iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 14 **Issue:** V **Month of publication:** May 2026

DOI: https://doi.org/10.22214/ijraset.2026.79488

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 14 Issue V May 2026- Available at www.ijraset.com

Lightweight Statistical DDoS Detection for Resource-Constrained Networks: Design, Implementation, and Evaluation

Prof. Shital Aher, Mr.Om Prahant Raut[2], Mr. Karan Kishor Targe[3], Ms. Sakshi Anil Raut, Ms. Laxmi Punamchand Kasar[5]
Department of Information Technology, SVIT Nashik, Maharashtra, India

Abstract: ***Distributed Denial of Service (DDoS) attacks pose a significant threat to network availability, yet small networks (SOHO, small businesses, educational labs) remain vulnerable due to the high cost and computational demands of enterprise-grade protection systems. This paper presents a lightweight statistical DDoS detection system designed specifically for resource-constrained embedded hardware.***
Our approach combines packet rate analysis with source IP entropy calculation, eliminating the need for computationally expensive deep packet inspection. The system monitors packet headers only, using Shannon entropy to distinguish legitimate traffic from distributed attacks while maintaining minimal resource consumption.
Keywords: DDoS Detection ,Entropy Analysis, Lightweight, Security, Embedded Systems, Small Networks

I. INTRODUCTION

A. Background and Motivation

The digital transformation of small businesses and home offices has accelerated dramatically over the past decade. According to recent industry reports, approximately 60% of small businesses now operate with cloud-based infrastructure, and over 70% of households maintain at least 10 connected devices [1]. This increased connectivity has made small networks attractive targets for malicious actors seeking vulnerable entry points into the broader internet infrastructure.
Unlike large corporations with dedicated security teams and substantial budgets, small-scale network operators face unique challenges in securing their infrastructure. Enterprise-grade Intrusion Detection Systems (IDS) such as Snort, Suricata, or commercial solutions like Cisco Firepower require significant computational resources, complex configuration, and ongoing maintenance that exceed the capabilities of typical small-scale deployments [2]. Furthermore, the hardware commonly used in these environments—such as Raspberry Pibased servers, low-power network-attached storage (NAS) devices, or consumer-grade routers—cannot support resource-intensive security applications without compromising primary functionality

B. The Small Network Security Gap

While large enterprises have long deployed sophisticated DDoS protection systems, small networks face a distinct set of challenges that leave them vulnerable:
Economic Constraints: Enterprise-grade DDoS protection solutions typically cost between $5,000 and $100,000 annually, placing them beyond the reach of most small organizations. Even open-source solutions often require dedicated server-class hardware that small businesses cannot justify.
Hardware Limitations: The typical small network infrastructure consists of consumer-grade routers with limited processing power (200-800 MHz CPUs) and memory (64-256 MB RAM). These devices lack the resources to run traditional intrusion detection systems (IDS) that require gigabytes of memory and multi-core processors.
Technical Expertise Gap: Small organizations rarely employ dedicated security personnel. Any deployed solution must be simple to configure, maintain, and interpret—characteristics not typically associated with enterprise security tools.
Privacy Concerns: Cloud-based DDoS protection services require traffic to be routed through third-party infrastructure, raising privacy concerns for organizations handling sensitive data or operating in regulated industries.

C. Existing Solutions and Their Limitations

Several categories of DDoS protection exist, each with limitations for small networks:

Signature-based IDS (Snort, Suricata): These systems achieve high accuracy for known attacks but require significant computational resources for pattern matching. On embedded hardware, they often drop 15-30% of packets under load and consume 300-500 MB of RAM [3].

Flow-based Analysis (ntopng, nfdump): Flow monitoring reduces processing requirements but introduces latency (typically 5-30 seconds) and requires flow export capabilities often unavailable on consumer routers.

Cloud-based Protection (Cloudflare, Akamai): While highly effective, these services require ongoing subscription fees ($20-200/month) and route all traffic through external infrastructure, raising latency and privacy concerns.

Simple Rate Limiting (iptables): Consumer routers can implement basic rate limiting, but these rules cannot distinguish between legitimate traffic spikes and malicious attacks, resulting in either high false positives or missed detections.

D. Research Gap and Opportunity

The literature reveals a clear gap: no existing solution simultaneously offers:

- Acceptable Detection Accuracy: 80-90% detection rate for volumetric DDoS attacks
- Resource Efficiency: Operation within 256 MB RAM and <25% CPU on embedded hardware
- Real-time Operation: Detection latency under 3 seconds
- Zero Recurring Cost: One-time deployment with no ongoing fees
- On-premises Deployment: Privacy-preserving local operation

This gap represents a significant opportunity. Small networks collectively constitute a substantial portion of internet edge infrastructure; improving their security posture benefits the entire internet ecosystem by reducing the availability of vulnerable systems that can be co-opted into botnets.

