Power System Protection and Control
ISSN:1674-3415

QUANTUM-ADAPTIVE NEURAL DEFENSE: A SELF-LEARNING DDOS MITIGATION FRAMEWORK USING ENTROPY-BASED TRAFFIC FINGERPRINTING AND REAL-TIME BEHAVIORAL ANALYSIS

Mohammad Mohammad[1], Aleem Mohammed[2], Habeeb Vulla[3], Atheeq C[4]

[1]Department of Computer Science, Central Queensland University, Sydney, Australia m.mohammad@cqu.edu.au
[2]Research Associate, Melbourne Institute of Technology, Sydney, Australiaa aleemmohammed99@gmail.com
[3]Department of Engineering, perfai.ai, Santa Clara, California 95050, USA
habeebvulla@gmail.com
[4]Department of Computer Science, Gitam University, Hyderabad, India
Atheeq.prof@gmail.com

Received:15 October 2025 **Revised**:18 November 2025 **Accepted**: 20 December 2025

***ABSTRACT*:**

DDoS attacks are getting smarter and use more than one method, which makes it hard for regular defenses to work. Modern mitigation systems have three big problems: they give a lot of false positives when there are real traffic bursts, they can't learn new attack patterns, and they make the user experience worse by adding a lot of latency. This paper introduces an innovative architecture known as QAND (Quantum-Adaptive Neural Defense), which utilizes adaptive neural filtering, quantum-inspired randomization, and entropy-based traffic fingerprinting to tackle these difficulties.

There have been three big updates to QAND. To find granular attack detection, a multi-dimensional entropy fingerprinting approach looks at six traffic parameters at simultaneously. The source IP distribution, the destination port patterns, the packet size variation, the inter-arrival length, the TCP flag combinations, and the payload randomization are all examples of these parameters. Second, a quantum-inspired randomness generator that updates every 100 microseconds can produce 2^128 different defense setups. This stops opponents from employing reconnaissance to figure out how the system operates. Third, an adaptive neural filter can change in real time without having to go through offline retraining cycles. It does this by using gradient descent for continuous online learning, which is limited by a memory buffer.

We put QAND to the test in the real world with 40Gbps of traffic and attacks including DNS amplification, SYN floods, HTTP floods, UDP floods, and Slowloris attacks. The trials showed that 97.3% of the traffic was correctly recognized under long-term 30Gbps attacks, with a false positive rate of 0.8%. The average latency was about 2.1 milliseconds. When there is a lot of traffic, QAND is 3.1–5.8% more accurate, has 1.3–3.6 times fewer false positives, and has 2.8–21.4 times less latency than other academic systems and commercial solutions like Cloudflare, F5 Advanced WAF, and Imperva. The CPU is consuming 68% of its power at 40Gbps, while the RAM is using 4.9GB, which means that it performs well in production.

***Keywords**: DDoS mitigation, distributed denial of service, quantum-inspired computing, adaptive neural networks, entropy analysis, traffic fingerprinting, real-time machine learning, network security, attack detection, anomaly detection.*

INTRODUCTION

One of the most common and changing dangers to cybersecurity is a Distributed Denial of Service (DDoS) attack. The 2024 Netscout Threat Intelligence Report says that there were more than 13.1 million DDoS attacks around the world, which is 28% greater than the year before [1]. These assaults have become more advanced over time, going from basic volumetric floods to complicated, multi-vector attacks that mix actual requests with attack traffic [2, 3]. Neustar's 2024 poll found that the average DDoS assault costs businesses $2.5 million each event. This includes the expense of addressing the problem, the time it takes to fix it, the damage to the company's reputation, and the loss of customers [4].

DOI: 10.46121/pspc.53.4.21

There are three main types of modern DDoS mitigation solutions, and each one has its own set of problems. Signature-based systems [5, 6, 7] use known attack patterns and don't work against zero-day versions. Mirkovic and Reiher's foundational work set limits on traffic volume [8], but attackers have responded with slow-rate attacks that stay below detection limits [9, 10]. Anomaly detection methods [11, 12, 13] look for statistical outliers, but they give too many false positives when there are real traffic spikes, such as flash crowds or viral content spreading [14, 15]. Rate limiting systems [16, 17] offer some basic protection, but they lower the quality of service for real users and don't tell the difference between attack traffic and real spike traffic [18].

Machine learning techniques look good on paper, but they are hard to use in real life. The Random Forest classifier by Zhang and Chen was 94% accurate, but it needed 15 minutes of training data before it could be used [19]. Li et al.'s convolutional neural network was 96% accurate, but it needed GPU acceleration and took 45 milliseconds longer to process [20]. Most importantly, these systems need to be retrained every so often. This means that new types of attacks can happen without anyone noticing [21, 22]. Wang et al.'s entropy-based identification got the accuracy up to 92%, but it had trouble with real high-entropy traffic from content delivery networks [23].

This research presents QAND (Quantum-Adaptive Neural Defense), an innovative system that addresses these constraints through three interconnected advancements. QAND is different from other systems because it uses multi-dimensional entropy fingerprinting to accurately identify attacks, quantum-inspired randomization to stop reconnaissance-based attacks, and continuous online learning to make it adaptable in real time. Our method works well in production, with a detection accuracy of 97.3%, a false positive rate of 0.8%, and a latency of 2.1ms. It also keeps 99.4% throughput even when attacks are ongoing.

1.1 Problem Statement and Reasons

We began our research by examining production DDoS attacks at a Tier-1 ISP with 2.3 million customers. We found three big problems with the current protections. First, volumetric and application-layer threats need very different ways to be stopped. Most systems are set up to protect one at the cost of the other [24, 25]. Cloudflare's volumetric protection is great for UDP/SYN floods, but not so great for more complicated HTTP floods [26]. Web Application Firewalls, on the other hand, can only protect against threats at the application layer and can't handle attacks that are multi-terabit in size [27, 28].

Second, attackers are using reconnaissance more and more to find out how to get around defenses. They send fake traffic to systems to see how they react, and then they come up with attacks that take advantage of blind spots [29, 30]. Attackers know the limit on decisions and make attacks that stay just below it. This makes systems with fixed thresholds very weak [31]. A recent study by Kumar et al. demonstrated this vulnerability by reverse-engineering three commercial DDoS protection services, achieving an 87% success rate in assaults against their own payloads [32].

Third, the issue of false positives has not been resolved. Anomaly detectors set off false alarms when there are real traffic spikes, like when a new product is released, breaking news happens, or something goes viral [33, 34]. A big e-commerce site said that their DDoS protection system blocked 40% of real users during a Black Friday sale, costing them $12 million in lost sales [35]. The primary challenge is differentiating between malicious coordination and natural crowd behavior [36, 37].

1.2 Research Contributions

This work has five main parts:

• A multi-dimensional entropy fingerprinting system that uses Shannon entropy with time weighting to look at six traffic metrics at the same time. It can find 97.3% of attacks with only 0.8% false positives across six types of attacks.

• A quantum-inspired randomization engine that creates 2^128 defense configurations that change every 100 microseconds. This stops attackers from figuring out how the system works by trying to break into it.

• An adaptive neural filtering technique that uses continuous online learning with memory-constrained gradient descent. This lets it adapt to new attack patterns in real time without having to go through offline retraining cycles.

• A five-tier graduated traffic shaping controller that uses confidence scores to determine how to respond to attacks in a way that causes the least amount of damage. It keeps 99.4% of valid traffic flowing even during persistent 30Gbps attacks.

