call Sequences and Attack Types in ADFA-LD

Category	Count
Training	833
Validation	4372
Attack Types	
Add Superuser	91
FTP Password Bruteforce	162
SSH Password Bruteforce	176
Java Meterpreter	124
Linux Meterpreter	75
Web Shell Attack	118
Total	5951

B. ADFA-WD

The ADFA-WD dataset consists of DLL call sequences from the Windows XP operating system. Each DLL call sequence is represented by DLL calls identified by DLL names and specific memory access addresses. ADFA-WD includes three subsets: training, validation, and attack. The training and validation subsets contain normal-type DLL call sequences, with 355 and 1827 sequences, respectively. The attack dataset includes 5542 DLL call sequences covering 12 different attack types. The numbers of DLL call sequences and attack types in ADFA-WD are given in Table 2.

TABLE II: Numbers of System Call Sequences and Attack Types in ADFA-WD

Category	Count
Training	355
Validation	1827
Attack Types	
V1-CesarFTP	454
V2-WebDAV	470
V3-Icecast	382
V4-Tomcat	418
V5-OS-SMB	355
V6-OS-Print-Spool	454
V7-PMWiki	430
V8-Wireless-Karma	487
V9-PDF	440
V10-Backdoored Executable	536
V11-Browser-Attack	495
V12-Infectious-Media	621
Total	5542

IV. Data Analysis

A. Analysis on ADFA-LD dataset

- Preprocessing Pipeline
  Our complete preprocessing workflow:

  1) Sequence normalization (z-score)
  2) Rare call grouping (frequency ¡ 0.1%)
  3) Adaptive windowing
  4) Context padding
- System Architecture
  Our framework comprises three main components:

$$\mathcal{F} = \{f_1, f_2, f_3\} = \{\text{Preprocessing}, \text{Feature Extract}, \text{Classification}\} \tag{1}$$

- Preprocessing Module
  - Sequence Segmentation:

$$W_i = \{s_j, s_{j+1}, ..., s_{j+k-1}\}, \quad k = \min(15, |S|) \tag{2}$$

    where S is the complete system call sequence.
  - Call Encoding:

$$e(s_i) = \begin{cases} \text{index}(s_i) & \text{if freq}(s_i) \geq \theta \\ \text{UNK} & \text{otherwise} \end{cases} \tag{3}$$

    with θ = 0.1% frequency threshold.
- Feature Extraction
  We implement and compare multiple approaches:
  Temporal difference features are computed as:

$$\Delta_t = \frac{1}{2n} \sum_{i=1}^{n} |x_{t+i} - x_{t-i}|, \quad n = 5 \tag{4}$$

TABLE III: Feature Extraction Methods

Method	Description	Dimension
TF-IDF+SVD	Term frequency with dimensionality reduction	50
N-grams	Sequential patterns (n=1-3)	342^3
TempDiff	Temporal differences (Eq. 5)	15
CNN	Automatic feature learning	128

- Feature Analysis
  Key findings from feature evaluation:

$$\mathcal{R} = \frac{\text{Detection Rate with Feature}}{\text{Baseline Rate}} - 1 \tag{5}$$

  Where:
  - TF-IDF: $\mathcal{R}_{TF-IDF}$ = +18%
  - Temporal Diff: $\mathcal{R}_{TempDiff}$ = +12%
  - CNN: $\mathcal{R}_{CNN}$ = +23%
- Classification Models
  We evaluate four model architectures:
  - **Random Forest**: 200 trees, max depth=15
  - **CNN**: Three 1D convolutional layers (kernels=3,5,7)
  - **BiLSTM**: 128 units, dropout=0.3
  - **Hybrid Ensemble**: Weighted combination of above

B. Analysis on ADFA-WD dataset

- Balancing Dataset
  - ADFA-WD was originally an imbalanced dataset with number of attack-type DLL call sequences being approximately 2.5 times larger than the normal type sequences.
  - In order to overcome this imbalance problem, the datasets of Training and Validation data were combined together to form a single dataset of normal-type sequences and 40 percent of the attack-type sequences were chosen randomly. [5]
  - This resulted in a total dataset size of 4399 sequences (2182 normal + 2217 attack).
  - This made sure that the model would not be biased towards the dominating attack class, leaving a negative impact on the system's overall performance
- Selection of First N System Calls
  - The first N (where N=100, 200, 300, 400, or 500) DLL calls were extracted from every sequence of trace for the purpose of early detection, before feature extraction, following the approach from [6]
  - Feature Extraction using N-gram and Bag of Words (BoW) Techniques
  - N=5-grams were taken from the DLL-call sequences, generating distinct 5-gram representations.
  - These 5-grams were converted into a Standard Bag-of-Words (BoW) dataset where each n-gram's frequency was recorded as a feature.
- Feature Selection by using MI (Mutual Information)
  - Mutual Information (MI) was calculated for each feature using Algorithm 1, following the approach of [5].

Algorithm 1 is outline below:
Algorithm 1: Feature Selection Algorithm

* Input: BoW dataset
* Output: BoW dataset with selected features and reduced dimensionality
* Step 1: k = 2 where k is the cluster number
* Step 2: Calculating MI values of features
* Step 3: k-means(k, MI values along with the indexes of the features)
* Step 4: Selecting features in the set containing large MI values according to their indexes
* return BoW dataset with selected features and reduced dimensionality.