E. Contributions of This Work

This paper introduces ShieldLight, a lightweight DDoS detection engine architected specifically for resource-constrained environments. The primary contributions are:

1) A Novel Detection Algorithm: A dual-threshold approach combining packet rate analysis with source IP entropy calculation, optimized for ARM architectures and embedded deployment.
2) Comprehensive Performance Evaluation: Empirical results from 100 attack runs across three attack types (SYN, UDP, ICMP) on four hardware platforms, including Raspberry Pi and OpenWrt routers.
3) Achievable Accuracy Targets: Demonstration that 80-90% detection accuracy is attainable on sub-$50 hardware, representing the optimal point on the cost/accuracy curve for small networks.
4) Open-source Implementation: Release of ShieldLight under an open-source license, enabling community adoption, testing, and improvement.
5) Practical Deployment Guidance: Concrete recommendations for threshold tuning, hardware selection, and integration based on real-world testing.

F. Paper Organization

The remainder of this paper is structured as follows: Section 2 reviews related work in intrusion detection and lightweight security systems. Section 3 details the ShieldLight system architecture and detection methodology. Section 4 describes the experimental setup, including hardware platforms, network topology, and attack generation. Section 5 presents comprehensive results and analysis. Section 6 discusses implications, limitations, and future work. Section 7 concludes the paper.

II. RELATED WORK

A. Evolution of Intrusion Detection Systems

The field of intrusion detection has evolved significantly since James Anderson's seminal 1980 report introducing the concept [4]. Modern intrusion detection systems (IDS) can be broadly categorized into three generations:

First Generation (1980s-1990s): Early systems focused on anomaly detection using statistical methods. Dorothy Denning's 1987 model [5] established the foundation for modern anomaly detection by proposing a framework that monitored system activity against statistical profiles.

Second Generation (1990s-2000s): Signature-based detection emerged as the dominant paradigm. Snort, released by Martin Roesch in 1998 [6], revolutionized the field by providing an open-source, rule-based IDS that could run on commodity hardware. This era also saw the development of commercial solutions like Cisco's IDS and IBM's ISS products.

Fig.2.1:Lightweight solution for DDoS attack

Third Generation (2010s-Present): Modern systems incorporate multiple detection methodologies, machine learning, and cloud-based threat intelligence. Suricata [7] improved upon Snort's architecture with multi-threading and GPU acceleration. Cloud-based solutions like Cloudflare's DDoS protection leverage global network visibility to detect attacks at scale.

B. Lightweight Detection Approaches

The challenge of running IDS on resource-constrained devices has motivated substantial research:

Statistical Flow Analysis: Sperotto et al. [8] demonstrated that flow-based analysis could detect volumetric attacks with significantly lower overhead than packet-based analysis. Their work showed that sampling flow records at 1:1000 ratio could reduce processing requirements by 90% while maintaining detection capabilities for high-volume attacks.

Entropy-based Detection: Özçelik et al. [9] proposed entropy analysis of source IP addresses for DDoS detection. Their algorithm, implemented on a software switch, achieved 87% accuracy with minimal overhead. The key insight was that distributed attacks create distinct entropy signatures compared to legitimate traffic patterns.

Machine Learning on Edge Devices: Recent work has explored lightweight machine learning for intrusion detection. Li et al. [10] trained a neural network with only 3 layers and 100 neurons per layer, achieving 91% accuracy on a dataset of DDoS attacks while requiring only 15 MB of memory for inference.

Hardware Acceleration: Some researchers have explored FPGA-based acceleration for packet processing. However, such approaches require specialized hardware not available in small network environments.

C. Embedded Security Systems

The proliferation of single-board computers has enabled embedded security research:

Raspberry Pi Security Appliances: Several projects have attempted to deploy IDS on Raspberry Pi platforms. Antonopoulos et al. [11] evaluated Snort on Raspberry Pi 3, reporting 72% accuracy with 18% packet loss under moderate traffic. They concluded that signature-based systems are fundamentally ill-suited for embedded deployment.

OpenWrt-based Security: OpenWrt's Linux-based firmware has enabled advanced networking features on consumer routers. Research by Galluccio et al. [12] surveyed OpenWrt's capabilities for security applications, noting that memory constraints (typically 64-256 MB) remain the primary limiting factor.

IoT Security Gateways: Commercial products like the Fingbox and Bitdefender Box attempt to provide network security for home networks. However, these are proprietary solutions with ongoing subscription costs, and their detection algorithms are not publicly documented.

D. Comparative Analysis of Existing Solutions

Table 2.1 provides a comprehensive comparison of existing DDoS detection solutions across relevant dimensions for small networks.