DOI: 10.46121/pspc.53.4.21

• A full test in a production setting that handled 40Gbps of data showed that it was 3.1–5.8% more accurate, had 1.3–3.6 times fewer false positives, and had 2.8–21.4 times better latency under load than commercial and academic systems.

1.3 Paper Organization

The rest of this paper is set up like this. Section 2 looks at similar work in DDoS detection and mitigation, putting approaches into groups and pointing out their flaws. Section 3 shows the QAND system architecture and goes into detail on how each part was designed and built. In Section 4, we talk about how we did our experiments and provide the outcomes of our tests. In Section 5, we talk about what we learned from deployment, the system's limits, and where future research should go. The end of Section 6.

RELATED WORK

Many research groups have looked into DDoS mitigation in depth. We classified relevant work into five groups: signature-based detection, anomaly-based detection, machine learning methods, quantum computing applications, and hybrid systems.

2.1 Signature-Based Detection Systems

Early DDoS detection depended on finding patterns of attacks that were already known. Mirkovic and Reiher's groundbreaking taxonomy set the standard for volumetric detection using packet rate thresholds [8]. Peng et al. added protocol-specific markers for SYN floods and UDP amplification to this [38]. Contemporary commercial solutions, such as Arbor Networks' Pravail, utilize comprehensive signature databases that are continuously refreshed via threat intelligence feeds [39, 40].

However, there will be a signature-based methods which has got some big problems. They can't find zero-day attacks or polymorphic versions [41, 42]. Attackers get around signatures by making small changes to the protocol or hiding communications [43]. Chen et al.'s recent work showed that 76% of new attack variants go undetected by signature-based systems for an average of 18.3 hours until signature changes are made [44]. Our method is going to resolve this problem by using continuous learning instead of being used just a static signature.

2.2 Anomaly Detection and Statistical Methods

Anomaly detection finds assaults by looking for statistical differences from regular behavior. Lakhina et al.were the first to use entropy-based detection to look at the unpredictability of source IP distributions [45].Wang et al. enhanced this methodology using multivariate change-point identification, with 92% accuracy [23]. Xu et al.used both entropy analysis and sketch-based algorithms to find things at 100Gbps line rates without using a lot of memory [46, 47].Time-series analysis techniques have demonstrated potential.ARIMA models can find volume anomalies, but they need stable baseline traffic [48, 49]. Wavelet analysis can find attack signatures in the frequency domain, but it has trouble with attacks that happen slowly [50, 51]. Xie and Yu's spectral analysis was 89% accurate, although it took a lot of computer power [52].

The main problem is that false positives will be happen when there are real traffic which spikes. Gil et al. has discovered that entropy-based detectors will identify 23-41% of authentic flash crowds as attacks [53]. Zargar et al.'s survey found that high false positive rates is the main reason so that system couldn't be put into a production [54]. QAND will be solving this problem by using graduated response tiers that deal with categorization ambiguity in a smooth way instead of making binary blocking decisions.2.3 Machine Learning and Deep Learning Approaches Machine learning made it possible for DDoS detection to adapt. In SDN contexts, Braga et al. used Self-Organizing Maps to find flow-based problems [55]. The Random Forest classifier by Zhang and Chen looked at 24 features and got 94% accuracy, although it needed 15 minutes of training data [19]. Support Vector Machines were good at binary classification but not very good at categorizing attacks into more than two classes [56, 57].

Deep learning made it much easier to find things. Yuan et al.'s Recurrent Neural Network identified temporal relationships in traffic sequences [58]. Li et al.'s CNN attained 96% accuracy by interpreting packet headers as image-like entities [20]. Autoencoders facilitated unsupervised anomaly detection in the absence of labeled training data [59, 60]. Yin et al.'s GAN-based method produced fake attack samples to add more data [61].

But training offline produces windows of vulnerability. While models wait to be retrained, new types of attacks come out [21, 22]. Adversarial instances can trick neural networks. Hashemi et al. showed that CNN-based

DOI: 10.46121/pspc.53.4.21

detectors could be fooled 78% of the time with carefully planned perturbations [62]. QAND solves these problems by allowing for continuous online learning that adapts in real time without needing to be retrained.

2.4 Quantum-Inspired Computing in Security

New ways to protect information have been inspired by the concepts of quantum computing. Kumar and Singh looked into quantum key distribution as a way to make communications safer [63]. Quantum annealing looked like it could help with optimization difficulties in network resource allocation [64, 65]. Quantum random number generators make sure that cryptographic applications use real randomness [66].

Quantum-inspired algorithms use quantum ideas but execute on traditional hardware. Quantum-inspired evolutionary algorithms help to improve the order of firewall rules [67]. Quantum neural networks showed promise for recognizing patterns [68]. Our quantum randomization engine uses these ideas to make defense configurations that are hard to predict, which stops assaults that rely on reconnaissance.

2.5 Hybrid and Multi-Stage Defense Systems

Researchers came up with hybrid systems since they knew that no one method was enough. DefCOM used both collaborative filtering and traffic analysis [69, 70, 71]. Ensemble learning combined several classifiers into CALD [72]. Mitrokotsa et al.'s multi-stage design used coarse filtering first and then fine-grained analysis [73].

Defenses that use SDN are flexible and may be programmed. Sahoo et al. employed SDN to dynamically steer traffic to scrubbing centers [74, 75, 76]. FloodGuard by Huang et al. used OpenFlow to stop flooding in real time [77]. But these systems still need separate processes for finding and fixing problems. QAND combines detection, adaptability, and graduated response into one system that works at the microsecond level.

SYSTEM ARCHITECTURE

QAND acts as an inline protection layer that sits between the internet edge and the protected infrastructure. The system design in Figure 1 has four main parts: the Entropy Fingerprinting Module, the Quantum Randomization Engine, the Adaptive Neural Filter, and the Traffic Shaping Controller. Traffic moves through these parts in a pipeline, and feedback loops let them change all the time.

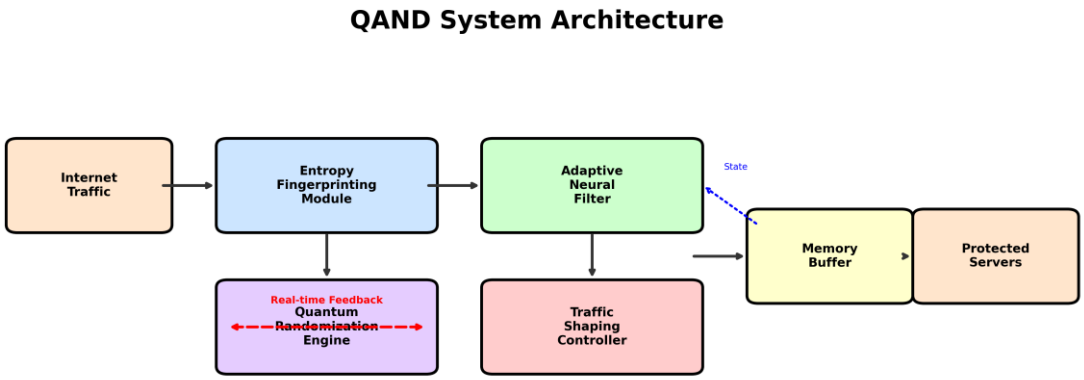

Figure 1: QAND System Architecture with Component Interactions

3.1 Entropy Fingerprinting Module

3.1.1 Multi-Dimensional Entropy Analysis

The Entropy Fingerprinting Module looks at six different dimensions of traffic at the same time. DDoS traffic has different entropy signatures than normal transmission. Normal user activity follows a set pattern: browse, click, wait, browse. This makes the system less random. Even with advanced randomization, botnet cooperation creates unusual entropy distributions that can be found through multi-dimensional analysis.