- After applying Algorithm 1, only the features with High MI Values were retained for training, thereby reducing dimensionality while preserving important features.

- Model Training and Evaluation
  - Machine learning models (KNN, DT, LR, RF) were trained using 75 percent of the data and evaluated on 25 percent.
  - Based on similar studies that used BoW and MI-based feature selection approaches, we anticipate a Recall metric performance to be in the range of 80-84 percent, with higher system call sequences are likely to give better results.
  - The variability in system calls can cause exact Recall metric values may fluctuate depending on the splitting of datasets.

C. Analysis on ADFA-WD:SAA dataset

V. Reporting and Presentation

A. Reporting on ADFA-LD dataset

- Overall Performance

TABLE IV: Comprehensive Performance Metrics

Model	Accuracy	Precision	Recall	F1	Time (ms)
Random Forest	91.2%	89.7%	90.1%	89.9%	3.2
CNN	93.8%	92.1%	93.4%	92.7%	8.9
BiLSTM	94.1%	93.8%	93.2%	93.5%	12.4
Hybrid	95.7%	94.9%	95.3%	95.1%	15.2

- Attack-Specific Analysis
  - **Privilege Escalation**:
    * Key Features: setuid() patterns, context switches
    * Best Model: CNN (98.2% detection)
  - **Code Injection**:
    * Key Features: ptrace() sequences, memory operations
    * Best Model: BiLSTM (92.7% detection)
  - **Rootkit Installation**:
    * Key Features: module operations, hook patterns
    * Best Model: Hybrid (94.8% detection)
- Computational Efficiency
  The hybrid model achieves:
  - Throughput: 65.8 sequences/second
  - Memory footprint: 1.1GB
  - Scalability: Linear with sequence length

B. Reporting on ADFA-WD dataset

- Key Observations:
  - Preliminary evaluation suggests that increasing the sequence length can improve the Recall metric up to a certain limit.
  - Standard BoW-based feature extraction datasets can effectively capture crucial patterns in DLL call sequences.
  - Mutual Information-based feature selection causes only the most relevant 5-grams to be retained.
  - In comparison to other frequency-based approaches, this approach maintains its high robustness against outliers.
- Expected Performance:
  - As observed from the past literature and similar methodologies, we expect a Recall metric performance lying in the range of 80-84
  - The impact of feature selection might lead to stabilized results, though minor fluctuations can occur in overall results.

C. Reporting on ADFA-WD:SAA dataset

VI. Conclusion and Future Work

A. Conclusion for ADFA-LD dataset

- Key Findings
  Our research demonstrates:
  - Hybrid approaches significantly outperform single-model techniques
  - Feature engineering is crucial for ADFA-LD analysis
  - Different attack types require specialized detection strategies
- Limitations
  - Processing overhead for very long sequences (¿10,000 calls)
  - Dependence on complete system call traces
  - Generalization to Windows systems needs verification
- Future Directions
  - Real-time streaming implementation
  - Federated learning for distributed deployment
  - Integration with kernel-level monitoring (eBPF)
  - Expansion to containerized environments

B. Conclusion for ADFA-WD dataset

- Summary of Findings:
  - The combination of 5-gram Standard BoW feature extraction and MI-based selection for preprocessing the ADFA-WD Dataset is expected to provide a stable host

intrusion detection framework for an effective Host-Based Intrusion Detection System for Windows.
  - Our approach follows the previously validated methodologies and should result in a consistent Recall metric performance.
- Future Work:
  - Further evaluation of feature extraction methodology can be conducted by using deep learning models for a potential improvement
  - Potential refining in the Feature Selection using alternative clustering methods than just k-means.

C. Conclusion for ADFA-WD:SAA Dataset

References

[1] M. R. Aziz and A. S. Alfoudi, "Different mechanisms of machine learning and optimization algorithms utilized in intrusion detection systems," in *AIP Conference Proceedings*, vol. 2839, no. 1. AIP Publishing, 2023.

[2] B. Subba and P. Gupta, "A tfidfvectorizer and singular value decomposition based host intrusion detection system framework for detecting anomalous system processes," *Computers & Security*, vol. 100, p. 102084, 2021.

[3] E. A. Shams, A. Rizaner, and A. H. Ulusoy, "A novel context-aware feature extraction method for convolutional neural network-based intrusion detection systems," *Neural Computing and Applications*, vol. 33, no. 20, pp. 13 647–13 665, 2021.

[4] Y. Kumar and B. Subba, "Stacking ensemble-based hids framework for detecting anomalous system processes in windows based operating systems using multiple word embedding," *Computers & Security*, vol. 125, p. 102961, 2023.

[5] H. Satilmiş, S. Akleylek, and Z. Y. Tok, "Development of various stacking ensemble based hids using adfa datasets," *IEEE Open Journal of the Communications Society*, 2025.

[6] X. Zhang, Q. Niyaz, F. Jahan, and W. Sun, "Early detection of host-based intrusions in linux environment," in *2020 IEEE International Conference on Electro Information Technology (EIT)*. IEEE, 2020, pp. 475–479.