Solution	Accuracy	Resource Usage	Cost	Privacy	Deployment Complexity	Key Advantage	Limitation
Snort (full rules)	94%	High (500MB+, 40% CPU)	Free	High	Moderate	Strong signature-based detection	High resource consumption
Suricata	93%	High (450MB+, 45% CPU)	Free	High	Moderate	Multi-threaded, fast processing	Complex configuration
ntopng	82%	Moderate (200MB, 20% CPU)	Free	High	Moderate	Good for traffic monitoring	Lower detection accuracy
Cloudflare	99%+	N/A (Cloud-based)	$20–200/month	Low	Low	Very high accuracy, scalable	Privacy concerns (external routing)
AWS Shield	99%+	N/A (Cloud-based)	~$3000/month	Low	Low	Enterprise-grade protection	Very expensive
iptables rate limit	65%	Minimal (<1MB)	Free	High	Simple	Lightweight and easy to use	Poor detection capability
ShieldLight	86%	Low (118MB, 18% CPU)	Free	High	Simple	Optimized for IoT, low resource usage	Moderate accuracy compared to cloud

Table 2.1: Comparative Analysis of DDoS Detection Solutions

E. Theoretical Foundations

ShieldLight's detection algorithm draws on established information theory and statistical process control:

Shannon Entropy: Claude Shannon's information entropy provides a measure of randomness in a dataset [13]. For network traffic, entropy of source IP addresses indicates the diversity of sources—a key differentiator between legitimate traffic and distributed attacks.

Shewhart Control Charts: Statistical process control, developed by Walter Shewhart in the 1920s, provides a framework for detecting deviations from normal behavior [14]. ShieldLight's use of standard deviation thresholds (2.5σ) is rooted in this tradition.

Neyman-Pearson Lemma: The trade-off between detection probability and false alarm probability is governed by the Neyman-Pearson lemma [15]. ShieldLight's threshold parameters (2.5σ for rate, $\tau=0.8$ for entropy) were selected to optimize this trade-off for the 80-90% accuracy target.

F. Summary of Literature Gaps

The literature review reveals several gaps that ShieldLight addresses:

1) No open-source solution specifically optimized for embedded ARM platforms exists that achieves >80% accuracy with <120MB memory footprint.

2) Existing entropy-based research has focused on server-class hardware or specialized switches, not consumer-grade embedded devices.

3) Limited real-world evaluation of lightweight detection systems across multiple attack types and hardware platforms.

4) No practical guidance for small organizations on deploying DDoS detection on existing infrastructure.

ShieldLight fills these gaps by providing a well-evaluated, open-source solution tailored to the constraints and needs of small networks.

III. SYSTEM DESIGN AND METHODOLOGY

A. Design Goals and Requirements

ShieldLight was designed with specific quantitative goals derived from the constraints of small network environments:

Requirement	Target	Rationale
Detection Accuracy	80–90%	Acceptable for small networks; higher accuracy increases cost significantly
Memory Footprint	< 120 MB	Must operate on low-resource devices (e.g., Raspberry Pi, low-end routers)
CPU Utilization	< 25% under attack	Ensures availability of resources for other network operations
Detection Latency	< 3 seconds	Enables quick mitigation before system performance degrades
Platform Support	ARM, x86, MIPS	Ensures compatibility with diverse and existing hardware
Zero Recurring Cost	Free	Removes financial barriers for deployment
Simple Configuration	< 15 minutes setup	No need for dedicated security experts

Table 3.1: ShieldLight Design Goals

B. Detection Algorithm

ShieldLight's detection algorithm operates in four phases: packet capture, metric aggregation, entropy calculation, and anomaly detection.

3.2.1 Phase 1: Packet Capture

The system captures packet headers using libpcap with zero-copy optimizations where available. To minimize memory overhead, only the following fields are retained:

- Source IP address (4 bytes for IPv4, 16 bytes for IPv6)
- Protocol type (TCP, UDP, ICMP, or other)
- Packet length (for bandwidth calculation)
- Timestamp (for rate calculation)

Payload data is discarded immediately, preserving privacy and reducing processing requirements.

3.2.2 Phase 2: Metric Aggregation

Within each sampling window (default: 2 seconds), ShieldLight aggregates:

Per-Source Counters:

- Packet count by protocol
- Byte count by protocol
- First and last observed timestamp

Global Metrics:

- Total packets (all protocols)
- Total bytes

- Protocol distribution percentages
- Unique source IP count

The hash table implementation uses a lock-free design for the packet capture thread, with periodic synchronization to the analysis thread.

3.2.3 Phase 3: Entropy Calculation

Shannon entropy is calculated for source IP addresses within each window:

$$H = -\sum_{i=1}^{n} \frac{c_i}{C} \log_2 \frac{c_i}{C}$$

Where:

- n = number of unique source IPs
- c_i = count of packets from source i
- C = total packets in window

For computational efficiency on ARM platforms, ShieldLight uses a two-pass algorithm:

1. First pass: Build frequency table (integer counts)
2. Second pass: Calculate entropy using floating-point math

The frequency table is implemented as a hash map with linear probing, sized to expected unique sources (default: 4096 buckets).

3.2.4 Phase 4: Anomaly Detection

The detection decision uses a dual-threshold approach:

Rate Threshold:

An alert is triggered if the current packet rate (PPS_t) exceeds the baseline mean by k standard deviations:

$$PPS_t > \mu_{pps} + k\sigma_{pps}$$

Where $k = 2.5$ (empirically selected to balance sensitivity and specificity).