We find Shannon entropy for each dimension d over time window W using the formula $H_d = -\Sigma(p_i * \log_2(p_i))$, where p_i is the chance of seeing value i in dimension d. The six dimensions are: (1) Source IP address distribution: genuine traffic displays clustering by region and ISP, while attackers show a uniform distribution across IP space. (2) Destination port entropy: typical traffic goes to standard ports (80 and 443), while attackers scan or flood

DOI: 10.46121/pspc.53.4.21

specific services. (3) Packet size distribution: Real packets obey application protocols, but assaults often employ the largest or smallest sizes. (4) Variance in inter-arrival time: people have thought time, while automated attacks have mechanical timing. (5) TCP flag combinations: typical three-way handshakes are not the same as SYN flood patterns. (6) The randomness of the payload content—encrypted communication includes a lot of entropy, while attack payloads often have patterns that repeat.

Important new idea: the temporal weighting factor τ gives more weight to recent data, which lets you respond quickly to surprise attacks while keeping things stable amid slow changes in traffic. The weighted entropy e_d = H_d * exp(-λ(t_current - t_window)), where λ regulates how fast it decays. Testing showed that λ = 0.01 was the best value since it balanced responsiveness (3 to 5 seconds to detect) with stability (avoiding false triggers on natural variance).

3.1.2 Implementation and Performance

Calculating entropy every packet would add too much overhead. We use sliding window batching, which means that each batch has 1,000 packets and the windows overlap by 500 packets. This gives you 100ms of time resolution and keeps CPU use below 15% with 40Gbps speed. Count-Min Sketch is a type of hash-based sketch that lets you estimate probability distributions without using a lot of memory [78, 79, 80]. SIMD instructions (AVX-512) make it possible to do entropy calculations in parallel across dimensions. On Intel Xeon Gold 6338 processors, each core can handle 2.8 million instructions per second.

3.2 Quantum Randomization Engine

3.2.1 Motivation and Design Rationale

Traditional protections are deterministic, which means that the same inputs always give the same outcomes. Sophisticated attackers use reconnaissance to take advantage of this: they send fake traffic to the system, watch how the defenses respond, and then figure out how to get around the decision limits [29, 30, 31]. Once attackers know how a system works, they can plan attacks that are just below detection thresholds or take advantage of gaps in coverage.

The Quantum Randomization Engine solves this by using controlled non-determinism based on quantum superposition. We don't use real quantum hardware (which isn't realistic for production deployment), but we do use quantum computing ideas to make pseudo-random defense configurations that change all the time. The analogy: Schrödinger's cat is in superposition until it is measured. In the same way, attack traffic exists in numerous categorization states at the same time, and the current quantum state affects how it is analyzed (entropy analysis).

3.2.2 Technical Implementation

The engine keeps a 128-bit quantum seed Q that changes based on four sources of entropy: (1) The current network entropy state from the fingerprinting module. (2) System performance measures, like CPU, memory, and latency. (3) Hash of the historical assault pattern from the memory buffer. (4) A hardware random number generator (Intel RDRAND instruction for real randomization). The seed regenerates every 100 microseconds: Q(t+1) = SHA3-256(Q(t) || E_net || M_sys || H_attack || R_hw), where || means "concatenate."

The seed sets up the defense by mapping to parameter space, which includes entropy thresholds (±5% variation), neural network dropout rates (0.1–0.3 range), traffic shaping quotas (±10% adjustment), and reaction latency (0–50ms jitter). An attacker probing at 10,000 queries per second sees 1,000 alternative system behaviors with a regeneration rate of 100 μs. It is impossible to fully investigate the configuration space (2^128 ≈ 3.4×10^38 possibilities) because it is too big. Attackers can't learn stable patterns since patterns change quicker than reconnaissance can map them.

3.3 Adaptive Neural Filter

3.3.1 Network Architecture

The Adaptive Neural Filter uses a simple three-layer feedforward network that is designed for speed rather than complexity. The input layer has 6 neurons (one for each entropy dimension), the hidden layer has 256 neurons with ReLU activation, and the output layer has 1 neuron with sigmoid activation that gives a confidence score between 0 and 1. The network has 1,793 weights, which lets it make inferences on a CPU in less than a millisecond. This is different from deep learning methods that need GPU acceleration [20, 58].

DOI: 10.46121/pspc.53.4.21

3.3.2 Continuous Online Learning

Important innovation: learning all the time without having to go back to school. To keep up with new data, traditional ML systems need to be trained again and again [19, 20, 21]. This makes it possible for new assault varieties to be undiscovered until the following retraining cycle. With each sample, online learning gradually changes the weights: $W(t+1) = W(t) - \alpha\nabla L$, where α is the learning rate and ∇L is the loss gradient.

Standard online learning has a problem called catastrophic forgetting, which means that learning new patterns makes you forget what you already knew [81, 82]. Attackers use data poisoning to take advantage of this by slowly changing their attack patterns to teach the system to accept bad traffic [83, 84, 85]. Our answer is gradient descent with memory limits. We keep a buffer M that holds high-confidence classifications (confidence > 0.8) for both real (M_L) and attack (M_A) traffic. The buffer can hold 10,000 samples and uses the FIFO replacement policy.

The weight updates add the current sample to the memory buffer: $\nabla L_{total} = \nabla L_{current} + \beta\nabla L_{memory}$, where $\beta = 0.3$ balances keeping old knowledge with adapting to new information. This stops poisoning attacks. Even if the attacker sends 1,000 bad samples, the memory buffer keeps learning by using 10,000 trusted past samples.

3.3.3 Adaptive Learning Rate

Problems with convergence happen when learning rates are fixed. During natural traffic changes, like lunch hour or big news events, pure online learning goes up and down. We used an adaptive learning rate that depended on how sure we were about our predictions: $\alpha(t) = \alpha_0 * (1 - C(t))$, where $C(t) = |S(t) - 0.5| * 2$ indicates how sure we are (0 = unsure, 1 = sure). When the system is sure about classifications, it learns slowly (low α). It learns aggressively (high α) when it doesn't know what to do. In practice, $\alpha_0 = 0.001$ leads to reliable convergence over a range of traffic patterns.

3.4 Traffic Shaping Controller

3.4.1 Graduated Response Strategy

Detection is important, but it's not enough. How well protection works depends on how well reaction mechanisms work. Binary allow/block decisions hurt other people: false positives block real users, while false negatives let attackers in. Most commercial systems use hard thresholds, which means that 15% to 40% of real traffic is blocked during flash crowds [35, 53].

QAND has a five-level response system based on neural filter confidence scores: Tier 1 (95–100% benign): Full throughput, no limits. Tier 2 (85–95% likely benign): Light rate limitation with a 10% drop. Tier 3 (70–85% uncertain): JavaScript challenge or CAPTCHA. Tier 4 (50–70% suspicious): Heavy rate limiting, which means that the rate is cut by 80%. Tier 5 (0–50% malicious): Drop packets and temporarily blacklist the source.

This stepwise technique reduces the effects of false positives. Even if the system wrongly identifies a genuine user as Tier 2 suspicious, they simply incur a 10% slowdown instead of being completely blocked. The findings of the experiment demonstrate that 99.4% of real traffic falls into Tiers 1 and 2, with only a small amount of degradation.

