Pak. j. life soc. Sci. (2026), 24(1): 446-453 E-ISSN: 2221-7630;P-ISSN: 1727-4915

Pakistan Journal of Life and Social Sciences
www.pjlss.edu.pk

https://doi.org/10.57239/PJLSS-2026-24.1.0038

RESEARCH ARTICLE

Hybrid Network Traffic Analysis System for Detecting Port Scanning Attacks Using Random Forest and Isolation Forest

A.Y. Poluyan[1*], K.S. Korovina[2]

[1]Assistant Professor, Don State Technical University, Rostov-on-Don, Russia

[2]Senior Lecturer, Don State Technical University, Rostov-on-Don, Russia

ARTICLE INFO

Received: APR 20,2026

Accepted: MAY 18, 2026

Keywords

Network traffic
Analysis
Port scanning
Cyber threat
Machine learning

***Corresponding Author:**
apoluyan@donstu.ru

ABSTRACT

The relevance of this research is driven by the increasing number of cyber threats, particularly port scanning attacks, which can lead to serious information security breaches. This circumstance creates a need for intelligent detection systems capable of analyzing traffic in real time and identifying attack signatures using machine learning technologies. This work focuses on developing a software tool for network traffic analysis and detection of port scanning packets using ensemble machine learning methods (Random Forest, Isolation Forest) and heuristic rules. In experimental evaluation on the CIC-IDS-2017 dataset, the system achieved an accuracy of 91.3%, recall of 88.7%, and an F1-score of 0.90.

INTRODUCTION

As the network perimeter becomes increasingly blurred due to the proliferation of cloud services, remote access, and the Internet of Things, and as threats grow in sophistication and speed of propagation, the ability to promptly detect port scanning activity becomes a critically important task. Port scanning is a fundamental reconnaissance operation in cyberspace; its goal is to identify specific vulnerabilities associated with open ports. Modern scanners, such as Nmap, possess immense flexibility, allowing attackers to mask their activity as legitimate traffic, use slow, distributed scanning from multiple sources, or fragment packets to bypass primitive filtering systems.

In this context, modern traffic analysis systems play a crucial role, evolving from simple signature-based intrusion detection systems into complex behavioral analysis platforms. Their role is not only to capture a known fingerprint of a scanning tool but also to identify the very anomaly in network behavior inherent to the reconnaissance process. These systems, employing deep packet and flow analysis methods, establish a behavioral baseline for the entire network. They continuously monitor and analyze communication patterns: the frequency of new connection establishments, the number of sequentially polled ports, etc.

Review of Network Traffic Analysis Technologies

Network traffic represents the aggregate of all data transmitted as packets between devices on a network. Physically, network traffic is expressed as a sequence of electrical pulses in cables, light flashes in optical fiber, or modulated radio waves in wireless environments. At the logical level, the physical nature of traffic is organized into a strict structure defined by network protocols.

Modern computer networks are complex distributed systems operating under constant opposition from various cyber threats. These threats can be classified by their scale of impact, implementation methods, and potential damage. Understanding the nature of these threats is fundamental to developing effective defense systems, including those for detecting port scanning as the initial stage of most attacks.

In local area networks, threats have their own specific characteristics. Internal threats include unauthorized access to network resources, where legitimate users or attackers compromising their credentials gain access to file storage, management systems, and databases. A significant danger is posed by the interception and analysis of network traffic using specialized software for eavesdropping on network communications and extracting credentials and confidential information. Among technical threats, ARP spoofing and Man-in-the-Middle (MitM) attacks stand out. The essence of the latter is to redirect traffic flows to the attacker's host, giving them the ability not only to eavesdrop but also to modify transmitted data. Furthermore, compromise of the network infrastructure itself—such as unauthorized configuration changes on switches, routers, or wireless access points—leads to serious risks.

Port scanning deserves special attention—it occupies a unique place among threats due to its versatility and key role in preparing cyberattacks. Essentially, this is initial reconnaissance: the attacker gathers data on network topology, open services, and OS versions. Based on this information, they search for vulnerabilities, determine attack vectors, and formulate a precise plan for further actions. Scanning methods are rapidly evolving: slow and distributed requests, packet fragmentation, evasion of detection systems, and adaptation of parameters to network responses are used. All this significantly complicates anomaly detection.