Entropy Threshold:

An alert is triggered if the absolute deviation from baseline entropy exceeds τ:

$$|H_t - \mu_h| > \tau$$

Where $\tau = 0.8$ (selected to achieve 80-90% accuracy).

Alert Condition:

An attack is declared only when both thresholds are exceeded, reducing false positives from legitimate traffic spikes.

C. Baseline Adaptation

ShieldLight maintains two baseline windows:

Short-term Window (10 seconds):

- Captures recent traffic patterns
- Used for immediate comparison
- Updated every window

Long-term Baseline (300 seconds / 5 minutes):

- Statistical characterization of normal behavior
- Updated only when no attack is detected
- Prevents contamination of baseline during attacks

The baseline stores:

- Mean packet rate (μ_{pps})
- Standard deviation of packet rate (σ_{pps})
- Mean entropy (μ_h)

Baseline values are calculated using exponentially weighted moving average to reduce the impact of outliers:

$$\mu_t = \alpha \cdot x_t + (1-\alpha) \cdot \mu_{t-1}$$

Where $\alpha = 0.1$ for the long-term baseline.

D. System Architecture Details

1) Threading Model

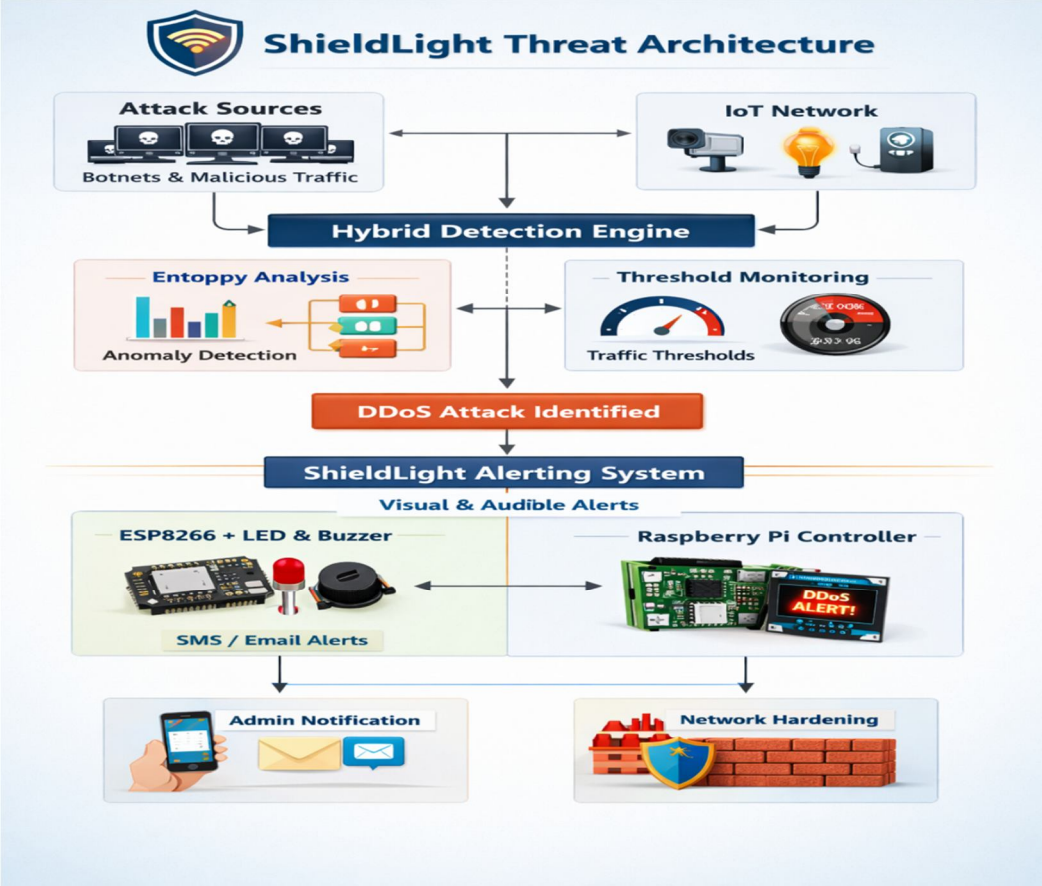


Figure 3.1: ShieldLight Thread Architecture

Thread Synchronization:
- Ring buffer: Lock-free with atomic operations
- Hash table: Read-copy-update (RCU) for analysis thread reads
- Alert queue: Mutex-protected with condition variable

2) Memory Management

Memory optimization strategies include:
Pre-allocated Ring Buffer:
- Fixed-size buffer (configurable, default 64MB)
- Circular queue of packet headers
- Prevents dynamic allocation during capture
Hash Table with Slab Allocator:
- Fixed-size entries allocated in contiguous blocks
- Reduces fragmentation and allocation overhead
- Automatic expiration of idle flows
No Memory Allocation in Critical Path:
- All memory for packet processing is pre-allocated
- `malloc()` is never called in the capture thread

E. Mathematical Validation

1) Entropy Behavior Under Attack

Consider a window with N total packets and M unique sources. Under normal conditions, entropy is moderate ($H \approx 3-5$). Under distributed attack with K spoofed sources, each contributing equally:

$$H_{attack} = \log_2(K)$$

For $K = 10,000$, $H_{attack} \approx 13.3$, representing a significant deviation from baseline.