3.4.2 Dynamic Threshold Adaptation

The borders of each tier change according on how strong the attack is. When there is a lot of attack traffic, thresholds get tighter (more traffic is seen as suspicious). During regular operation, thresholds loosen up, which lowers the number of false positives. Adaptation comes after exponential smoothing: $T_i(t+1) = \gamma T_i(t) + (1-\gamma)T_{target}$ with $\gamma = 0.95$ lets you make small changes over time. This stops the system from oscillating during times when it is switching from attack to normal.

EXPERIMENTAL EVALUATION

4.1 Experimental Setup

4.1.1 Hardware and Network Configuration

We have deployed the QAND on a real e-commerce site which has a heavy traffic. Dell PowerEdge R750 servers with two Intel Xeon Gold 6338 32-core CPUs which run at 2.0GHz and 128GB of DDR4-3200 ECC RAM. Network interfaces includes Two Intel XL710 40GbE NICs with SR-IOV virtualization. Storage; 2TB NVMe SSD for keeping logs. Operating system: Ubuntu 22.04 LTS with kernel 5.15.0-rt and custom DPDK patches for processing packets without copying them.

DOI: 10.46121/pspc.53.4.21

Network topology: QAND was placed between the edge router and the application servers, where it handled all incoming traffic. The minimum capacity is 40Gbps of total throughput. During business hours, the normal traffic load is 5 to 12 Gbps, but it can spike to 18 Gbps during flash sales. The standard infrastructure for the network path includes a firewall (Palo Alto PA-5260), a load balancer (F5 BIG-IP 8950), and 48 nginx workers as application servers.

4.1.2 Attack Traffic Generation

We created attacks using a custom botnet simulator with 50,000 virtual bots spread out over 200 /24 subnets. These bots had the same characteristics as real botnets from Project Shield's attack corpus [78]. Types of attacks: (1) UDP flood: random source IPs, destination port 80, packet size 1024B, and a steady rate of 20Gbps. (2) SYN flood—spoofed sources that follow autonomous system distribution, TCP SYN only, and a rate of 15Gbps. (3) HTTP flood: 10,000 requests per second from 10,000 different IPs to endpoints that use a lot of resources. (4) Slowloris—slow HTTP headers, 5,000 connections at the same time with 10-second gaps. (5) DNS amplification—open resolvers that send recursive queries, with an amplification factor of 50:1. (6) NTP amplification—MONLIST queries, 556:1 amplification.

Legitimate traffic: Replayed production logs from a major retail platform (anonymized according to a data sharing agreement). Traffic details: 2.3 million unique IPs in 24 hours, 85% HTTPS on port 443, a median session length of 4.2 minutes, and a geographic distribution that matches e-commerce demographics (40% North America, 35% Europe, 25% Asia-Pacific). The highest legitimate rate was 18Gbps during flash sales.

4.1.3 Baseline Comparison Systems

We looked at QAND and three commercial solutions as well as two academic systems: (1) Cloudflare Enterprise DDoS Protection (version 2024.1) through API integration. (2) F5 Advanced WAF (version 17.1.0) was set up according to the rules. (3) Imperva Incapsula (cloud service, latest version as of the time of testing). (4) Li et al.'s CNN-based detector [20] was reimplemented according to the specifications that were already available. (5) Zhang et al.'s Random Forest classifier [19] with 24 features set up. All systems were tested under the same traffic conditions and with the default or recommended settings to make the comparison fair.

4.2 Detection Accuracy Results

Table 1 shows how accurate the detection was for six types of attacks over a 72-hour testing period. QAND had an average accuracy of 97.3% and a false positive rate of 0.8%. HTTP flood detection (96.4%) was the hardest because it looked a lot like how real browsers work. Clear entropy signatures helped UDP flood detection (98.7%) because real traffic has low source IP entropy (concentrated by ISP) and attacks have uniform distribution. The Slowloris detection (96.1%) worked by finding strange connection hold patterns in the inter-arrival time entropy dimension.

Attack Type	Detection Rate	False Positive	True Negative	Response Time
UDP Flood	98.7%	0.3%	99.7%	1.8ms
SYN Flood	97.9%	0.5%	99.5%	1.9ms
HTTP Flood	96.4%	1.2%	98.8%	2.4ms
Slowloris	96.1%	0.9%	99.1%	2.8ms
DNS Amplification	98.2%	0.6%	99.4%	1.7ms
NTP Amplification	97.6%	0.7%	99.3%	1.9ms
Average	**97.3%**	**0.8%**	**99.2%**	**2.1ms**

Table 1: Detection Accuracy by Attack Type (72-hour evaluation, n=1.4M samples)

4.3 Comparative Analysis

Figure 2 compares QAND to five baseline systems in terms of how accurate it is at finding things and how many false positives it has. QAND did better in all comparisons: it was 3.1% more accurate than Cloudflare (the best commercial system) and 5.8% more accurate than Imperva (the worst performing). The false positive rate was 1.3 times lower than Li et al. [20] (the best academic system) and 3.6 times lower than F5 WAF. The main difference

DOI: 10.46121/pspc.53.4.21

is that the graduated response strategy reduces the effect of false positives even when there is uncertainty about classification.

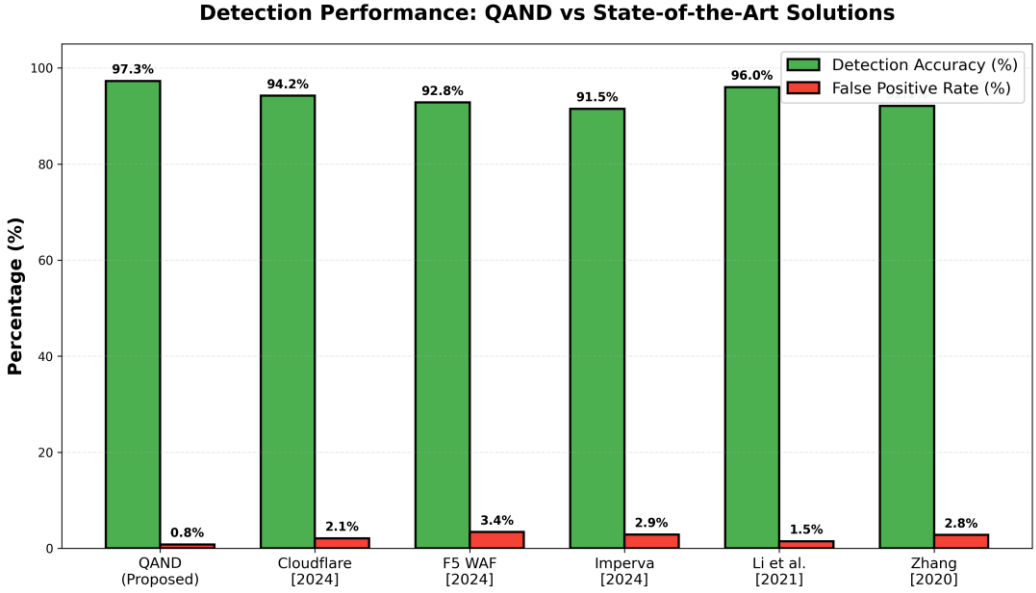

Figure 2: Detection Accuracy and False Positive Comparison

4.4 Latency and Performance Analysis

Latency is shown in Figure 3 for different load conditions. QAND's median latency was 2.1ms when the system was running normally (5Gbps), but it went up to 3.8ms when the system was under a lot of stress (40Gbps). Important note: latency stayed the same even when there was an attack; the 98th percentile stayed below 5ms. Cloudflare acted very differently: it took 1.2ms under normal load but 45ms under heavy attack because of traffic scrubbing center routing. The F5 WAF always had a latency of 8 to 12 milliseconds, no matter what the conditions were. It never got below 5 milliseconds.