Modern network threats increasingly employ artificial intelligence and machine learning methods both for conducting attacks and for masking them. Reinforcement learning methods allow malware to adapt to specific network conditions and defense systems, developing optimal strategies to achieve its goals.

The implementation of network threats entails serious consequences. Direct damage manifests in financial losses related to theft of funds, system recovery costs, and fines for regulatory non-compliance; leakage of confidential information leading to compromise of intellectual property, personal data, and trade secrets; and disruption of business processes due to downtime of production and management systems. Indirect consequences include reputational damage, expressed in loss of customer and partner trust; legal liability for violations of data protection legislation; and technological risks requiring complete restructuring of the security infrastructure.

Modern methods of network traffic analysis can be classified according to various criteria, including operating principles, degree of automation, algorithms used, and level of analysis. By operating principle, signature-based, statistical, behavioral, and hybrid methods are distinguished. By analysis level, methods operating at the packet level, session level, and application layer are identified (Sangeen et al., 2025).

1. **Signature-Based Analysis**

Relies on comparing observed traffic with predefined patterns of known attacks. The advantages of signature-based analysis include a low false positive rate for known attacks, high speed when using optimized algorithms, and ease of implementation and understanding. However, the method has significant drawbacks, including ineffectiveness against new, unknown attacks, the need for constant signature database updates, and vulnerability to attack obfuscation and modification. Modern approaches to signature-based analysis include the use of hash signatures based on cryptographic hashes, behavioral signatures describing attack behavior patterns, and network signatures describing the network characteristics of attacks.

2. **Statistical Methods**

Analyze quantitative traffic characteristics to identify anomalies. Statistical analysis methods include threshold analysis through comparison with preset thresholds, time series analysis to identify trends and seasonality, cluster analysis for grouping similar objects, and outlier analysis for detecting anomalous values.

3. **Machine Learning Methods**

Offer the most promising approaches to network traffic analysis due to their ability to identify complex patterns. Machine learning methods as applied to network tasks are typically divided into three categories. The first is supervised learning: here the model classifies traffic types or predicts

numerical characteristics. The second category is unsupervised learning, which helps group similar data flows and find deviations from normal behavior. The third is reinforcement learning, enabling the construction of adaptive defense systems: the algorithm independently seeks optimal strategies for countering attacks.

In practice, applying ML to traffic analysis faces several challenges. These include proper feature extraction, working with large volumes of data, and the problem of imbalanced datasets. Regarding the features themselves, they can be divided into several types: static (IP addresses, port numbers, protocols), dynamic (traffic intensity, inter-packet intervals), statistical (mean values, variance, entropy), and behavioral (sequences of actions, typical interaction scenarios).

3.1 **Specifics of Port Scanning Detection**

Port scanning has several characteristics that complicate its detection:

- **Temporal Distribution**

Slow Nmap/Masscan modes (delays up to seconds) allow attackers to bypass systems analyzing short time windows (1–5 seconds).

- **Multiple Sources**

In distributed scanning, each botnet host operates within normal limits, but collectively creates an anomalous load on the target.

- **No Malicious Payload.**

Packets contain no exploits → signature analysis is useless.

- **Similarity to Normal Behavior.**

Modes such as nmap -sT -T0 mimic ordinary users: rare connections, no clear patterns. Differences are only visible on large samples or long intervals.

These characteristics necessitate combining approaches:

- time windows (30–300 seconds) — against slow scanning;
- event correlation — against distributed scanning;
- behavioral models (Isolation Forest) — for finding anomalies in multidimensional feature space.

3.2 **Hybrid Methods:**

Today, traffic analysis systems increasingly use combinations of different approaches—hybrids. The following variants are encountered.

Signature-Statistical Hybrids.

Here signatures work against known attacks, while statistics help detect new threats not yet in the databases.

ML-Signature Hybrids.

Machine learning takes on signature generation and automatic database updates—without human intervention.

Multi-Level Systems.

They apply different methods at different stages of analysis and correlate events from multiple detectors. This improves detection reliability.

To better understand the difference between traditional approaches and those using AI, Table 1 is presented below. It clearly shows the key differences.