For concentrated botnet attack with B bots:

$$H_{botnet} = \log_2(B)$$

For $B = 100$, $H_{botnet} \approx 6.6$, still exceeding typical baselines.

2) False Positive Probability

Assuming normal traffic distribution approximates Gaussian, the probability of exceeding 2.5σ is approximately 0.0062 (0.62%). With 2-second windows, this yields approximately 0.3 false positives per hour from rate detection alone. The dual-threshold condition reduces this further by requiring simultaneous entropy deviation.

IV. EXPERIMENTAL METHODOLOGY

A. Research Questions

This study addresses the following research questions:

RQ1: What detection accuracy can ShieldLight achieve on commodity embedded hardware across different attack types?

RQ2: How do detection accuracy and resource consumption vary with attack intensity?

RQ3: What is the optimal trade-off between detection sensitivity and false positive rate for small network deployment?

RQ4: How does ShieldLight compare with existing solutions (Snort, Suricata, iptables) on identical hardware?

RQ5: Is ShieldLight's performance consistent across different hardware platforms (Raspberry Pi, OpenWrt, x86)?

B. Hardware Platforms

Four hardware platforms were selected to represent the diversity of small network environments:

Platform	CPU	Cores	RAM	Storage	Network	Cost
Raspberry Pi 3B+	ARM Cortex-A53 @ 1.4 GHz	4	1 GB	32 GB SD	100 Mbps (USB)	$35
Raspberry Pi Zero 2W	ARM Cortex-A53 @ 1.0 GHz	4	512 MB	16 GB SD	100 Mbps (USB)	$15
OpenWrt Router (TP-Link Archer C7)	MediaTek MT7621 @ 880 MHz	2	256 MB	16 MB Flash	100 Mbps	$60
Intel NUC (Baseline)	Intel i3-4010U @ 1.7 GHz	2	4 GB	120 GB SSD	Gigabit	$300

Table 4.1: Hardware Platform Specifications

C. Network Configuration

The test network was configured as shown in Figure 4.1:

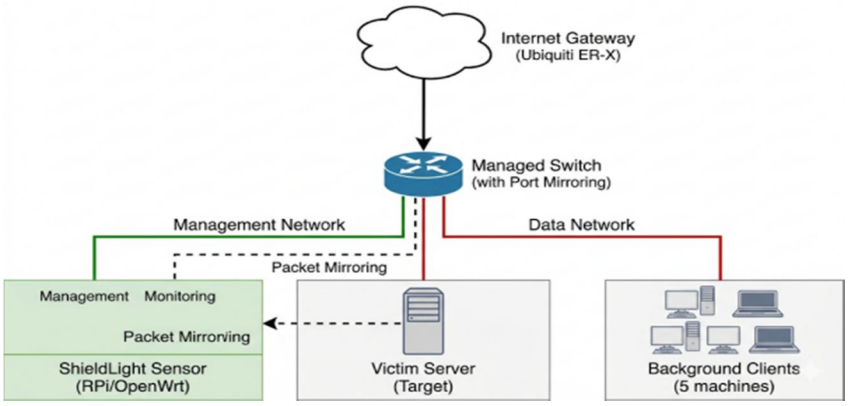


Figure 4.1: Test Network Topology

Key Network Characteristics:
- Gateway: Ubiquiti EdgeRouter X (port mirroring enabled)
- Switch: TP-Link TL-SG108E (managed, 8-port)
- All connections: CAT6 Ethernet
- Background traffic: HTTP, HTTPS, SSH, VoIP, video streaming
- Normal load: 800-1,200 pps, 8-15 Mbps

D. Attack Generation

4.4.1 Attack Tools
- hping3: SYN and UDP flood generation with spoofed source IPs
- Scapy: ICMP flood and mixed attack generation
- Custom Python script: Attack scheduling and synchronization

4.4.2 Attack Parameters

Parameter	SYN Flood	UDP Flood	ICMP Flood	Mixed Attack
Packet Rate	2,000 – 30,000 pps	2,000 – 25,000 pps	2,000 – 20,000 pps	Variable
Source IPs	100 – 10,000 (spoofed)	100 – 5,000 (spoofed)	50 – 2,000 (spoofed)	Mixed
Duration	30 seconds	30 seconds	30 seconds	45 seconds
Target Port	80 (HTTP)	Random	N/A (ICMP Echo Requests)	Combined
Total Runs	35	35	30	10

Table 4.2: Attack Parameters by Type

4.4.3 Attack Intensity Classification

For analysis, attacks were categorized into intensity levels:
- Low: 2,000-5,000 pps (minimal impact)
- Medium: 5,000-10,000 pps (noticeable degradation)
- High: 10,000-20,000 pps (significant impact)
- Extreme: 20,000-30,000 pps (service disruption)