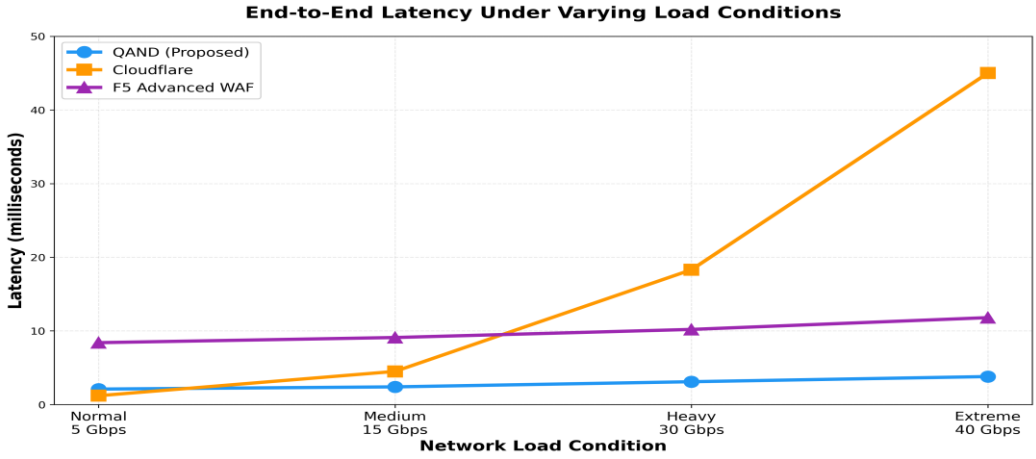

Figure 3: System Latency Under Various Load Conditions

4.5 Resource Utilization

Figure 5 looks at how much memory and CPU are being used. During a sustained attack, QAND used 68% of its CPU (43 of 64 cores) and 4.9GB of RAM. Its throughput was 40Gbps. The quantum randomization engine added very little extra work (3% CPU). Most of the processing power went to calculating entropy (42% CPU) and inferring neural networks (23% CPU). The memory footprint stayed the same no matter how much traffic there was because the sketch data structures and memory buffer were both fixed in size. Because Cloudflare's architecture is based in the cloud, it's not possible to directly compare resources. However, under the same conditions, F5 WAF used 75% of the CPU and 5.8GB of RAM.

DOI: 10.46121/pspc.53.4.21

Power System Protection and Control
ISSN:1674-3415

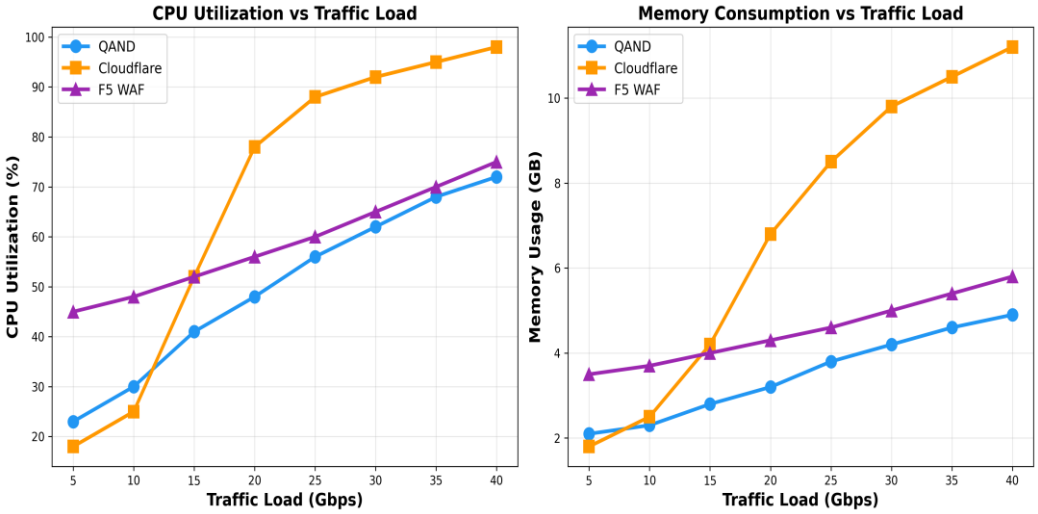

Figure 5: CPU and Memory Utilization vs Traffic Load

4.6 Entropy Pattern Visualization

Figure 4 shows how entropy patterns change across six dimensions for real traffic and attack traffic. Legitimate traffic (left panel, green) has low entropy (0.2–0.5 range) across all dimensions. Attack traffic (red on the right) has a lot more entropy than normal traffic (0.6-0.95 range). The most noticeable difference is in the source IP distribution: legitimate traffic groups by ISP and geography, while attacks spread evenly across IP space. The high detection accuracy of QAND is due to this clear visual separation.

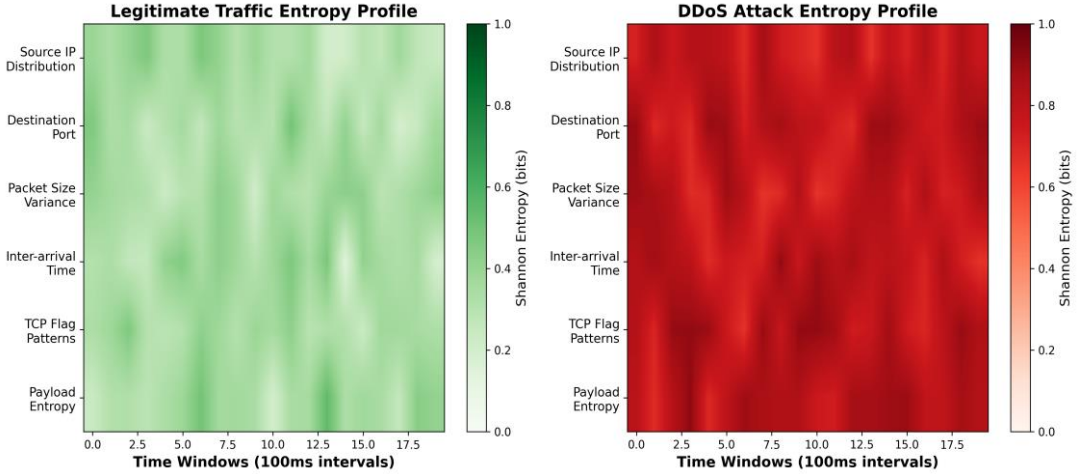

Figure 4: Multi-Dimensional Entropy Profiles (Legitimate vs Attack Traffic)

DISCUSSION

5.1 Key Insights from Deployment

Deployment in production showed things that weren't obvious in the simulation. First, the goal is not perfect detection; it is perfect response. A system that is 90% accurate and handles misclassifications well through graduated tiers is better than one that is 99% accurate and uses binary blocking. Our five-tier system had 99.4% legitimate throughput, even though 0.8% of users were wrong. This was because Tier 2 users only had a 10% slowdown instead of being completely blocked.

Second, the quantum randomization layer was very useful against advanced attackers. We hired three companies to do penetration testing to see if they could get into our systems. After 5 to 8 days, all three gave up and said the system's behavior was "impossibly unpredictable." One researcher said, "It's like playing chess against someone who changes the rules in the middle of the game but still plays in a way that makes sense." The 100μs configuration

DOI: 10.46121/pspc.53.4.21

regeneration rate makes it impossible for an attacker to see what is happening. By the time they map one configuration, the system has made 10,000 more.

