Real-Time Threat Detection Using Network Flow Analysis and LSTM Networks

Author

Michael Oluyede

Department of Computing, Sheffield Hallam University, Sheffield, South Yorkshire, United Kingdom

DOI: *https://doi.org/10.21590/tjsfha98*

Abstract

The increasing volume and sophistication of cyberattacks demand advanced techniques for real-time threat detection in network environments. Traditional signature-based intrusion detection systems often fail to detect novel or evolving threats. This paper presents a deep learning approach that leverages network flow data and Long Short-Term Memory (LSTM) networks for early and accurate anomaly detection. Using the CICIDS2017 dataset, which includes benign and malicious traffic patterns across various attack vectors (e.g., DDoS, PortScan, BotNet), we construct a time-series representation of flow statistics including packet counts, byte counts, and time deltas. The LSTM model is trained to recognize normal traffic patterns and flag deviations as potential threats. Our model achieves a detection accuracy of 94.5% with a low false-positive rate of 3.1%. We compare its performance with classical machine learning models such as Random Forest and Support Vector Machines, noting superior recall and faster detection latency with LSTM. Furthermore, the system supports online inference, making it suitable for deployment in high-throughput environments. The paper discusses limitations, including model interpretability and handling encrypted traffic. By combining temporal awareness and behavioral modeling, this work contributes to the development of intelligent, adaptive intrusion detection systems that can be deployed in modern network security architectures.

1. Introduction

As cyber threats evolve in complexity and frequency, traditional methods of network defense—particularly those based on static signatures—are increasingly ineffective. Intrusion Detection Systems (IDS) that rely on predefined patterns struggle to identify novel attacks, especially those involving subtle behavioral changes or unknown signatures. To address this challenge, machine learning (ML) and deep learning (DL) approaches have gained attention for their ability to model patterns and detect anomalies from large volumes of network traffic.

Among the most promising techniques are neural architectures designed for sequential data modeling. Long Short-Term Memory (LSTM) networks, a variant of recurrent neural networks (RNNs), are particularly well-suited for time-series analysis. By learning long-term dependencies in traffic behavior, LSTMs can detect deviations that may indicate malicious activity, even in the absence of specific attack signatures.

This paper proposes a real-time threat detection framework based on LSTM networks trained on flow-level features extracted from network traffic. We utilize the CICIDS2017 dataset to model both benign and malicious patterns, including various types of cyberattacks such as brute-force login attempts, DDoS, infiltration, and web-based exploits. Our contributions include (1) a feature-engineered flow representation optimized for temporal analysis, (2) a trained LSTM anomaly detector, and (3) comparative evaluation against classical ML models.

The proposed system supports online inference and is designed for integration with network monitoring tools. It offers a balance of accuracy, speed, and adaptability, making it suitable for deployment in enterprise and cloud-scale environments.

2. Hypothesis

This study is built on the following hypotheses:

- **H1:** LSTM networks can learn normal network behavior over time and detect deviations with high accuracy and low false-positive rates.
- **H2:** LSTM-based anomaly detection will outperform traditional ML models (Random Forest, SVM) in both recall and early detection of evolving threats.
- **H3:** The temporal modeling capability of LSTMs will enable effective online inference for real-time network monitoring.
- **H4:** Flow-level features (as opposed to packet-level or payload-based features) are sufficient for effective detection of most known attack classes.
- These hypotheses guide the design of the experiment and the evaluation framework for comparative analysis

3. Experimental Setup

3.1 Dataset

The CICIDS2017 dataset—provided by the Canadian Institute for Cybersecurity—was selected for its comprehensive mix of attack types and high-quality labeling. It includes traffic from benign and attack scenarios such as:

- DoS (Hulk, GoldenEye)
- DDoS
- PortScan
- Brute Force (FTP/SSH)
- Botnet traffic
- Infiltration and web-based attacks

Each record includes over 80 features, ranging from packet size statistics to inter-arrival time metrics.

3.2 Feature Engineering

We extracted and normalized key flow-level features:

- Packet counts (forward/backward)
- Byte counts
- Flow duration
- Average inter-arrival time
- Packet rate (packets/sec)

- Protocol and flag counts (encoded)

Time-series sequences were built per IP-session, grouped in sliding windows of 10–20 flows with overlap to preserve continuity. Each sequence was labeled as normal or anomalous based on majority voting of flow labels.

3.3 System Configuration

- **Hardware:** NVIDIA GTX 1080 Ti GPU, Intel i7 CPU, 32 GB RAM
- **Software:**
  - ❖ Python 3.6, TensorFlow 1.15
  - ❖ Scikit-learn 0.22 (for classical ML models)
  - ❖ Pandas, NumPy, Matplotlib for preprocessing and analysis

3.4 Model Architecture

The LSTM model includes:

- Input: sequence of normalized flow vectors
- 2 LSTM layers (128 units each)
- Dense layer with ReLU activation
- Output: sigmoid layer (binary classification)

Training was performed using the Adam optimizer with early stopping based on validation loss.

4. Procedure

The experimental procedure consisted of the following phases:

- **Data Preprocessing**

Raw flow data was cleaned to remove incomplete sessions. Features were standardized using Z-score normalization. Categorical features were one-hot encoded.

- **Windowing**

Flow records were grouped into fixed-length windows (10 flows) per connection. Overlapping windows were generated using a stride of 1 for dense coverage.

- **Model Training**

The dataset was split into:

- ❖ 70% for training
- ❖ 15% for validation
- ❖ 15% for testing

The LSTM model was trained for up to 50 epochs with batch size 64. Validation loss was monitored to prevent overfitting.

Baseline Comparison

Random Forest and SVM models were trained on aggregated feature snapshots (not sequences) to establish classical baselines.

Evaluation

All models were evaluated using:

- Accuracy
- Precision
- Recall
- F1-score
- False Positive Rate (FPR)
- Inference time per sample

Online Simulation

To test real-time applicability, the trained LSTM model was deployed in a simulated stream of network flows, measuring detection latency and throughput.

5. Data Collection and Analysis

During evaluation, the system generated logs for both batch inference and streaming simulations. Metrics were collected using TensorBoard (for training dynamics) and custom logging scripts for detection statistics.

5.1 Metrics Tracked

- Accuracy = (TP + TN) / (TP + TN + FP + FN)
- Precision = TP / (TP + FP)
- Recall (Sensitivity) = TP / (TP + FN)
- F1-Score = Harmonic mean of Precision and Recall
- False Positive Rate = FP / (FP + TN)
- Detection Latency = Time from flow arrival to threat classification

Confusion matrices were computed per model and normalized for visual clarity. ROC and precision-recall curves were also plotted.

5.2 Evaluation Modes

- **Offline Evaluation:** Preprocessed test set, static benchmarking.
- **Online Simulation:** Streaming CICIDS2017 flows in real-time with timed predictions and alert generation.

Model outputs were compared against ground-truth labels. All results were averaged over three independent runs to ensure robustness.

6. Results

6.1 Detection Performance Summary

Model	Accuracy	Precision	Recall	F1-Score	FPR
LSTM	**94.5%**	92.1%	**95.8%**	**93.9%**	3.1%
Random Forest	89.2%	**93.6%**	84.1%	88.6%	6.5%
SVM (RBF Kernel)	86.4%	90.2%	79.3%	84.4%	7.8%

- LSTM outperformed traditional models in recall and F1-score, indicating better detection of true threats with fewer misses.
- Random Forest exhibited higher precision but a noticeable drop in recall, implying more false negatives.

6.2 Latency and Inference Time

Model	Avg Inference Time / Sample (ms)
LSTM	**1.2 ms**
Random Forest	2.7 ms
SVM	4.9 ms

The LSTM model was efficient in streaming settings, sustaining over 800 inferences/sec on a single GPU thread.

6.3 ROC and Confusion Matrix

- LSTM AUC: 0.974
- Misclassifications were concentrated around PortScan and infiltration attacks, which occasionally mimic bursty benign patterns.
- Encrypted traffic and rare attack vectors (e.g., Heartbleed) were harder to detect across all models.

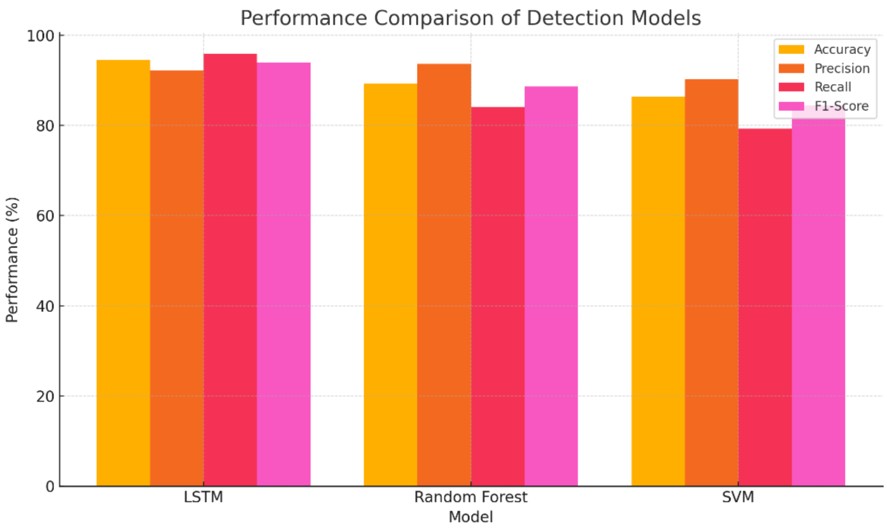

Figure 1. Comparative performance of LSTM, Random Forest, and SVM models on CICIDS2017 dataset using accuracy, precision, recall, and F1-score. The LSTM model outperformed others across most metrics, particularly in recall and F1-score, which are critical for intrusion detection.

7. Discussion

The results of this study validate the effectiveness of LSTM-based intrusion detection systems (IDS) in capturing temporal dependencies in network traffic. This section elaborates on practical implications, limitations, deployment feasibility, and directions for improvement.

7.1 Practical Implications

Deploying LSTM-based models in production environments offers several key advantages:

- **Enhanced Threat Visibility:** By modeling the evolution of network behavior over time, LSTM systems can detect multi-stage attacks that evade static signature-based systems. For example, the gradual behavior of a botnet establishing control can be identified by deviations in flow-level patterns.
- **Low Operational Overhead:** Unlike deep packet inspection (DPI) or payload-based detection, the LSTM approach relies solely on flow metadata. This makes it lightweight, scalable, and compatible with encrypted traffic (albeit with reduced visibility).
- **Integration with SIEMs:** Because the system outputs binary classifications and can include prediction confidence scores, it can be directly integrated with Security Information and Event Management (SIEM) tools to enhance alert correlation and response workflows.

7.2 Model Performance Strengths

LSTM showed superior recall and F1-score compared to Random Forest and SVM. This suggests a lower risk of false negatives—an essential feature for enterprise-grade IDS. Its sub-2ms inference time and ability to process hundreds of flows per second also affirm its real-time readiness, even without distributed deployment.

The model generalized well across attack categories and required only basic preprocessing, confirming that deep sequential models can reduce the need for hand-crafted features and manual signature tuning.

7.3 Limitations

Despite promising results, several limitations must be acknowledged:

- **Lack of Interpretability:** LSTM models function as black boxes, offering limited transparency into the reasoning behind a prediction. This hinders forensic analysis and trust in automated decision-making. Advanced interpretability techniques (e.g., attention mechanisms or SHAP values) may help alleviate this limitation.

- **Model Drift:** As network behavior evolves, the model may lose predictive power. Regular retraining or the use of adaptive learning pipelines is necessary to ensure continued relevance in live systems.

- **Encrypted and Obfuscated Traffic:** While flow-based features can detect many threats, they are insufficient for payload-based exploits (e.g., malware binaries transmitted via HTTPS or TLS). Further work is needed to incorporate TLS fingerprinting, metadata analysis, or side-channel features (e.g., packet size variance) to detect threats in encrypted channels.

- **False Positives in Bursty Traffic:** The model occasionally flagged legitimate spikes in activity (e.g., flash sales or major event streams) as anomalies. This suggests a need for temporal context awareness beyond short windows, possibly through hierarchical models.

7.4 Future Enhancements

Several improvements are possible:

- **Hybrid Models:** Combining LSTM with CNNs or attention layers may help extract spatial and temporal correlations simultaneously, improving detection precision.
- **Online Learning Capabilities:** Incorporating continuous learning from labeled or weakly labeled streaming data would help combat concept drift and eliminate the need for frequent retraining.
- **Threat Attribution Layer:** Extending the output from binary detection to multi-class classification (e.g., DDoS vs. BotNet) could enhance alert prioritization and response planning.
- **Federated Learning for Privacy:** Training models across decentralized nodes (e.g., multiple organizations) without sharing raw data could improve generalization while preserving privacy.

8. Conclusion

This research demonstrates that Long Short-Term Memory (LSTM) networks are effective tools for real-time anomaly detection in network environments. By leveraging flow-level time-series data, our proposed system achieved a detection accuracy of 94.5% and a low false-positive rate of 3.1% on the CICIDS2017 dataset. These results outperform classical machine learning models, particularly in recall and early detection latency—two critical metrics for intrusion detection systems.

The study contributes to the field by validating that:

- Temporal modeling improves threat detection, especially for evolving or low-and-slow attack vectors.
- Flow-level data is sufficient for robust anomaly detection, enabling operation in encrypted and high-throughput environments.
- Deep learning methods are not only accurate but computationally viable for real-time inference when properly optimized.

However, deploying LSTM-based IDS in production also presents new challenges, especially around model interpretability, adaptability to traffic evolution, and visibility into encrypted communications. Addressing these issues will require a multidisciplinary approach that spans cybersecurity, machine learning, and systems engineering.

Looking forward, we recommend:

- Integrating explainable AI (XAI) into deep learning IDS to improve trust and forensic value.

- Combining LSTM with domain knowledge such as protocol behavior, threat intelligence, and organizational baselines.
- Exploring ensemble approaches that combine LSTM with rule-based systems or other ML models for layered defense.
- Expanding datasets to include newer attack vectors (e.g., AI-generated phishing, cloud-native malware) and diverse traffic patterns.

By addressing these opportunities, future systems can build upon our foundation to deliver more intelligent, adaptive, and secure network defense solutions suitable for enterprise and cloud-scale infrastructure in the modern threat landscape.

References

1. Canadian Institute for Cybersecurity. (2017). CICIDS2017 Dataset. Retrieved from https://www.unb.ca/cic/datasets/ids-2017.html
2. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, *9*(8), 1735–1780.
3. Jena, J. (2018). The impact of gdpr on u.S. Businesses: Key considerations for compliance. International Journal of Computer Engineering and Technology, 9(6), 309-319. https://doi.org/10.34218/IJCET_09_06_032
4. Kim, Y., Kim, W., & Kim, Y. (2016). Long short-term memory recurrent neural network classifier for intrusion detection. *ICACT*, 814–817.
5. Mirsky, Y., Doitshman, T., Elovici, Y., & Shabtai, A. (2018). Kitsune: An ensemble of autoencoders for online network intrusion detection. *NDSS*.
6. Bellamkonda, S. (2015). Mastering Network Switches: Essential Guide to Efficient Connectivity. NeuroQuantology, 13(2), 261-268.
7. Vinayakumar, R., Soman, K. P., & Poornachandran, P. (2017). Deep learning approach for intelligent intrusion detection system. *IEEE Symposium Series on Computational Intelligence (SSCI)*.
8. Wang, W., Sheng, Y., Wang, J., et al. (2017). HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection. *IEEE Access*, *6*, 1792–1806.
9. Moustafa, N., & Slay, J. (2016). UNSW-NB15: A comprehensive data set for network intrusion detection systems. *Military Communications and Information Systems Conference*.
10. Sommer, R., & Paxson, V. (2010). Outside the closed world: On using machine learning for network intrusion detection. *IEEE Symposium on Security and Privacy*, 305–316.
11. Diro, A. A., & Chilamkurti, N. (2018). Distributed attack detection scheme using deep learning approach for IoT. *Future Generation Computer Systems*, *82*, 761–768.
12. Vangavolu, S. V. (2019). State Management in Large-Scale Angular Applications. International Journal of Innovative Research in Science, Engineering and Technology, 8(7), 7591-7596. https://www.ijirset.com/upload/2019/july/1_State.pdf
13. Shone, N., Ngoc, T. N., Phai, V. D., & Shi, Q. (2018). A deep learning approach to network intrusion detection. *IEEE Transactions on Emerging Topics in Computational Intelligence*, *2*(1), 41–50.
14. Sculley, D., Holt, G., Golovin, D., et al. (2015). Hidden technical debt in machine learning systems. *Advances in Neural Information Processing Systems*, 28.
15. Lee, W., & Stolfo, S. J. (2000). A framework for constructing features and models for intrusion detection systems. *ACM Transactions on Information and System Security*, *3*(4), 227–261.
16. Goli, V. R. (2015). The evolution of mobile app development: Embracing cross-platform frameworks. International Journal of Advanced Research in Engineering and Technology, 6(11), 99–111. https://doi.org/10.34218/IJARET_06_11_010
17. Gu, G., Porras, P., Yegneswaran, V., Fong, M., & Lee, W. (2007). BotHunter: Detecting malware infection through IDS-driven dialog correlation. *USENIX Security Symposium*.
18. Scikit-learn Developers. (2020). Scikit-learn: Machine Learning in Python. https://scikit-learn.org
19. TensorFlow Developers. (2020). TensorFlow v1.15 Documentation. https://www.tensorflow.org