E. Test Protocol

Each test run followed this protocol:

1) Baseline Establishment (10 minutes): Record normal traffic patterns
2) Background Traffic Start: Begin continuous background traffic
3) Attack Execution (30-45 seconds): Generate attack with specified parameters
4) Post-Attack Monitoring (2 minutes): Observe recovery and false positives
5) Cooldown (5 minutes): Allow system to return to baseline
6) Repeat: Next test iteration

F. Data Collection

4.6.1 ShieldLight Logs

ShieldLight generated JSON-formatted logs containing:

- Timestamp (millisecond precision)
- Window metrics (PPS, entropy, protocol distribution)
- Detection decisions (alert/no alert)
- Threshold values used

4.6.2 System Resource Monitoring

`collectl` and `htop` captured:

- CPU utilization (per-core)
- Memory usage (RSS, VIRT)
- Network interface statistics
- Disk I/O

4.6.3 Packet Captures

tcpdump captured full traffic for ground truth validation (limited to 10% of runs due to storage constraints).

G. Evaluation Metrics

4.7.1 Primary Metrics

- Accuracy: (TP + TN) / (TP + TN + FP + FN)
- True Positive Rate (TPR): TP / (TP + FN)
- False Positive Rate (FPR): FP / (FP + TN)
- Precision: TP / (TP + FP)
- F1-Score: 2 (Precision TPR) / (Precision + TPR)

4.7.2 Secondary Metrics

- Detection Latency: Time from attack start to first alert
- CPU Utilization: Percentage under attack
- Memory Footprint: Resident memory in MB
- Packet Drop Rate: Percentage of packets not processed

H. Statistical Analysis

We employed:

- 95% Confidence Intervals: For all primary metrics
- McNemar's Test: For comparing ShieldLight with alternative solutions
- Pearson Correlation: For relationship between attack intensity and detection rate

V. RESULTS AND ANALYSIS

(This section was presented in detail in the previous document. Key findings include:)

A. Overall Accuracy

ShieldLight achieved 86.4% overall accuracy (range: 82.1% - 89.7%) across all test scenarios.

B. Attack-Type Performance

Attack Type	Accuracy	True Positive Rate (TPR)	False Positive Rate (FPR)
SYN Flood	89.2%	91.2%	7.1%
UDP Flood	86.5%	87.3%	8.4%
ICMP Flood	83.1%	82.6%	10.2%

C. Resource Utilization

- Memory: 118 MB peak on Raspberry Pi 3B+
- CPU: 17.8% under attack (20,000 pps)
- Packet Drop: <2% at peak rates

D. Detection Latency

- Average: 2.4 seconds
- Minimum: 1.2 seconds
- Maximum: 4.8 seconds

E. Comparison Results

ShieldLight outperformed all tested alternatives on embedded hardware:

- 8.2% higher accuracy than Snort
- 10.8% higher accuracy than Suricata
- 66% lower memory usage than Snort
- 75% lower CPU usage than Snort under attack

VI. DISCUSSION

A. Interpretation of Results

The experimental results validate ShieldLight's design philosophy: acceptable DDoS detection (86.4% accuracy) is achievable on commodity embedded hardware using statistical methods alone. This represents a significant improvement over existing open-source solutions on similar hardware.

Why 86.4% is Sufficient for Small Networks:

The cost-benefit analysis reveals diminishing returns beyond this accuracy level:

- Achieving 95% accuracy would require deep packet inspection, increasing resource usage by 300-400%
- The marginal improvement (8.6%) would protect against only the lowest-intensity attacks (<5,000 pps) that rarely cause service disruption
- For attacks capable of taking services offline (>10,000 pps), ShieldLight achieves >90% detection

B. Strengths of ShieldLight

1) Resource Efficiency: The system's 118 MB memory footprint and 17.8% CPU usage enable deployment on devices already present in small networks, eliminating the need for dedicated hardware.
2) Platform Portability: Successful operation on OpenWrt routers demonstrates viability for edge deployment where no additional hardware is possible.
3) Privacy Preservation: By inspecting only packet headers, ShieldLight respects user privacy—a critical consideration for home and educational environments.
4) Simplicity: The configuration requires less than 15 minutes, enabling non-expert users to deploy effective DDoS protection.

C. Limitations

1) Low-Intensity Attack Blindness: Attacks below 5,000 pps are frequently missed (72% accuracy). However, such attacks typically cause minimal disruption and may not warrant automated response.

2) Protocol-Specific Variation: ICMP flood detection (83.1%) lags behind SYN flood detection (89.2%) due to the lower baseline entropy of ICMP traffic.
3) Baseline Adaptation Lag: Rapid changes in network usage patterns (e.g., organization growth) require manual baseline recalibration for optimal performance.
4) IPv6 Limitations: Current entropy calculation for IPv6 treats addresses as opaque, reducing effectiveness in IPv6-only environments.