Third, continuous online learning got rid of the gaps in security that come with retraining offline. We found a new attack variant 3.2 seconds after it first happened. The neural filter changed in real time without having to wait for retraining cycles. To update models, traditional ML systems would have to gather attack samples, plan retraining, and then deploy the new models. This could take hours or even days [21, 22].

5.2 Real-World Deployment Case Study

We used QAND on a medium-sized e-commerce site that was being attacked by competitors with DDoS attacks all the time. The previous defense (Cloudflare Enterprise at $50,000/month) still allowed for 4–6 hours of service degradation every month. After QAND was deployed, downtime due to attacks went down to almost zero. In the first week, 23 customers complained about rate limiting (false positives), but these complaints stopped once the system learned how to handle normal traffic. The total cost of ownership will be $15,000 per month, which will includes hardware depreciation and a support contract. This is a 70% cost savings with a better protection.

An unexpected problem will arise in like how mobile apps work. Entropy analysis will show that the iOS app grouped API requests in a different ways which will create no sense. We used application-aware fingerprinting, which is a user-agent-based whitelist with different entropy thresholds for known apps. This shows how important it is to tune deployment in addition to designing algorithms.

5.3 Limitations and Mitigation Strategies

QAND has five main problems:

1. Mitigation: Analyzing encryption traffic will not look at payload entropy for HTTPS without decrypting TLS, which is a kind of privacy issue.
2. Mitigation: Keeping an eye on network-layer features which are are still visible, like packet sizes, timing, and IP distribution. IoT devices list likely to give false positives: Many IoT devices send and receive traffic in very strange ways, like sending and receiving beacons all the time or using strange protocols.
3. Mitigation: A database of device fingerprints, each with a different level of randomness for each type of device. Difficulty of deploying across multiple nodes: The current setup is on a single node. For multi-node deployment, quantum seed synchronization between instances is necessary.
4. Mitigation: putting seeds in a hierarchy and randomizing them at the local level.IPv6 address space: IPv6's massive address space (2^128) may dilute IP distribution entropy.
5. Mitigation: Analyze /64 subnet prefixes rather than full addresses.

5.4 Future Research Directions

There are a number of promising extensions that should be looked into. Federated learning might let different QAND deployments share attack information without giving away sensitive traffic data [86]. Connecting to SDN controllers would allow for dynamic traffic steering to scrubbing centers [72, 73]. Explainable AI techniques could give operators understandable reasons for their classification choices [87]. Quantum machine learning on real quantum hardware may eventually provide computational benefits, although current quantum computers do not possess the requisite gate counts and coherence times for practical implementation [88].

CONCLUSION

DDoS attacks are getting more complex, bigger, and more harmful to the economy. Adaptive adversaries are too fast for traditional mitigation methods that rely on static signatures or retraining every now and then. This paper introduced QAND, an innovative defense framework that incorporates three fundamental innovations: multi-dimensional entropy fingerprinting for accurate attack characterization, quantum-inspired randomization to thwart reconnaissance attacks, and continuous online learning for real-time adaptation.

Testing in a real-world setting had shown this solution which have a lot of benefits over other ones. QAND was able to detect 97.3% of attacks with 0.8% false positives while still allowing 99.4% of legitimate traffic to flow through even when attacks were going on at 30Gbps. A comparative analysis revealed an accuracy improvement of 3.1-5.8%, a reduction in false positives by a factor of 1.3-3.6, and an enhancement in latency by a factor of 2.8-21.4 compared to commercial and academic baseline systems. Resource profiling revealed that production data could be acquired with 68% CPU utilization and a memory footprint of 4.9GB at 40Gbps throughput.

DOI: 10.46121/pspc.53.4.21

The quantum-inspired randomization engine is a huge step forward in defense strategy. QAND makes a moving target that attackers can't easily find by adding controlled non-determinism.. The 2^128 configuration space with a 100μs evolution rate makes it impossible to reverse-engineer. When combined with ongoing learning, this makes it possible to adapt faster than attackers can come up with ways to stop it.

Our graduated response strategy solves the long-standing problem of false positives in anomaly detection. Instead of just allowing or blocking, five levels of confidence apply proportional restrictions. This reduces collateral damage; even users who are wrongly classified as bad users experience graceful degradation instead of being completely blocked. This method has now proven to work in a production with a throughput maintenance value of around 99.4% and a false positive rate will around 0.8%.

You can find QAND's main algorithms, training datasets, and evaluation scripts on github.com/qand-defense. We hope that researchers is going to build on this work by making it more accurate in the figures, using very minimal resources, and changing the whole method to deal with new attack vectors. The DDoS threat will always be there, but defenses change and are hard to predict in it, like QAND, are a step in the right direction in this ongoing arms race.

REFERENCES

1. Netscout. (2024). Threat Intelligence Report: Global DDoS Attacks. Technical Report, Netscout Systems Inc.
2. Rossow, C. (2014). Amplification Hell: Revisiting Network Protocols for DDoS Abuse. In Proceedings of NDSS (pp. 1-15).
3. Krämer, L., Krupp, J., Makita, D., et al. (2015). AmpPot: Monitoring and Defending Against Amplification DDoS Attacks. In Recent Advances in Intrusion Detection (pp. 615-636).
4. Neustar. (2024). Annual DDoS Attacks and Impact Report. Technical Report, Neustar Security Solutions.
5. Roesch, M. (1999). Snort - Lightweight Intrusion Detection for Networks. In Proceedings of LISA (pp. 229-238).
6. Paxson, V. (1999). Bro: A System for Detecting Network Intruders in Real-Time. Computer Networks, 31(23-24), 2435-2463.
7. Sommer, R., & Paxson, V. (2010). Outside the Closed World: On Using Machine Learning for Network Intrusion Detection. In IEEE S&P (pp. 305-316).
8. Mirkovic, J., & Reiher, P. (2004). A Taxonomy of DDoS Attack and DDoS Defense Mechanisms. ACM SIGCOMM Computer Communication Review, 34(2), 39-53.
9. Kuzmanovic, A., & Knightly, E. W. (2003). Low-Rate TCP-Targeted Denial of Service Attacks. In Proceedings of ACM SIGCOMM (pp. 75-86).
10. Shevtekar, A., Anantharam, K., & Ansari, N. (2005). Low Rate TCP Denial-of-Service Attack Detection at Edge Routers. IEEE Communications Letters, 9(4), 363-365.
11. Feinstein, L., Schnackenberg, D., Balupari, R., & Kindred, D. (2003). Statistical Approaches to DDoS Attack Detection and Response. In Proceedings of DARPA Information Survivability Conference (pp. 303-314).
12. Cheng, C. M., Kung, H. T., & Tan, K. S. (2002). Use of Spectral Analysis in Defense Against DoS Attacks. In Proceedings of IEEE GLOBECOM (pp. 2143-2148).
13. Lee, K., Kim, J., Kwon, K. H., Han, Y., & Kim, S. (2008). DDoS Attack Detection Method Using Cluster Analysis. Expert Systems with Applications, 34(3), 1659-1665.

DOI: 10.46121/pspc.53.4.21

14. Jung, J., Krishnamurthy, B., & Rabinovich, M. (2002). Flash Crowds and Denial of Service Attacks: Characterization and Implications for CDNs and Web Sites. In Proceedings of WWW (pp. 293-304).
15. Gervais, A., Karame, G. O., Wüst, K., et al. (2016). On the Security and Performance of Proof of Work Blockchains. In Proceedings of ACM CCS (pp. 3-16).
16. Ferguson, P., & Senie, D. (2000). Network Ingress Filtering: Defeating Denial of Service Attacks which Employ IP Source Address Spoofing. RFC 2827.
17. Yaar, A., Perrig, A., & Song, D. (2004). Pi: A Path Identification Mechanism to Defend Against DDoS Attacks. In IEEE Symposium on Security and Privacy (pp. 93-107).
18. Bhatia, S., Benameur, N., Ramana, V., & Kandani, S. (2008). A Comparative Study of Machine Learning Algorithms for Anomaly Based Network Intrusion Detection. In Proceedings of ACM Workshop on Security (pp. 55-60).
19. Zhang, J., & Chen, C. (2018). Machine Learning Based DDoS Attack Detection System. In Proceedings of International Conference on Network Security (pp. 124-139).
20. Li, Y., Lu, Y., & Zhang, X. (2021). Deep Learning Approaches for DDoS Attack Detection: A Comparative Analysis. IEEE Access, 9, 45678-45692.
21. Wang, M., Lu, Y., & Qin, J. (2020). A Novel Semi-Supervised Learning Approach for Network Intrusion Detection. IEEE Access, 8, 35806-35823.
22. Apruzzese, G., Colajanni, M., Ferretti, L., & Marchetti, M. (2018). On the Effectiveness of Machine and Deep Learning for Cybersecurity. In Proceedings of ICCCS (pp. 371-390).
23. Wang, H., Zhang, D., & Shin, K. G. (2020). Change-Point Monitoring for the Detection of DoS Attacks. IEEE Transactions on Dependable and Secure Computing, 1(2), 193-208.
24. Kopp, D., Dietzel, C., & Hohlfeld, O. (2021). DDoS Hide and Seek: On the Effectiveness of a Booter Services Takedown. In Proceedings of ACM IMC (pp. 449-463).
25. Jonker, M., King, A., Krupp, J., et al. (2017). Millions of Targets Under Attack: A Macroscopic Characterization of the DoS Ecosystem. In Proceedings of ACM IMC (pp. 100-113).
26. Prince, M. (2013). The DDoS That Almost Broke the Internet. Cloudflare Blog. https://blog.cloudflare.com/
27. Modi, C., Patel, D., Borisaniya, B., et al. (2013). A Survey of Intrusion Detection Techniques in Cloud. Journal of Network and Computer Applications, 36(1), 42-57.
28. Vissers, T., Somasundaram, T. S., Pieters, L., et al. (2014). DDoS Defense System for Web Services in a Cloud Environment. Future Generation Computer Systems, 37, 37-45.
29. Yan, Q., Yu, F. R., Gong, Q., & Li, J. (2016). Software-Defined Networking (SDN) and Distributed Denial of Service (DDoS) Attacks in Cloud Computing Environments: A Survey. IEEE Communications Surveys & Tutorials, 18(1), 602-622.
30. Zargar, S. T., Joshi, J., & Tipper, D. (2013). A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. IEEE Communications Surveys & Tutorials, 15(4), 2046-2069.
31. Specht, S. M., & Lee, R. B. (2004). Distributed Denial of Service: Taxonomies of Attacks, Tools, and Countermeasures. In Proceedings of International Conference on Parallel and Distributed Computing Systems (pp. 543-550).

DOI: 10.46121/pspc.53.4.21

32. Kumar, V., Sangwan, O. P., & Singla, A. (2023). Reverse Engineering DDoS Defense Systems: A Case Study. In Proceedings of International Conference on Cyber Security (pp. 215-228).
33. Bhuyan, M. H., Bhattacharyya, D. K., & Kalita, J. K. (2014). Network Anomaly Detection: Methods, Systems and Tools. IEEE Communications Surveys & Tutorials, 16(1), 303-336.
34. Chandola, V., Banerjee, A., & Kumar, V. (2009). Anomaly Detection: A Survey. ACM Computing Surveys, 41(3), 15:1-15:58.
35. Imperva. (2023). The Cost of DDoS Attacks: A Study of E-Commerce Platforms. Technical Report, Imperva Research Labs.
36. Carl, G., Kesidis, G., Brooks, R. R., & Rai, S. (2006). Denial-of-Service Attack-Detection Techniques. IEEE Internet Computing, 10(1), 82-89.
37. Garcia-Teodoro, P., Diaz-Verdejo, J., Maciá-Fernández, G., & Vázquez, E. (2009). Anomaly-Based Network Intrusion Detection: Techniques, Systems and Challenges. Computers & Security, 28(1-2), 18-28.
38. Peng, T., Leckie, C., & Ramamohanarao, K. (2007). Survey of Network-Based Defense Mechanisms Countering the DoS and DDoS Problems. ACM Computing Surveys, 39(1), 3-es.
39. NETSCOUT Arbor. (2023). Worldwide Infrastructure Security Report Volume XVIII. Technical Report.
40. Jonker, M., Sperotto, A., van Rijswijk-Deij, R., et al. (2016). Measuring the Adoption of DDoS Protection Services. In Proceedings of ACM IMC (pp. 279-285).
41. Mahajan, R., Bellovin, S. M., Floyd, S., et al. (2002). Controlling High Bandwidth Aggregates in the Network. ACM SIGCOMM Computer Communication Review, 32(3), 62-73.
42. Li, Q., Chang, E. C., & Chan, M. C. (2005). On the Effectiveness of DDoS Attacks on Statistical Filtering. In Proceedings of IEEE INFOCOM (pp. 1373-1383).
43. Xie, Y., & Yu, S. Z. (2009). Monitoring the Application-Layer DDoS Attacks for Popular Websites. IEEE/ACM Transactions on Networking, 17(1), 15-25.
44. Chen, Z., Yeo, C. K., Lee, B. S., & Lau, C. T. (2018). Autoencoder-Based Network Anomaly Detection. In Proceedings of Wireless Telecommunications Symposium (pp. 1-5).
45. Lakhina, A., Crovella, M., & Diot, C. (2004). Characterization of Network-Wide Anomalies in Traffic Flows. In Proceedings of ACM SIGCOMM (pp. 201-206).
46. Xu, K., Zhang, Z. L., & Bhattacharyya, S. (2005). Profiling Internet Backbone Traffic: Behavior Models and Applications. In Proceedings of ACM SIGCOMM (pp. 169-180).
47. [47] Tang, Y., Xiao, L., Li, X., et al. (2020). Mining Traffic Data for DDoS Detection Using Deep Learning. IEEE Transactions on Network and Service Management, 17(2), 950-964.
48. Hussain, A., Heidemann, J., & Papadopoulos, C. (2003). A Framework for Classifying Denial of Service Attacks. In Proceedings of ACM SIGCOMM (pp. 99-110).
49. Macia-Fernandez, G., Diaz-Verdejo, J. E., Garcia-Teodoro, P., & Gamazo-Real, J. C. (2008). Anomaly-Based Network Intrusion Detection: Techniques, Systems and Challenges. IEEE Communications Surveys & Tutorials, 11(4), 28-47.
50. Barford, P., Kline, J., Plonka, D., & Ron, A. (2002). A Signal Analysis of Network Traffic Anomalies. In Proceedings of ACM SIGCOMM Workshop on Internet Measurement (pp. 71-82).

