Data Driven Optimum Cyberattack Mitigation

Erol Gelenbe
Inst. of Theoretical & App. Informatics
Polish Acad. of Sciences (IITIS-PAN)
44100 Gliwice, PL
& CNRS I3S, Univ. Côte d'Azur, 06103 Cedex 2 Nice, FR
& Dept. of Eng., King's College London, UK
ORCID:0000-0001-9688-2201

Mohammed Nasereddin
Inst. of Theoretical & App. Informatics
Polish Acad. of Sciences (IITIS-PAN)
44100 Gliwice, PL
ORCID:0000-0002-3740-9518

***Abstract*—Gateways to the Internet of Things (IoT) are typically servers that communicate with IoT devices, providing them with low-latency services, and connecting them to the internet and other backbone networks. Since IoT devices are often simple and have limited storage and computational capabilities, gateways can be equipped with Attack Detection (AD) software to analyze incoming traffic, detect potential cyberattacks, and protect both the gateway and connected devices from threats that could overwhelm the system as a whole. This paper presents an enhanced gateway system that combines a traffic shaping technique with an attack detection module and an optimum attack mitigation scheme aimed at protecting the gateway and the overall system from cyberattacks. The optimum mitigation approach selects a sampling interval for the AD, that minimizes the total overhead of AD and mitigation. The proposed approach is implemented in a practical test-bed, so that the performance of the mitigation scheme may be evaluated in the presence of flood attacks. The experiments show its practical value and illustrate the agreement obtained between the analysis and the measurements obtained from several experiments**

***Index Terms*—IoT Gateways, Optimum Attack Detection and Mitigation, Flood Attacks, Quasi-Deterministic Transmission Policy**

I. Introduction

The Internet of Things (IoT) is composed of over 22 billion devices that are often connected to IoT gateways, which are servers designed to process the traffic to and from a set of devices, while providing them with low-latency edge services. These gateway servers can be subject to cyberattacks [1], [2] that can compromise both the gateways and IoT devices, disabling them through malware and packet floods [3], such as the 2017 attack that took down $180,000$ servers with 2.54 $Tbps$ of traffic [4]. Such attacks [5] not only compromise their targets but can also convert them into attackers [6], creating packet floods that cause congestion, system crashes [7], and even overload the Attack Detection (AD) systems that are often installed on gateways. Therefore it is crucial to protect IoT infrastructures with effective AD, but also with mitigation techniques.

Thus, in this paper we first review the literature on leading-edge AD and mitigation solutions in Section II, highlighting the limitations of existing models and explaining how the proposed approach in this paper addresses these shortcomings. Then, in Section III, we introduce a novel approach that combines traffic shaping to protect the AD from attack overloads, together with periodic AD that examines the incoming traffic, and adaptively drops packets when an attack is detected. This approach is based on an Adaptive Attack Mitigation (AAM) algorithm that minimizes a cost function combining the overhead of the AD when it examines packets, and the loss of a fraction of the benign packets due to the packet drops when an attack is detected. This approach is then evaluated with extensive measurements, showing a good agreement between the theoretical results in Section III and the experimental measurements in Section IV. Conclusions and suggestions for future work are presented in Section V.

II. Literature Review

Attack detection and mitigation systems often employ machine learning (ML) and deep learning (DL) techniques, such as Decision Trees [8], Convolutional Neural Networks (CNNs) [9], and Generative Adversarial Networks (GANs) [10], to identify anomalies and detect cyber threats, although analytical modelling techniques have also been used in this area [11]. Most evaluations are conducted offline, without considering the real-time impact of attacks and the consequence of the attack on the targeted system [12]–[14], including the impact that attacks can have on its energy consumption [15]. However, some test-beds have been constructed to conduct experimental attacks [16], [17]. Several security protocols and architectures have been developed to meet the specific

demands of gateway environments, with emphasis on autonomous vehicles [18] and 6G-enabled Internet of Vehicles (IoV) environments [19]. Recent advances also emphasize collaborative and adaptive solutions to address data-sharing limitations across heterogeneous gateways. For instance, transfer [20] and federated learning [21] enable knowledge transfer between devices and networks without compromising data privacy, with improvements in attack detection rates. Advanced AD systems can incorporate innovative techniques such as spatial-temporal analysis, Bayesian networks [22], and hybrid approaches to secure smart cities [23], [24] and critical SCADA systems [25]. Furthermore, integrating DL models with edge computing and Multi-access Edge Computing (MEC) has been useful in real-time attack mitigation [26]. Ensemble learning and proactive detection methods using Bayesian DL and Discrete Wavelet Transform, have shown the importance of adaptive techniques to counter threats targeting gateways [27].

By focusing on mitigation solutions, Table I presents a selection of recent research papers, including a comparison with the system proposed in this paper. It compares the adopted approaches and highlights the limitations and challenges associated with each solution. The proposed system integrates traffic regulation, attack detection, and a novel adaptive attack mitigation technique that dynamically drops packets during an attack to minimize computational overhead. Notably, the training phase of the system does not require high computational power, making it deployable on resource-constrained devices such as those used in IoT environments. It can be trained on a limited number of packets—just a few thousand—within a short time frame. A key advantage of this system is that it only requires training on normal traffic, eliminating the need for attack-specific samples. This feature also enhances its robustness and adaptability to diverse attack patterns and behaviors. Furthermore, the effectiveness of the employed AD has already been validated across various environments and datasets, as mentioned in Section III.

III. Methodology

In this section, we present the proposed Adaptive Attack Mitigation (AAM) system. It consists of a Smart QDTP Forwarder (SQF) which shapes the incoming traffic to protect the server from excessive packet backlog during a flood attack, followed by the AD attack detection module, which is enhanced by the novel AAM algorithm for attack mitigation. The AD takes its decision based on a window of W packets. As long as the AD does not detect an attack, the AAM is not activated. However, when the AD detects an attack, it triggers the AAM which takes the following actions: it first drops the packets at the input to the SQF, and then directs the AD to skip the following $m + W$ incoming packets and then sample the subsequent $m + 1$-th window of incoming packets repeating the sequence of drops and samples until the AD announces a "$NO - ATTACK$" decision, at which time normal operation resumes. The value m is selected in a manner that minimizes a cost function $C(AAM)$ which combines the overhead of testing incoming packet streams and dropping benign as well as malicious packets.

This AD [34] uses the Random Neural Network [35] based learning [36], [37], and uses clusters of neurons with soma-to-soma interactions [38], [39]. It was evaluated with a variety of datasets [40], including the Kitsune attack dataset [41], [42], and within an experimental test-bed [43]. Its high accuracy of 99.69% with a True Positive Rate (TPR) of 99.71%, and a True Negative Rate (TNR) of 98.48% is summarized in Figure 2.

However, relying solely on the AD is insufficient for system protection. Indeed, flood attacks lead to a huge packet accumulation at the AD, causing large queuing delays as illustrated by the red curves in Figure 5 in subsection IV-B. For instance, the Figure (above) shows a 10-second attack that floods the server with over $153,667$ malicious packets mixed with normal traffic from various devices in the network, overwhelming the AD processing capacity. During longer, 60-second attacks, Figure (below) shows that the attack causes over $400,000$ packets to accumulate and severely congest the AD, leading to the paralysis of the resource-constrained gateway server and prolonged downtimes or even system failure.

A. The Smart QDTP Policy Forwarder (SQF)

Thus, to protect the gateway server from being paralysed by the congestion that occurs when it is targeted by a flood attack, the initial system architecture shown in Figure 1 (above) was modified to incorporate traffic shaping with the Quasi-Deterministic Transmission Policy (QDTP) [44], which can be installed on a low-cost Raspberry Pi. The resulting architecture with the Smart QDTP Forwarder (SQF), is shown in Figure 1 (below). The SQF forwards the n-th arriving packet that arrives at time $a_n,\ n \geq 0$, to the server at time t_n defined by $t_0 = a_0$ and:

$$t_{n+1} = \max(t_n + D, a_{n+1}),\ n \geq 0, \quad (1)$$

$$hence:\ t_{n+1} - t_n \geq D\ , \quad (2)$$

TABLE I
COMPARISON OF RECENT MITIGATION APPROACHES IOT ENVIRONMENTS

No.	Reference	Approach	Limitations and Challenges
1	(Varalakshmi & Thenmozhi, 2025) [28]	The authors use entropy-based detection and stochastic techniques for mitigation and adaptive resource allocation to optimize energy efficiency and security. The approach targets DDoS attacks in SDN-IoT environments.	• May struggle in dynamic or large-scale IoT environments due to variability in entropy metrics.
2	(Mihoub el al., 2022) [29]	The authors propose ML-based detection method that is "Looking Back" for DoS/DDoS attacks in the IoT.	• High computational cost for some models. • Minimal performance gain in general classifiers. effectiveness depends on specific scenarios.
3	(Hayat et al., 2022) [30]	The authors introduce a framework that uses a distributed, device-level verification mechanism to identify and exclude (isolate) malicious devices via blockchain and smart contracts.	• Block-chain operations increase energy consumption, limiting feasibility on IoT devices.
4	(Li et al., 2021) [31]	The authors use federated learning at the fog layer for collaborative, privacy-preserving DDoS mitigation.	• Struggles with heterogeneous devices and various types of data. • High overhead and limiting applicability in low-resource WSNs.
5	(Lawal et al., 2021) [32]	The authors apply Fog-based DDoS mitigation using k-NN and a signature database for known threats.	• Relies on pre-labelled attack signatures, which are less effective for zero-day attacks. • Higher latency due to its dependency of the fog service.
6	(Galeano-Brajones et al., 2020) [33]	The authors utilize an entropy-based detection method integrated into a stateful SDN data plane to identify and mitigate DoS/DDoS attacks in IoT environments. The system dynamically analyzes flow states to detect abnormal traffic patterns.	• May overload SDN controllers. • Entropy-based statistical methods can be bypassed by low-rate or adaptive attacks.
7	**The system described in this paper**	**It employs a smart traffic shaping strategy that avoids IDS overlaod, accurately detects attacks, coupled with an adaptive packet dropping technique for operational efficiency during flood attacks.**	• **Works well in dynamic environments.** • **Offers low overhead and requires low computational power.** • **Can be trained on small datasets and has short training time.** • **Short decision time.**

where $D > 0$ is a constant parameter. Thus, the total delay Q_n experienced by the n-th packet is given by:

$$\begin{aligned} &Q_0 = t_0 - a_0 = 0,\ Q_{n+1} = t_{n+1} - a_{n+1}, \\ &= \max(t_n + D, a_{n+1}) - a_{n+1}, \\ &= 0,\ if\ t_n + D \leq a_{n+1},\ and \\ &= t_n + D - a_{n+1},\ otherwise. \end{aligned} \tag{3}$$

Since $t_n = Q_n + a_n$, we obtain the recursive expression:

$$\begin{aligned} &Q_{n+1} = \max(0, t_n + D - a_{n+1}), \\ &= \max(0, Q_n + D - a_{n+1}),\ n \geq 0. \end{aligned} \tag{4}$$

The actual transmission time from the Raspberry Pi to the gateway server was measured, including the Simple Network Management Protocol (SNMP) time taken at the server, and it was found to be less than 15% of the approximately $\tau \approx 3\ ms$ taken by the AD to process one packet. Therefore, when the n-th packet is forwarded by the SQF, we can assume that it instantly reaches the server's input queue for AD processing. The effect of the SQF was experimentally validated, and the blue curve in Figures 5 (above) and (below), clearly show that the SQF successfully protects the server during a flood attack, resulting in a server queue length that remains very short, both during and after

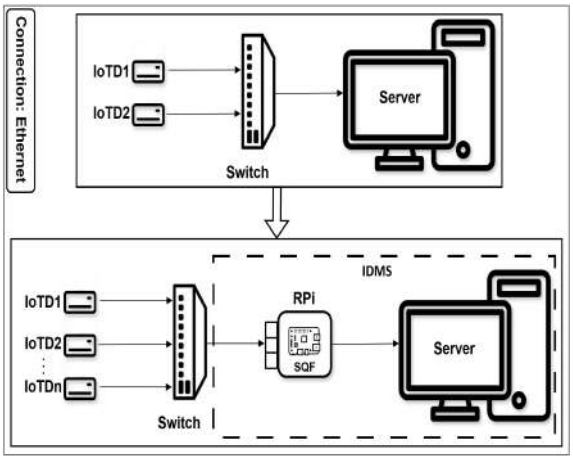

Fig. 1. The experimental test-bed consists of gateway devices connected directly to the server via a switch using Ethernet (above). In the modified architecture, the SQF is placed between the server and the gateway devices, isolating the server and serving as a traffic-shaping interface (below).

the attack.

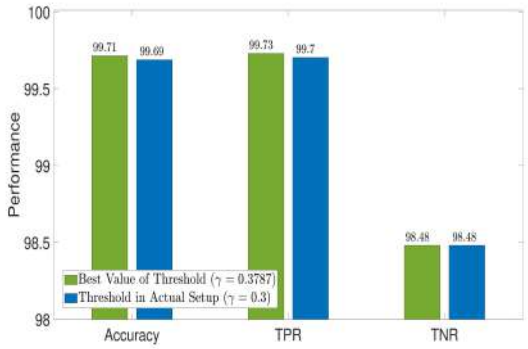

Fig. 2. Performance of the AADRNN attack detector that was evaluated on the test-bed [43].

B. Adaptive Attack Mitigation (AAM)

The blue curve in Figures 5 (above) and (below) shows that the SQF successfully protects the server from severe packet accumulation during a flood attack, maintaining the queue length of the server at normal levels both during and after the attack. However, while this approach safeguards the server from being overwhelmed, it does not stop the flow of attack packets; instead, these packets accumulate at the entrance of the SQF, creating a substantial backlog that must eventually be processed by the server. Thus, although the server is protected from paralysis, challenges such as high delays and benign packet loss persist.

To address these challenges, we introduce a novel AAM algorithm. This algorithm, featuring early attack detection, reduces the AD processing workload during attacks, thereby decreasing the computational overhead on the server, actively drops attack packets to reduce congestion, and promptly stops packet dropping after the attack ends to avoid the excessive loss of legitimate packets.

The algorithm begins by testing the first window of $W > 0$ consecutive packets sequentially. The size of this window W, is selected experimentally to optimize the accuracy of attack detection. If the AD identifies that a majority of the packets in the window are classified as "$ATTACK$," it concludes that an attack is occurring. It then drops the preceding $m+W$ packets to reduce congestion and skips $m > 0$ packets ahead in the incoming packet stream to test the next window of W packets. This ensures that the AD system is not overwhelmed during an attack and can continue operating efficiently. Conversely, if the AD detects "$NO-ATTACK$," the current window of W packets is forwarded to the server, and the algorithm proceeds to test the next W packets in the same manner, and the process is repeated.

C. Optimization of the Adaptive Attack Mitigation

During a flood attack, a fraction $(0 < f \leq 1)$ of incoming packets are part of the attack, while the remaining $1 - f$ are benign, and f is unknown in advance. While dropping packets, the AAM will also drop some benign packets originating from various devices in the network. Although the AAM reduces the number of packets tested by the AD during an attack, it still introduces computational overhead by testing W packets after every m-packet interval. Thus, in this subsection, we finalize the AAM by calculating the optimal value of m.

Let us denote by X the total number of packets received at the SQF during an attack. Since X is unknown in advance, it is treated as a random variable with the expected value $E[X]$. When AD initially identifies an attack within a W-packet window by identifying a majority of attack packets within the W-packet window, the first detection window serves as the starting point of the attack. The attack is considered to have ended when the AD detects a majority of non-attack packets in a subsequent W-packet window. The total number of detection windows N during the attack, and its expected value $E[N]$, are therefore:

$$N = \lceil \frac{X - W}{m + W} \rceil, \; E[N] \approx \frac{E[X] - W}{m + W} + \frac{1}{2}, \qquad (5)$$

where the expression for $E[N]$ is based on a mathematically proven [45] first order approximation.

Since the SQF ensures that the AD processing time per packet remains constant at a value τ, the server overhead Ω, and its expected value, are:

$$\Omega = N\tau W,\ E[\Omega] \approx \tau W\lceil\frac{E[X]-W}{m+W}+\frac{1}{2}\rceil\ . \quad (6)$$

During the attack, the number of packets dropped by the AAM and its expected value, are:

$$\delta = W + N(m+W),\ E[\delta] \approx E[X] + \frac{1}{2}(m+W).$$

Note that the dropped packets include those in the initial window classified as "$ATTACK$" and the following m packets, in a pattern that repeats N times. The final W-packet window is classified as "$NO-ATTACK$" and is not dropped.

Within the X packets which constitute the attack, we can assume that a fraction $0 < f \leq 1$ are attack packets, but there may also be $X(1-f)$ non-attack packets. Since the last $\delta - X$ packets that are dropped after the attack ends contain only benign packets, the AD's overhead for reprocessing all the lost benign packets, denoted by K, assuming that they are all re-sent by their sources sequentially and tested by the AD, in windows of W packets, will be:

$$K = \tau W\lceil\frac{fX+\delta-X}{W}\rceil,$$
$$E[K] \approx \tau[fE[X]+\frac{1}{2}m+W]. \quad (7)$$

Thus the total average cost is:

$$C(AAM) = \alpha E[K] + \beta E[\Omega], \quad (8)$$

where $\alpha > 0$ is the importance we attribute to the cost of AD processing of packets that were lost during the attack and which came back after the attack ended, while $beta > 0$ is the importance we attribute to the AD processing of packets during the attack. Presumably, we should have $\beta > \alpha$ if, during an attack, the server is overloaded with other urgent tasks such as packet dropping. In addition, we know that during an attack, and despite the presence of the SQF, the actual processing of packets by the AD is increased. Thus we would take $\frac{\beta}{\alpha} > 1$ if we wish to reduce the overhead of AD processing during an attack, while we would take $\frac{\beta}{\alpha} < 1$ if we wish to minimize the overhead throughout the system (including at the sources of traffic) caused by re-sending and re-processing the benign packets that were dropped by mistake during an attack.

Taking the derivative of the right-hand side of (8) with respect to m we get:

$$\frac{1}{\tau}\frac{dC(AAM)}{dm} \approx \frac{1}{2}\alpha - \beta W\frac{E[X]-W}{(m+W)^2}\ , \quad (9)$$

and equating it to zero, we see that the total average cost $C(AAM)$ is approximately minimized when setting $m = m^*$:

$$m^* \approx \sqrt{2\frac{\beta}{\alpha}W[E[X]-W\ ]}\ - W. \quad (10)$$

We note that m^* does not depend on f and τ, and it increases with the square root of $E[X]$. The minimum value of $C(AAM)$ computed at m^* is:

$$C^*(AAM) \approx \alpha\tau[fE[X]+\sqrt{\frac{\beta}{2\alpha}W[\ E[X]-W\ ]}+\frac{W}{2}] + \beta\tau W[\frac{E[X]-W}{\sqrt{2\frac{\beta}{\alpha}W[\ E[X]-W]}}+\frac{1}{2}]\ . \quad (11)$$

Figure 3 shows m^*, the value that minimizes the cost $C(AAM)$, for different values of $\frac{\beta}{\alpha}$ and $W = 20$ (which is the actual value we have set for the AD in our experimental work), and different values of $E[X]$, Such figures can be used to choose the value of m^* rapidly for different parameter sets.

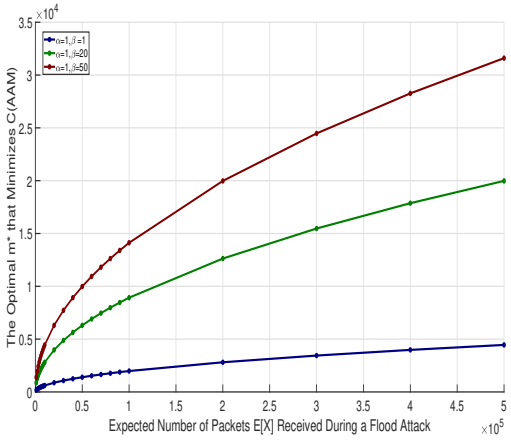

Fig. 3. Graph of the theoretical optimum value of m denoted m^*, which minimizes $C(AAM)$, plotted as a function of $E[X]$ for $W = 20$ and different values of $\frac{\beta}{\alpha}$.

IV. Results and Evaluation

In this section, we present the measurements and evaluation of the proposed system's behavior through real-time UDP flood attack experiments conducted on a test-bed. First, we describe the hardware and software configurations and settings of the test-bed prepared for the experiments. Then, we show the attack detection experiments conducted with and without the SQF, followed by the experimental measurements of the optimization of the proposed AAM algorithm.

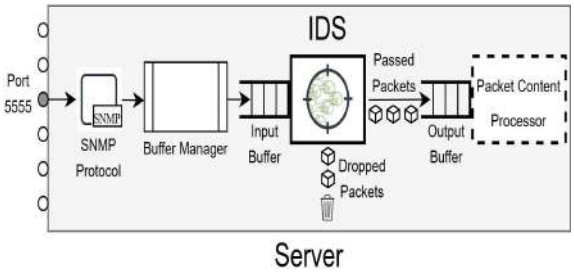

Fig. 4. Schematic representation of the server's software organization, featuring the simple network management processor (SNMP), the AD or Intrusion Detection System (labeled IDS in the figure), followed by the processing software for incoming data.

A. Hardware & Software Configurations

The test-bed on which we run our experiments includes Raspberry Pi 4 Model B Rev 1.2 processors acting as mobile or sensor devices of the IoT system. Each one has a 1.5 GHz ARM Cortex-A72 quad-core processor, 2 GB RAM, and runs Raspbian GNU/Linux 11. Some send attack traffic randomly or in a predetermined manner, while others send legitimate UDP packets with periodic machine temperature data to the server. The devices have a network buffer of 176 kB and communicate with the server via Ethernet as shown in Figure 1.

The gateway server is emulated by 3.1 GHz Intel 8-Core $i7$-8705G processor, 16 GB RAM, and Linux 5.15.0 (Ubuntu SMP), receives packets at port 5555 using the UDP protocol and processes them using SNMP 6.2.0-31-generic as shown in Figure 4. Its NIC supports 1000 $Mbps$ speeds in full duplex mode, with a 208 kB network buffer. We used UDP protocol due to its lightweight nature, which avoids the overhead of connection establishment and acknowledgments (ACKs) [46].

The Maximum Transmission Unit (MTU) is set to 1.5 $kB/packet$ for efficient packet transmission. Tests with 1000 packets showed low latency, averaging 0.437 ms, i.e., less than 15% of the server's AD processing time of $T_n \approx 3$ ms; therefore, this transmission delay is considered negligible in Section III.

To generate attack traffic, we used the MHDDoS public repository [47], which includes 56 real-world DoS emulators, enabling comprehensive testing with up-to-date scenarios.

B. Attack Detection

Without the SQF, the gateway server is exposed to a huge accumulation of packets at the entrance of the AD during a flood attack, as shown by the experimental results shown in red in Figure 5 (above) and (below). Even if the attack lasts only a few seconds, as seen in the first experiment (above), a queue of approximately 153, 667 packets forms when the attack lasts 10 seconds, and the server requires around 15 minutes to process them and resume normal operation. For the 60-second attack shown in the second experiment in the Figure (below), a substantially large queue exceeding 400, 000 packets forms at the server. This results in the system experiencing continuous interruptions during various periods, such as the period between minutes 133 and 183 of the experiment, where the server becomes paralyzed and unable to operate the AD or other services. Furthermore, the system remains at constant risk of failure under these conditions.

In contrast, when the SQF is used, the blue curves show that in both experiments, a short queue forms at the entrance of the server, and packets are processed normally without delays or interruptions. Naturally, fluctuations in the server's packet processing time result in the formation of a very small queue of no more than 20 to 30 packets.

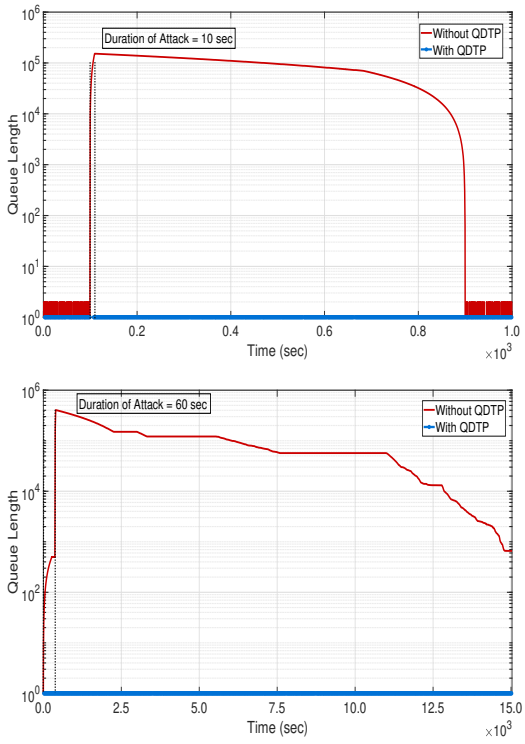

Fig. 5. The server queue length, measured experimentally and displayed on a logarithmic scale, is shown for a UDP flood attack lasting 10 seconds (above) and 60 seconds (below). The red curves represent the queue lengths without SQF, while the blue curves show the impact of SQF in reducing the queue length during both attacks duration, with $D = 3$ ms.

C. Attack Mitigation

This subsection presents the experimental evaluation of the proposed AAM algorithm that is described earlier in subsection III-C.

We conducted several experiments in which the parameter X was randomly generated. To evaluate the performance of the AAM algorithm, each experiment was repeated 30 times for a fixed expected number of packets $E[X]$ received during a flood attack. This process was repeated for different values of $E[X]$. The cost function $C(AAM)$ was calculated for each experiment, and its average value was computed by averaging the cost outcomes for different values of X and each specific $E[X]$. The experimental measurements in Figure 6 illustrate how the value of m increases with $E[X]$, and empirically demonstrate the effectiveness of cost minimization by choosing m^*.

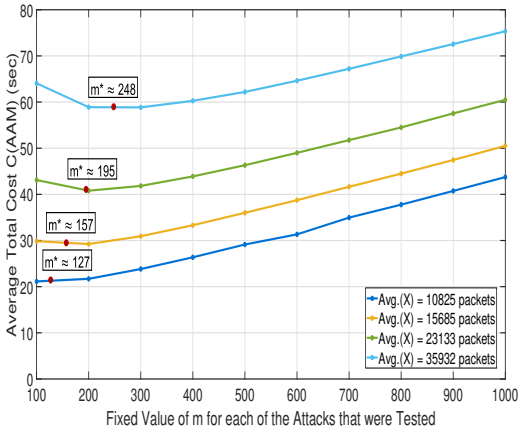

Fig. 6. Graph of the measured experimental value of the average total cost $C(AAM)$ against the parameter m.

Figure 7 (above) shows the queue length at the entrance of the SQF during an experiment in which the system is targeted by two flood attacks: the first containing approximately $10,000$ packets and the second $40,000$ packets. Upon receiving an alert from the AD indicating an attack, AAM calculates the value of m^*, which is 127 for the first attack and 248 for the second, as previously described by the theoretical formula. Figure 7 (below) shows the queue length at the entrance of the AD, which remains below 22 packets, illustrating the effectiveness of the AAM in making rapid decisions to drop packets, even when two attacks occur consecutively within a short time frame.

Combining the SQF and the AAM ensures a low queue length at the AD immediately after an attack starts, and facilitates decision-making to drop packets and minimize $C(AAM)$.

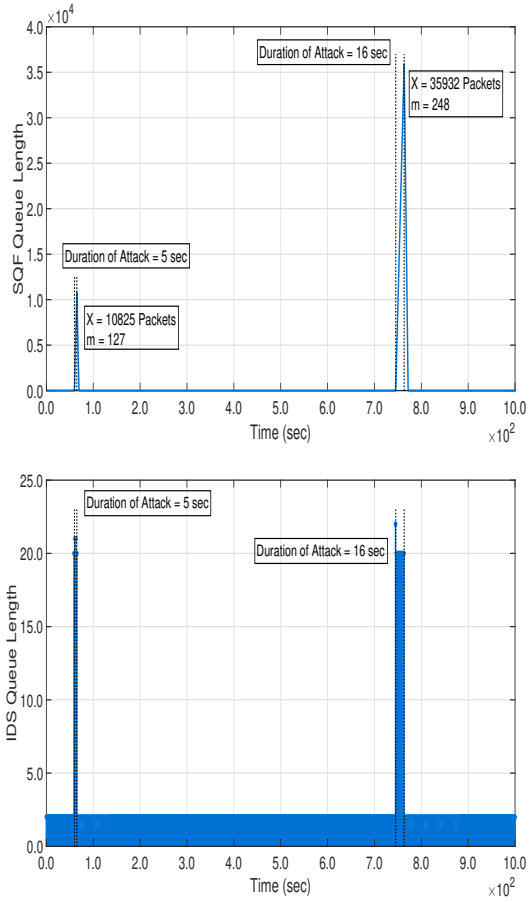

Fig. 7. Timeline of the queue length measurements at the entrance of the SQF (above) during two successive attacks: the first involves approximately $10,000$ packets, and the second about $40,000$ packets. At the entrance of the AD system, the combined use of SQF and AAM limits the input queue to around 20 packets (below). When AAM activates, it computes m^*, which is 127 for the first attack and 248 for the second attack.

V. Conclusions and Future Work

This paper sheds light on the impact of UDP flood attacks on resource-constrained gateway servers, showing that even short-duration attacks can overwhelm the server, resulting in prolonged backlogs and delays.

To address this challenge, we propose a new architecture featuring an SQF on a lightweight device implementing traffic shaping with the QDTP policy, which was introduced in [44] to protect the server from congestion, combined with our proposed AAM algorithm that samples and drops attack packets from the input stream, minimizing a cost function associated with benign packet drops and the sampling overhead.

The experimental results show that combining SQF with AAM effectively mitigates severe flood attacks.

Future work will explore IoT Systems with multiple

gateways, evaluate dynamic AD policies for complex networks of interconnected gateway networks with static and mobile nodes, and mitigation techniques that also minimize energy consumption.

ACKNOWLEDGMENT

This work was partially supported by the EU Horizon 2020 Project DOSS, Grant Agreement No. 101120270.

REFERENCES

[1] Cisco, *Cisco Annual Internet Report (2018–2023)*, Mar. 2020. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[2] J. Liu, L. Song *et al.*, "A novel congestion reduction scheme for massive machine-to-machine communication," *IEEE Access*, vol. 5, pp. 18 765–18 777, 2017.

[3] E. Johns and M. Ell, "Cyber security breaches survey 2023," April 2023. [Online]. Available: https://www.gov.uk/government/statistics/cyber-security-breaches-survey-2023/cyber-security-breaches-survey-2023

[4] Cloudflare. [Online]. Available: https://www.cloudflare.com/learning/ddos/famous-ddos-attacks/

[5] S. Evmorfos, G. Vlachodimitropoulos, N. Bakalos, and E. Gelenbe, "Neural network architectures for the detection of SYN flood attacks in IoT systems," in *Proceedings of the 13th ACM International Conference on PErvasive Technologies Related to Assistive Environments*, 2020, pp. 1–4.

[6] H. Sinanović and S. Mrdovic, "Analysis of mirai malicious software," in *2017 25th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 2017, pp. 1–5.

[7] A. Iqbal, S. Aftab, I. Ullah, M. A. Saeed, and A. Husen, "A classification framework to detect DoS attacks." *International Journal of Computer Network & Information Security*, vol. 11, no. 9, 2019.

[8] Q. He, X. Meng, R. Qu, and R. Xi, "Machine learning-based detection for cyber security attacks on connected and autonomous vehicles," *Mathematics*, vol. 8, no. 8, p. 1311, 2020.

[9] T. H. Aldhyani and H. Alkahtani, "Attacks to automatous vehicles: A deep learning algorithm for cybersecurity," *Sensors*, vol. 22, no. 1, p. 360, 2022.

[10] A. Kavousi-Fard, M. Dabbaghjamanesh, T. Jin, W. Su, and M. Roustaei, "An evolutionary deep learning-based anomaly detection model for securing vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 7, pp. 4478–4486, 2020.

[11] G. Gorbil, O. H. Abdelrahman, M. Pavloski, and E. Gelenbe, "Modeling and analysis of rrc-based signalling storms in 3g networks," *IEEE Transactions on Emerging Topics in Computing*, vol. 4, no. 1, pp. 113–127, 2015.

[12] M. Banerjee and S. Samantaray, "Network traffic analysis based iot botnet detection using honeynet data applying classification techniques," *International Journal of Computer Science and Information Security (IJCSIS)*, vol. 17, no. 8, 2019.

[13] E. Y. Güven and Z. Gürkaş-Aydın, "Mirai botnet attack detection in low-scale network traffic," *Intelligent Automation & Soft Computing*, vol. 37, no. 1, pp. 419–437, 2023.

[14] E. Gelenbe and M. Nakip, "Traffic based sequential learning during botnet attacks to identify compromised iot devices," *IEEE Access*, vol. 10, pp. 126 536–126 549, 2022.

[15] E. Gelenbe, "Energy packet networks: smart electricity storage to meet surges in demand," in *SIMUTOOLS '12: Proceedings of the 5th International ICST Conference on Simulation Tools and Techniques*, 2012, pp. 1–7.

[16] M. Kaouk, F.-X. Morgand, and J.-M. Flaus, "A testbed for cybersecurity assessment of industrial and IoT-based control systems," in *Lambda Mu 2018 - 21è Congrè de Maîtrise des Risques et Sûreté de Fonctionnement, Oct 2018, Reims, France*. [Online]. Available: https://hal.science/hal-02074654v1/document

[17] O. A. Waraga, M. Bettayeb, Q. Nasir, and M. A. Talib, "Design and implementation of automated iot security testbed," *Computers & security*, vol. 88, p. 101648, 2020.

[18] K. Kim, J. S. Kim, S. Jeong, J.-H. Park, and H. K. Kim, "Cybersecurity for autonomous vehicles: Review of attacks and defense," *Computers & Security*, vol. 103, p. 102150, 2021.

[19] H. Sedjelmaci, N. Kaaniche, A. Boudguiga, and N. Ansari, "Secure attack detection framework for hierarchical 6g-enabled internet of vehicles," *IEEE Transactions on Vehicular Technology*, 2023.

[20] X. Li, Z. Hu, M. Xu, Y. Wang, and J. Ma, "Transfer learning based intrusion detection scheme for internet of vehicles," *Information Sciences*, vol. 547, pp. 119–135, 2021.

[21] E. Gelenbe, B. C. Gul, and M. Nakip, "Disfida: Distributed self-supervised federated intrusion detection algorithm with online learning for health internet of things and internet of vehicles," *Internet of Things*, vol. 28, no. https://doi.org/10.1016/j.iot.2024.10134, p. 101340, 2024.

[22] M. Aloqaily, S. Otoum, I. Al Ridhawi, and Y. Jararweh, "An intrusion detection system for connected vehicles in smart cities," *Ad Hoc Networks*, vol. 90, p. 101842, 2019.

[23] S. Aurangzeb, M. Aleem, M. T. Khan, H. Anwar, and M. S. Siddique, "Cybersecurity for autonomous vehicles against malware attacks in smart-cities," *Cluster Computing*, pp. 1–16, 2023.

[24] F. Pascale, E. A. Adinolfi, S. Coppola, and E. Santonicola, "Cybersecurity in automotive: An intrusion detection system in connected vehicles," *Electronics*, vol. 10, no. 15, p. 1765, 2021.

[25] A. Ghaleb, S. Zhioua, and A. Almulhem, "Scada-sst: a scada security testbed," in *2016 World Congress on Industrial Control Systems Security (WCICSS)*. IEEE, 2016, pp. 1–6.

[26] T. Alladi, V. Kohli, V. Chamola, F. R. Yu, and M. Guizani, "Artificial intelligence (ai)-empowered intrusion detection architecture for the internet of vehicles," *IEEE Wireless Communications*, vol. 28, no. 3, pp. 144–149, 2021.

[27] E. Eziama, F. Awin, S. Ahmed, L. Marina Santos-Jaimes, A. Pelumi, and D. Corral-De-Witt, "Detection and identification of malicious cyber-attacks in connected and automated vehicles' real-time sensors," *Applied Sciences*, vol. 10, no. 21, p. 7833, 2020.

[28] I. Varalakshmi and M. Thenmozhi, "Energy optimization using adaptive control algorithm to enhance the performance of sdn_iot environment," *Discover Internet of Things*, vol. 5, no. 1, p. 27, 2025.

[29] A. Mihoub, O. B. Fredj, O. Cheikhrouhou, A. Derhab, and M. Krichen, "Denial of service attack detection and mitigation for internet of things using looking-back-enabled machine learning techniques," *Computers & Electrical Engineering*, vol. 98, p. 107716, 2022.

[30] R. F. Hayat, S. Aurangzeb, M. Aleem, G. Srivastava, and J. C.-W. Lin, "Ml-ddos: A blockchain-based multilevel ddos mitigation mechanism for iot environments," *IEEE Transactions on Engineering Management*, 2022.

[31] J. Li, L. Lyu, X. Liu, X. Zhang, and X. Lyu, "Fleam: A federated learning empowered architecture to mitigate ddos in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 6, pp. 4059–4068, 2021.

[32] M. A. Lawal, R. A. Shaikh, and S. R. Hassan, "A ddos attack mitigation framework for iot networks using fog computing," *Procedia Computer Science*, vol. 182, pp. 13–20, 2021.

[33] J. Galeano-Brajones, J. Carmona-Murillo, J. F. Valenzuela-Valdés, and F. Luna-Valero, "Detection and mitigation of dos

and ddos attacks in iot-based stateful sdn: An experimental approach," *Sensors*, vol. 20, no. 3, p. 816, 2020.
[34] O. Brun, Y. Yin, and E. Gelenbe, "Deep learning with dense random neural network for detecting attacks against iot-connected home environments," *Procedia Computer Science*, vol. 134, pp. 458–463, 2018.
[35] E. Gelenbe, "Random neural networks with negative and positive signals and product form solution," *Neural Computation*, vol. 1, no. 4, pp. 502–510, 1989.
[36] E. Gelenbe and M. Nakip, "Iot network cybersecurity assessment with the associated random neural network," *IEEE Access*, vol. 11, pp. 85 501–85 512, 2023.
[37] E. Gelenbe and Y. Yin, "Deep learning with random neural networks," in *2016 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2016, pp. 1633–1638.
[38] E. Gelenbe, "G-networks with instantaneous customer movement," *Journal of Applied Probability*, vol. 30, no. 3, pp. 742–748, 1993.
[39] ——, "G-networks: a unifying model for neural and queueing networks," *Annals of Operations Research*, vol. 48, no. 5, pp. 433–461, 1994.
[40] E. Gelenbe and M. Nakıp, "G-networks can detect different types of cyberattacks," in *2022 30th International Symposium on Modeling, Analysis, and Simulation of Computer and Telecommunication Systems (MASCOTS)*. IEEE, 2022, pp. 9–16.
[41] Y. Mirsky, T. Doitshman, Y. Elovici, and A. Shabtai, "Kitsune: An ensemble of autoencoders for online network intrusion detection," in *The Network and Distributed System Security Symposium (NDSS) 2018*, 2018.
[42] "Kitsune Network Attack Dataset," August 2020. [Online]. Available: https://www.kaggle.com/ymirsky/network-attack-dataset-kitsune
[43] M. Nasereddin, M. Nakıp, and E. Gelenbe, "Measurement based evaluation and mitigation of flood attacks on a lan test-bed," in *The 48th IEEE Conference on Local Computer Networks (LCN) October 1-5, 2023, Daytona Beach, Florida, USA*. IEEEXpress, 2023, pp. 1–4. [Online]. Available: https://zenodo.org/record/8094796
[44] E. Gelenbe and K. Sigman, "Iot traffic shaping and the massive access problem," in *ICC 2022, IEEE International Conf. on Comms., 16–20 May 2022, Seoul, South Korea*. https://zenodo.org/record/5918301, 2022, pp. 1–6.
[45] E. Gelenbe, J. C. A. Boekhorst, and J. L. W. Kessels, "Minimizing wasted space in partitioned segmentation," *Commun. ACM*, vol. 16, no. 6, p. 343–349, jun 1973. [Online]. Available: https://doi.org/10.1145/362248.362253
[46] M. Masirap, M. H. Amaran, Y. M. Yussoff, R. Ab Rahman, and H. Hashim, "Evaluation of reliable udp-based transport protocols for internet of things (iot)," in *2016 IEEE Symposium on Computer Applications & Industrial Electronics (ISCAIE)*. IEEE, 2016, pp. 200–205.
[47] "MHDDoS - DDoS Attack Script With 56 Methods," Online, May 2022, accessed: 2023-02-22. [Online]. Available: https://github.com/MatrixTM/MHDDoS