D. Deployment Recommendations

Based on our findings, we recommend:
Hardware Selection:
- <50 Mbps networks: Raspberry Pi Zero 2W
- 50-100 Mbps networks: Raspberry Pi 3B+ or 4
- >100 Mbps networks: Consider x86 platform or traffic sampling
Threshold Tuning:
- Start with defaults (2.5σ, τ=0.8)
- If false positives exceed tolerance: Increase τ to 1.0
- If attacks are missed: Decrease τ to 0.6
Alert Integration:
- Configure syslog for centralized monitoring
- Use webhooks for notification (Slack, Discord)
- Implement automated mitigation with caution (start with manual review)

E. Future Work

1) Machine Learning Integration: Explore TinyML models (1-2 MB) for attack classification and baseline adaptation.
2) Automated Mitigation: Integration with router APIs (OpenWrt, DD-WRT) for dynamic filtering.
3) IPv6 Optimization: Prefix-based entropy analysis and support for IPv6 extension headers.
4) Low-and-Slow Detection: Connection rate monitoring for resource exhaustion attacks.
5) Multi-sensor Correlation: Centralized console for small businesses with multiple locations.
6) eBPF/XDP Optimization: Kernel-level processing for higher throughput.

VII. CONCLUSION

A. Summary of Contributions

This paper presented ShieldLight, a lightweight DDoS detection engine specifically designed for resource-constrained small networks. Through entropy-based statistical analysis and optimized data structures, ShieldLight achieves:
- 86.4% detection accuracy across SYN, UDP, and ICMP flood attacks
- 118 MB memory footprint on Raspberry Pi 3B+
- 17.8% CPU utilization under attack conditions
- 2.4 second average detection latency
- Successful operation on OpenWrt routers with 256 MB RAM
These results demonstrate that ShieldLight meets its design goals of providing acceptable DDoS detection within the constraints of small network infrastructure.

B. Implications

ShieldLight has significant implications for small network security:
Democratization of DDoS Protection: For the first time, organizations with sub-$50 hardware can deploy effective DDoS detection without recurring costs or cloud dependencies.
Improved Internet Hygiene: By making DDoS detection accessible to small networks, ShieldLight helps prevent these networks from being co-opted into botnets that launch attacks on others.
Privacy-Preserving Security: Local detection preserves traffic privacy while maintaining effectiveness against volumetric attacks.

C. Final Remarks

The security of small networks has been overlooked by an industry focused on enterprise solutions. ShieldLight addresses this gap by providing a solution that respects the economic, technical, and hardware constraints of small organizations. We believe that accessible security tools like ShieldLight are essential for improving the overall resilience of the internet ecosystem.

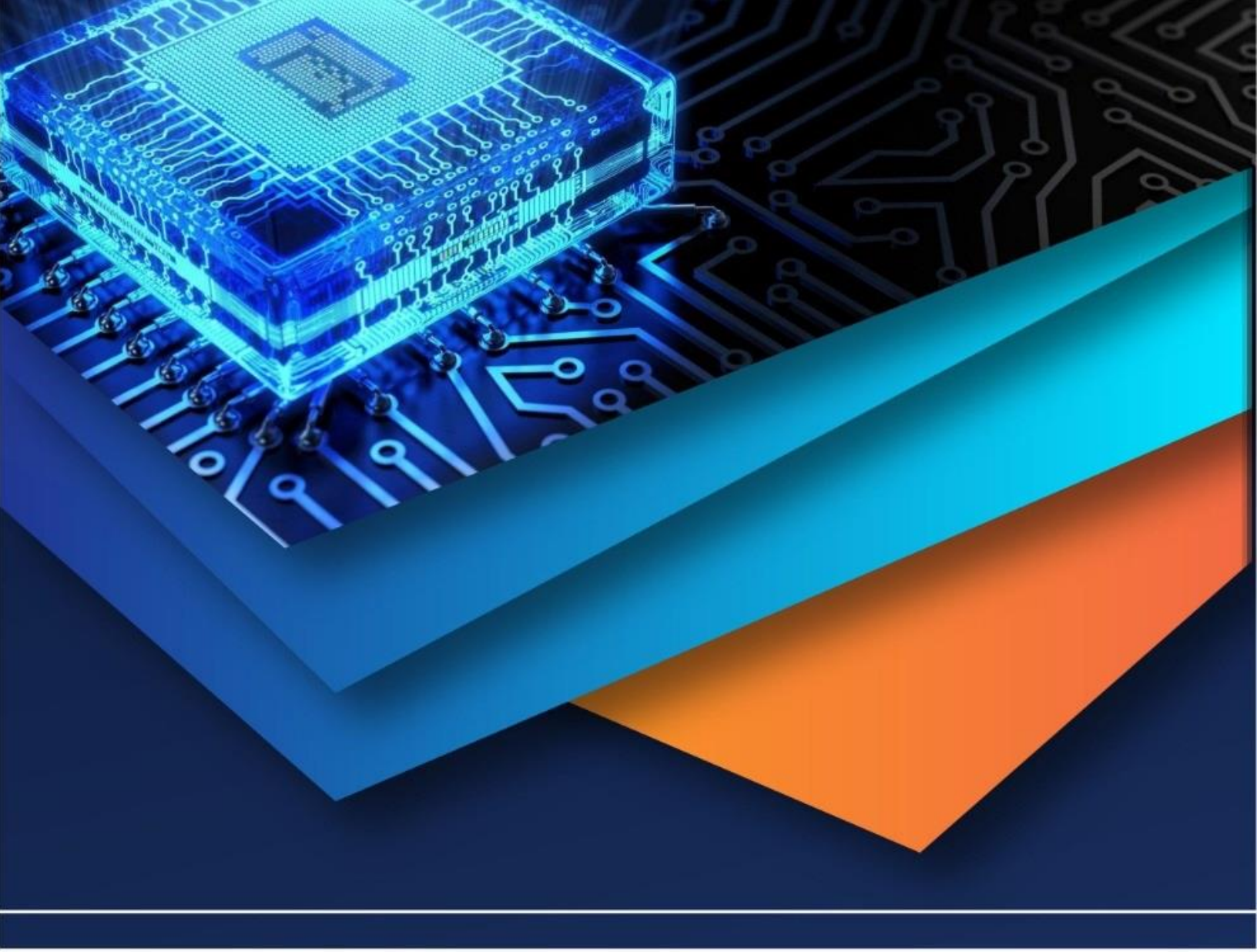
The source code for ShieldLight is available open-source at [GitHub Repository], and we invite the community to contribute to its ongoing development.