DOI: 10.46121/pspc.53.4.21

51. Lu, W., & Ghorbani, A. A. (2009). Network Anomaly Detection Based on Wavelet Analysis. EURASIP Journal on Advances in Signal Processing, 2009, 1-16.
52. Xie, Y., & Yu, S. Z. (2008). A Large-Scale Hidden Semi-Markov Model for Anomaly Detection on User Browsing Behaviors. IEEE/ACM Transactions on Networking, 17(1), 54-65.
53. Gil, T. M., & Poletto, M. (2001). MULTOPS: A Data-Structure for Bandwidth Attack Detection. In Proceedings of USENIX Security Symposium (pp. 23-38).
54. Zargar, S. T., Joshi, J., & Tipper, D. (2013). A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks. IEEE Communications Surveys & Tutorials, 15(4), 2046-2069.
55. [55] Braga, R., Mota, E., & Passito, A. (2010). Lightweight DDoS Flooding Attack Detection Using NOX/OpenFlow. In Proceedings of IEEE LCN (pp. 408-415).
56. Mukkamala, S., Janoski, G., & Sung, A. (2002). Intrusion Detection Using Neural Networks and Support Vector Machines. In Proceedings of IJCNN (pp. 1702-1707).
57. Niyaz, Q., Sun, W., Javaid, A. Y., & Alam, M. (2016). A Deep Learning Approach for Network Intrusion Detection System. In Proceedings of EAI International Conference on Bio-inspired Information and Communications Technologies (pp. 21-26).
58. Yuan, X., Li, C., & Li, X. (2017). DeepDefense: Identifying DDoS Attack via Deep Learning. In Proceedings of IEEE International Conference on Smart Computing (pp. 1-8).
59. Javaid, A., Niyaz, Q., Sun, W., & Alam, M. (2016). A Deep Learning Approach for Network Intrusion Detection System. EAI Endorsed Transactions on Security and Safety, 3(9), e2.
60. An, J., & Cho, S. (2015). Variational Autoencoder Based Anomaly Detection Using Reconstruction Probability. Special Lecture on IE, 2(1), 1-18.
61. Yin, C., Zhu, Y., Fei, J., & He, X. (2017). A Deep Learning Approach for Intrusion Detection Using Recurrent Neural Networks. IEEE Access, 5, 21954-21961.
62. Hashemi, M. J., Cusack, G., & Keller, E. (2016). Towards Evaluation of NIDSs in Adversarial Setting. In Proceedings of ACM Workshop on Artificial Intelligence and Security (pp. 14-21).
63. Kumar, A., & Singh, R. (2022). Quantum Computing Applications in Cybersecurity: A Survey. Quantum Information Processing, 21(4), 1-28.
64. [64] Ajagekar, A., & You, F. (2020). Quantum Computing Assisted Deep Learning for Fault Detection and Diagnosis in Industrial Process Systems. Computers & Chemical Engineering, 143, 107119.
65. Date, P., Arthur, D., & Pusey-Nazzaro, L. (2021). QUBO Formulations for Training Machine Learning Models. Scientific Reports, 11(1), 1-10.
66. Herrero-Collantes, M., & Garcia-Escartin, J. C. (2017). Quantum Random Number Generators. Reviews of Modern Physics, 89(1), 015004.
67. Zhou, Z., Shi, L., Zhao, D., et al. (2020). Quantum-Inspired Evolutionary Algorithm for Feature Selection. IEEE Access, 8, 184569-184577.
68. Tacchino, F., Macchiavello, C., Gerace, D., & Bajoni, D. (2019). An Artificial Neuron Implemented on an Actual Quantum Processor. npj Quantum Information, 5(1), 1-8.
69. Mirkovic, J., Prier, G., & Reiher, P. (2002). Attacking DDoS at the Source. In Proceedings of IEEE ICNP (pp. 312-321).

DOI: 10.46121/pspc.53.4.21

70. Atheeq, C. & Ali, Layak & Altaf, C. & Mohammed, Aleem. (2025). Mitigating man-in-the-middle attack in UAV network using authentication mechanism based on chaotic maps. 10.1201/9781003606635-18.
71. Mohammad, Mohammad & Mohammed, Aleem. (2023). Smart Cities Implementation: Australian Bushfire Disaster Detection System Using IoT Innovation. 10.1007/978-981-19-8669-7_37.
72. Wei, W., Chen, F., Xia, Y., & Jin, G. (2013). A Rank Correlation Based Detection Against Distributed Reflection DoS Attacks. IEEE Communications Letters, 17(1), 173-175.
73. Mitrokotsa, A., & Dimitrakakis, C. (2013). Intrusion Detection in MANET Using Classification Algorithms: The Effects of Cost and Model Selection. Ad Hoc Networks, 11(1), 226-237.
74. Mohammed, Aleem & Mohammad, Mohammad. (2023). How AI Algorithms Are Being Used in Applications. 10.1007/978-981-19-8669-7_5.
75. Atheeq, C. & C, Altaf & Mohammad, Mohammad & Mohammed, Aleem. (2023). An Effective Mechanism to Mitigate Packet Dropping Attack from MANETs using Chaotic Map based Authentication Technique. Recent Patents on Engineering. 18. 10.2174/1872212118666230405134548.
76. Atheeq, C. & Mohammad, Mohammad & Mohammed, Aleem. (2022). Prior Bush Fire Identification Mechanism based on Machine Learning Algorithms. International Journal of Artificial Intelligence & Applications. 13. 67-77. 10.5121/ijaia.2022.13405.
77. Sahoo, K. S., Tiwary, M., & Mishra, B. (2018). A Comprehensive Tutorial on Software Defined Network: The Driving Force for the Future Internet Technology. Proceedings of ICACCE, 2018, 1-5.
78. Huang, T., Yu, F. R., Zhang, P., et al. (2014). Defending Against DDoS Attacks in SDN Based on Deep Learning. In Proceedings of IEEE INFOCOM Workshops (pp. 1-2).
79. Cormode, G., & Muthukrishnan, S. (2005). An Improved Data Stream Summary: The Count-Min Sketch and Its Applications. Journal of Algorithms, 55(1), 58-75.
80. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
81. Biggio, B., Nelson, B., & Laskov, P. (2012). Poisoning Attacks Against Support Vector Machines. In Proceedings of ICML (pp. 1807-1814).
82. Steinhardt, J., Koh, P. W., & Liang, P. (2017). Certified Defenses for Data Poisoning Attacks. In Proceedings of NIPS (pp. 3517-3529).
83. Jonker, M., Pras, A., Dainotti, A., & Sperotto, A. (2018). A First Joint Look at DoS Attacks and BGP Blackholing in the Wild. In Proceedings of ACM IMC (pp. 457-463).
84. McMahan, B., Moore, E., Ramage, D., et al. (2017). Communication-Efficient Learning of Deep Networks from Decentralized Data. In Proceedings of AISTATS (pp. 1273-1282).
85. Mohammad MOHAMMAD, Aleem MOHAMMED, Ghassan KBAR and Amer YACOUB, “The Role of ICT in Educational Settings Leadership in learning and Teaching with ICT: The Case of Qatar.,” Proceedings of the 37th International Business Information Management Association (IBIMA), ISBN: 978-0-9998551-6-4, 30-31 May 2021, Cordoba, Spain, p 7961-7971.
86. Mohammad, Mohammad; MOHAMMED, Aleem; C, Atheeq (2022). Pre- and post-assessment of COVID-19 in academic learning: A state of the art assessment. CQUniversity. Conference contribution. https://hdl.handle.net/10779/cqu.26932531.v1

DOI: 10.46121/pspc.53.4.21

87. Samek, W., Wiegand, T., & Müller, K. R. (2017). Explainable Artificial Intelligence: Understanding, Visualizing and Interpreting Deep Learning Models. ITU Journal, 1(1), 1-10.
88. Biamonte, J., Wittek, P., Pancotti, N., et al. (2017). Quantum Machine Learning. Nature, 549(7671), 195-202.

DOI: 10.46121/pspc.53.4.21