Table 1: Comparative analysis of systems using artificial intelligence and traditional methods

Criterion	Systems Using Neural Networks	Traditional Systems
Detection Accuracy	High (85-95%), capable of detecting complex and previously unknown attacks	Medium (70-85%), effective against known attack patterns
Detection of New Threats	Excellent, due to generalization ability and identification of hidden patterns	Limited, requiring constant signature updates
Adaptability to Traffic Changes	High, automatic adjustment to changes in network behavior	Low, requiring manual configuration and rule updates
Data Processing Speed	High after training, but requires significant resources during training	Very high, optimized for stream processing
False Positives	Low percentage (5-10%) after quality training	High percentage (15-25%)
Computational Resource Requirements	High	Moderate
Effectiveness Against Targeted Attacks	Very high	Medium
Example Solutions	Darktrace, Vectra AI, ExtraHop	SNORT, Suricata, Zeek

The estimates provided are based on a synthesis of data from sources (Abu Bakar, Kijsirikul, 2023; Lapina et al., 2025; Lin et al., 2025; Sangeen et al., 2025).

According to research (Abu Bakar, Kijsirikul, 2023; Alazawy, 2024; Almoabady et al., 2025; Lin et al., 2024; Miranda et al., 2025; Sangeen et al., 2025; Sharafaldin et al., 2018), the implementation of neural network-based systems involves higher initial investments compared to traditional solutions. However, they can provide a deeper level of protection and reduce operational costs associated with incident handling. Such systems are most in demand in large organizations with complex network infrastructure—in environments where the potential damage from a successful cyberattack may exceed implementation costs.

Traditional systems remain a cost-effective solution for medium-sized businesses and organizations with typical network architecture. They provide a good baseline level of protection at significantly lower costs but require greater human involvement to maintain relevance.

PORT SCANNING DETECTION USING ARTIFICIAL INTELLIGENCE METHODS

Modern analysis tools must employ advanced machine learning algorithms to identify complex and previously unknown threats. The system must use both supervised learning models for classification and unsupervised learning algorithms for detecting anomalies and new attack vectors. Implementation of continuous learning mechanisms and model adaptation to changing network conditions without the need for complete retraining is required.

The system must automatically extract and analyze hundreds of network activity features, including temporal characteristics, statistical distributions, entropy indicators, and behavioral patterns. The use of time series processing methods for analyzing network activity dynamics and identifying long-term trends is mandatory. The system must provide correlation of events from various sources and construction of relationship graphs between network objects.

The developed software tool is a comprehensive solution for detecting port scanning in computer networks, built on the principles of hybrid analysis combining machine learning methods and heuristic algorithms. Architecturally, the program consists of two main modules functioning within a single ecosystem. The first module is responsible for training models on historical network traffic data, the second for real-time network monitoring with result visualization.

Training Module

The module configuration includes specifying paths to directories containing data and saved models. The program automatically creates necessary folders to ensure correct functioning. The training module is based on the load_and_preprocess_file function, which loads CSV traffic files and prepares them for further analysis. Its algorithm works as follows: first, data is read with the appropriate encoding; then the function checks which of the predefined feature columns are present in the file. Subsequently, cleaning is performed: missing values and duplicates are removed. Finally, data is labeled according to its traffic type.

Training is initiated from the program's main block. Here, the program sequentially iterates through all specified files, determining the traffic type for each. Once all data is loaded, it is combined into a single dataset. The system then verifies that all required classes are present.

Data preparation for training is a multi-stage process including several important steps:

- **Label encoding.** Text labels are converted to numbers using LabelEncoder.
- **Data cleaning.** Infinite values are replaced, and missing values in numeric columns are filled with means.
- **Category encoding.** Categorical variables are also encoded.
- **Normalization.** Numeric data is scaled to a uniform range using MinMaxScaler.

The Random Forest model itself is configured with standard parameters: n_estimators=100, max_depth='auto', random_state=42. Class balancing is performed using the class_weight='balanced' parameter, as the proportion of anomalous traffic in real networks is small (less than 1%).

The Isolation Forest model, used in the monitoring module, is configured with the parameter contamination='auto', which allows automatic estimation of the anomaly proportion in the data stream. After model training, anomaly prediction with label transformation is performed. Model quality assessment is conducted using classification metrics, including precision, recall, and F1-score. The resulting report is saved to a text file, and the model itself, along with auxiliary objects, is saved for subsequent use in the monitoring module.

Monitoring Module is implemented as a NetworkMonitorApp class that encapsulates functionality for capturing and analyzing network traffic in real time. The class initializes main variables for tracking monitoring status, including activity flags, packet counters, and connection statistics.

The graphical interface is built using the Tkinter library and organized into several logical areas:

- A control panel containing start/stop monitoring buttons, real-time statistics display, detection settings, and network interface selection;
- A traffic monitoring area displaying detailed information about each captured packet in a table format with columns: time, source, destination, protocol, and other parameters;
- A log panel recording all system events with timestamps.

For each packet, headers are analyzed, source and destination addresses, port numbers, flags, and temporal characteristics are extracted. Special attention is paid to TCP packets, as most modern scanning attacks use this protocol. The system maintains detailed statistics for each network connection, using a defaultdict data structure to store information about unique ports, packet counts, temporal characteristics, and port access rates.

The following heuristics with threshold values (empirically determined on a testbed) are used:

1. **Unique port threshold:** ≥ 100 unique ports from a single source within 60 seconds.
2. **Scanning rate:** ≥ 10 new ports per second.
3. **Interval regularity:** coefficient of variation of inter-packet intervals ≤ 0.2 (characteristic of automated scanners).
4. **TCP flags:** proportion of packets with SYN flag without subsequent ACK ≥ 80% (indicator of SYN scanning).

Periodic connection checks are performed at specified time intervals, allowing detection of slow attacks that might remain unnoticed during stream processing. Detection of distributed scanning (from multiple sources to a single target) is not implemented in the current version and requires development of additional event correlation mechanisms.

Network packet processing is carried out in a separate thread to ensure graphical interface responsiveness. The packet_handler function processes each packet: extracts features, updates connection statistics, and checks for scanning indicators. Interface updates are performed using

the root.after mechanism, ensuring thread-safe interaction between background processes and the graphical interface.

Traffic visualization is implemented through a Treeview component, displaying detailed information about each packet. The system supports limiting the number of displayed records to prevent interface overload. When a scanning attack is detected, a detailed incident description is generated, including the attack source and target, number of affected ports, scanning duration, and list of scanned ports.

System configurability is one of its key advantages. The user can adjust threshold values for detection algorithms through a settings dialog, allowing system adaptation to specific network conditions and security requirements. Network interface selection for monitoring is supported, with automatic detection of available interfaces through the Scapy library.

A distinctive feature of the program is the integration of machine learning methods with traditional heuristic approaches. Unlike the Random Forest-based training module described above, the monitoring module uses an Isolation Forest model, which is loaded at application startup and applied for real-time anomaly detection. Heuristic rules, in turn, provide detection of known scanning patterns. This hybrid approach achieves high detection accuracy with a minimal number of false positives. The combination of statistical methods and machine learning algorithms produces a synergistic effect: the reliability of detecting complex and camouflaged attacks increases.

Experiments have shown that the developed system effectively handles different types of scanning attacks—TCP SYN scanning, slow scanning, and attacks using techniques to evade detection systems.

Ultimately, the program represents a practical network security monitoring tool in which modern traffic analysis approaches are combined with a user-friendly interface for rapid response.

Practical Application and System Implementation

The developed system can be used in various organizations—both the requirements and benefits will differ.

In Educational Institutions the system addresses several tasks. First, it protects research network infrastructure. Second, it monitors laboratory networks: these are isolated from the main network to prevent accidental or intentional scanning. Third, the system itself can be used in the educational process—as a clear example of network security principles. Finally, it helps defend against internal threats.

In the Corporate Sector application areas differ. These include network segmentation and access control, as well as ensuring compliance with industry standards (e.g., PCI DSS or ISO/IEC 27001). Additionally, the system is suitable for protecting remote workstations and ensuring supply chain security.

A comparative analysis of the proposed method with Russian analogues is presented in Table 2.

Table 2: Comparative analysis of the proposed method with analogues

Criterion	Proposed Tool	Kaspersky Security Center	InfoWatch Traffic Monitor
Detection Methods	Random Forest, Isolation Forest, heuristics (4 rules)	Signature analysis, behavioral models	Statistical analysis, DLP engine
Slow Scanning Detection	Yes (time window analysis)	Yes	Limited
Distributed Scanning Detection	No (future work)	Yes (event correlation)	No
Real-time Operation	Yes (Scapy, 100 Mbit/s limit)	Yes (industrial-grade)	Yes (industrial-grade)
Code Openness / Customization	Full (Python)	No (closed)	No (closed)
Expertise Requirements	High (ML knowledge required)	Medium	Medium

Based on the presented comparative table, the competitive advantages of the proposed port scanning detection method can be identified.

Experimental Evaluation

Dataset

The publicly available CIC-IDS-2017 dataset was used as the basis for experiments. It contains labeled traffic records—both normal and attack traffic. In particular, it includes port scanning using Nmap, which is important for our task. Four days of recording were selected: Monday (normal), Tuesday (attacks), Wednesday-Thursday (various scenarios).

Metrics Used

- **Precision** = TP / (TP + FP)
- **Recall** = TP / (TP + FN)
- **F1-score** = 2 * (Precision * Recall) / (Precision + Recall)

RESULTS

Model	Precision	Recall	F1
Heuristics only (thresholds)	0.78	0.72	0.75
Isolation Forest only	0.71	0.85	0.77
Random Forest only	0.84	0.80	0.82
Proposed method (heuristics + IF + RF)	0.91	0.88	0.90

Error Analysis

False positives (10% of total alerts) occurred on:

- Legitimate torrent client traffic (many simultaneous connections)
- Background activity of monitoring systems (Zabbix, Nagios)

Missed attacks (12%) are associated with extremely slow scanning (1 port per 10 seconds), which did not exceed the heuristic thresholds.

Comparison with Analogues (based on known sources)

Method	F1-score
XGBoost algorithm	0.88
Stream data analysis	0.86
Proposed method	0.90

CONCLUSION

This work proposes a hybrid method for port scanning detection, combining heuristic rules, Random Forest, and Isolation Forest. Experimental evaluation on the CIC-IDS-2017 dataset showed an F1-score of 0.90, which is 5-8% higher than using each method individually.

Limitations of the current implementation:

1. Lack of distributed scanning detection.
2. Low performance when capturing traffic > 100 Mbit/s.

Directions for future research:

1. Integration of graph-based methods for tracking distributed attacks.
2. Use of high-performance capture libraries (PF_RING, DPDK).
3. Adaptive threshold adjustment based on analysis of normal traffic of a specific network.

REFERENCES

Abu Bakar R, Kijsirikul B. Enhancing network visibility and security with advanced port scanning techniques. Sensors 2023,23:7541. https://doi.org/10.3390/s23177541

Alazawy A. Effectiveness of Supervised and Unsupervised Algorithms in Detecting RAPs in Wireless Networks, Master's thesis, National College of Ireland, Dublin, Ireland; 2024. 85 p.

Almoabady TA, Alblawi YM, Albalawi AE, Aborokbah MM, Manimurugan S, Aljuhani A, Aldawood H, Karthikeyan P. Protecting digital assets using an ontology based cyber situational awareness system. Front Artif Intell 2025,7:1394363. https://doi.org/10.3389/frai.2024.1394363

Lapina MA, Podruchny NV, Rusanov MA, Babenko MG. Research of machine learning methods for detecting network attacks. Proc ISP RAS 2025,37:147-74.

Lin Y, Chen Y, Tian H, Zhuang X. Covert timing channel detection based on isolated binary trees. Comput Secur 2025,150:104200. https://doi.org/10.1016/j.cose.2024.104200

Lin Z-Z, Pike TD, Bailey MM, Bastian ND. A hypergraph-based machine learning ensemble network intrusion detection system. **IEEE Trans Syst Man Cybern Syst** 2024,54:6911-23. https://doi.org/10.1109/TSMC.2024.3446635

Miranda D, Monteiro R, Silva JMC. p4SD: A lightweight port scan detection for programmable networks. In: Proceedings of the International Conference on Software, Telecommunications and Computer Networks (SoftCOM 2025), Split, Croatia, September 18-20, 2025. New York: IEEE; 2025. p. 1-6. https://doi.org/10.23919/SoftCOM66362.2025.11197439

Sangeen M, Bhatti NA, Kifayat K. PortScout: A communication flow-based approach to detect port scanning evasion attacks. In: Proceedings of IEEE International Conference on Communications (ICC 2025), Montreal, QC, Canada, June 8-12, 2025. New York: IEEE; 2025. p. 3045-50. https://doi.org/10.1109/ICC52391.2025.11160775

Sharafaldin I, Lashkari AH, Ghorbani AA. Toward generating a new intrusion detection dataset and intrusion traffic characterization. In: Proceedings of the 4th International Conference on Information Systems Security and Privacy (ICISSP 2018). Funchal: SciTePress; 2018. p. 108–16. https://doi.org/10.5220/0006639801080116