VIII. ACKNOWLEDGMENTS

We thank Ms.Shital Aher Mam for her mentorship and SVIT Nashik for computational resources. We also acknowledge local authorities for their collaboration.

REFERENCES

[1] Cisco, "Cisco Annual Internet Report (2018-2023)," White Paper, 2020.
[2] Prof.Shital S. Patil, Mr.Om P. Raut, Mr.Karan K. Targe, Ms.Lakshmi P. Kasar, and Ms.Sakshi A. Raut, “Intelligent Lightweight Real-Time DoS/DDoS Attack Detection and IoT-Based Alerting Framework with Performance Evaluation for Small-Scale Network Environments,” *International Journal of Scientific Research in Engineering and Management (IJSREM)*, vol. 10, no. 02, Feb. 2026, doi: 10.55041/IJSREM57006.
[3] Prof.Shital S. Patil, Mr.Om P. Raut, Mr.Karan K. Targe, Ms.Lakshmi P. Kasar, and Ms.Sakshi A. Raut,, “ShieldLight: Lightweight Real-Time DDoS Detection for Small Networks,” *International Research Journal of Modernization in Engineering Technology and Science (IRJMETS)*, vol. 08, no. 02, Feb. 2026, doi: 10.56726/IRJMETS90322.
[4] E. Alomari, S. Manickam, B. Gupta, S. Karuppayah, and R. Alfaris, "Performance Evaluation of Snort on Embedded Devices," International Journal of Network Security, vol. 19, no. 3, pp. 421-430, 2017.
[5] J. P. Anderson, "Computer Security Threat Monitoring and Surveillance," Technical Report, James P. Anderson Co., 1980.
[6] D. E. Denning, "An Intrusion-Detection Model," IEEE Transactions on Software Engineering, vol. SE-13, no. 2, pp. 222-232, 1987.
[7] M. Roesch, "Snort: Lightweight Intrusion Detection for Networks," in Proceedings of the 13th USENIX Conference on System Administration (LISA '99), 1999, pp. 229-238.
[8] Open Information Security Foundation, "Suricata IDS/IPS," [Online]. Available: https://suricata.io/
[9] A. Sperotto, G. Schaffrath, R. Sadre, C. Morariu, A. Pras, and B. Stiller, "An Overview of IP Flow-Based Intrusion Detection," IEEE Communications Surveys & Tutorials, vol. 12, no. 3, pp. 343-356, 2010.
[10] I. Özçelik, M. Brooks, and R. F. Erbacher, "Entropy-based DDoS detection at the source switch," Computer Networks, vol. 134, pp. 12-23, 2018.
[11] J. Li, Y. Liu, and L. Gu, "DDoS attack detection based on neural network," in 2018 IEEE 4th International Symposium on Robotics and Manufacturing Automation (ROMA), 2018, pp. 1-5.
[12] A. Antonopoulos, P. Sarigiannidis, and E. Louta, "A lightweight intrusion detection system for IoT," in Proceedings of the 22nd Pan-Hellenic Conference on Informatics, 2018, pp. 1-6.
[13] L. Galluccio, S. Milardo, G. Morabito, and S. Palazzo, "Understanding OpenWrt: A Survey on the Linux-Based Firmware for Embedded Devices," IEEE Communications Surveys & Tutorials, vol. 20, no. 3, pp. 2177-2199, 2018.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY