Hunting IoT Cyberattacks With AI-Powered Intrusion Detection

Sevasti Grigoriadou, Panagiotis Radoglou-Grammatikis, Panagiotis Sarigiannidis
University of Western Macedonia
Kozani, Greece
sevinagrigoriadou@gmail.com, {pradoglou, psarigiannidis}@uowm.gr

Ioannis Makris
MetaMind Innovations P.C.
Kozani, Greece
imakris@metamind.gr

Thomas Lagkas
International Hellenic University
Kavala, Greece
tlagkas@cs.ihu.gr

Vasileios Argyriou
Kingston University London
London, UK
vasileios.argyriou@kingston.ac.uk

Anastasios Lytos, Eleftherios Fountoukidis
Sidroco Holdings Ltd
Nicosia, Cyprus
{alytos, efountoukidis}@sidroco.com

***Abstract*—The rapid progression of the Internet of Things allows the seamless integration of cyber and physical environments, thus creating an overall hyper-connected ecosystem. It is evident that this new reality provides several capabilities and benefits, such as real-time decision-making and increased efficiency and productivity. However, it also raises crucial cybersecurity issues that can lead to disastrous consequences due to the vulnerable nature of the Internet model and the new cyber risks originating from the multiple and heterogeneous technologies involved in the IoT. Therefore, intrusion detection and prevention are valuable and necessary mechanisms in the arsenal of the IoT security. In light of the aforementioned remarks, in this paper, we introduce an Artificial Intelligence (AI)-powered Intrusion Detection and Prevention System (IDPS) that can detect and mitigate potential IoT cyberattacks. For the detection process, Deep Neural Networks (DNNs) are used, while Software Defined Networking (SDN) and Q-Learning are combined for the mitigation procedure. The evaluation analysis demonstrates the detection efficiency of the proposed IDPS, while Q-Learning converges successfully in terms of selecting the appropriate mitigation action.**

I. Introduction

Although the Internet of Things (IoT) started as a conceptual viewpoint regarding remote monitoring and pervasive control, it has revolutionised the interconnection between the cyber and physical environments, thus forming a hyper-connected ecosystem with several benefits, such as enhanced efficiency, cost savings and environmental sustainability [1]. In particular, nowadays, IoT is composed of several and heterogeneous technology enablers that are seamlessly integrated into various industrial sectors, such as energy, healthcare and transportation. However, this evolution raises critical security risks that can lead to devastating effects, especially in critical domains, as those mentioned before. In particular, first, IoT relies on the conventional Internet model, which is characterised by severe cybersecurity issues [2]. On the other hand, IoT includes various technologies that introduce their own vulnerabilities. Finally, it is worth mentioning that IoT applications can gather and collect sensitive data, which is an attractive goal for potential cybercriminals and Advanced Persistent Threat (APT) campaigns [3]. Characteristic examples of APT campaigns that target IoT ecosystems are `Indstroyer`, `Dragonfly` and `APT28`.

Therefore, it is evident that the presence of intrusion detection and prevention mechanisms plays a pivotal role in safeguarding IoT environments. Their goal is to detect and mitigate potential malicious activities in real-time, thus allowing the organisations to maintain the normal operation of their environments in terms of the principal security requirements, such as (a) confidentiality, (b) integrity and (c) availability. The first intrusion detection model was defined by D. Denning in 1987 [4]. In particular, there are two main intrusion detection techniques: (a) signature/specification-based detection and (b) anomaly-based detection [5]. The first technique adopts predefined rules that can be used to detect malicious patterns. On the other hand, anomaly-based detection relies on statistical analysis and Artificial Intelligence (AI) that can identify potential anomalies. On the one hand, signature/specification-based detection is more accurate but cannot discriminate zero-day attacks and unknown anomalies. Finally, such mechanisms can activate mitigation measures, such as firewall rules, in order to mitigate a potential cyberattack after its detection [6].

In light of the aforementioned remarks, in this paper, we introduce an AI-powered Intrusion Detection and Prevention System (IDPS) that can detect and mitigate potential cyberattacks against IoT environments. For this purpose, Deep Neural Networks (DNNs) are utilised for the detection process, while Software-Defined Networking (SDN) and Q-Learning are combined to mitigate the potential attacks. In particular, for the detection process, Multi-Layer Perceptron (MLP) models are trained with the `CIC IoT Dataset 2022` [7], selecting the model with the best detection efficiency. A particular emphasis is given to two cyberattacks: Flood attacks and Real Time Streaming Protocol (RTSP) bruteforce attacks. On the other hand, given an SDN environment, a Q-Learning agent is used to indicate to the SDN Controller (SDN-C) the appropriate mitigation action.

The rest of this paper is organised as follows. Section II discusses similar works in this field. Section III introduces the architecture of the proposed IDPS. Next, section IV emphasises on the detection process, while section V focuses on the mitigation mechanisms. In section VI, the evaluation results are presented. Finally, section VII concludes this paper.

II. Related Work

Several works have already investigated the security issues of IoT. First, some indicative and comprehensive survey papers in this field are summarised, while next, we focus on IDPS for IoT applications. In particular, in [2], the authors analyse the security and privacy issues of IoT, following a four-layered IoT architecture. Similarly, in [8], I. Makhdoom provide an anatomy of IoT threats. In [9], B. R. Zarpelao et al. provide a survey paper about intrusion detection in IoT. In [10], the authors focus on Machine Learning (ML)-based intrusion detection for IoT. In [11], E. F. Jesus et al. introduce a detailed analysis about how blockchain can be used to secure IoT. In [12], I. Farris et al. investigate SDN and Network Function Virtualisation (NFV)-enabled security mechanisms for the IoT. Finally, in [13] J. Franco et al. introduce a survey of honeypots and honeynets for IoT and industrial IoT. Therefore, based on the aforementioned remarks, it is evident that security is one of the most critical aspects of the IoT lifecycle.

In [14], X. Liu et al. present a specification-based Intrusion Detection System (IDS) that focuses on securing smart meter communications. The authors begin by introducing an information model for smart meter modules using Coloured Petri Net (CPN). They also define a threat model, which encompasses two main classes of attacks: (a) data attacks and (b) command attacks. Subsequently, they propose an IDS specifically targeting false data injection attacks. The architectural design of this IDS consists of three modules: (a) Secret Information, (b) Event Log, and (c) Spying Domain. The Secret Information module is a private data structure accessible through legitimate actions and is used to encrypt the Event Log, which stores activities related to the smart meters. The Spying Domain comprises random storage areas containing hash codes of the Secret Information. Whenever a malicious user attempts to access the storage units of the Event Log or the Spying Domain, a security alert is generated. The effectiveness of the proposed IDS is demonstrated through evaluation diagrams showcasing the True Positive Rate (TPR).

M. Attia et al. developed a specification-based IDS for Advanced Metering Infrastructure (AMI) in their work [15]. The IDS incorporates temporal and spatial detection methods and specifically focuses on detecting blackhole and time delay attacks. A blackhole attack is a type of Denial-of-Service (DoS) attack, while a time delay attack aims to introduce additional time during the transmission of network packets. The security specifications of the IDS are defined based on the number of transmitted packets and the delay time between them. This involves calculating the mean value and standard deviation of a Gaussian distribution. The authors showcase the efficiency of their proposed method by comparing it with three other methods: (a) spatial-based method, (b) temporal-based method, and (c) SVM classifier. While the SVM classifier achieves the best True Positive Rate (TPR), the proposed IDS performs the best in terms of False Positive Rate (FPR).

In [16], the authors provide an AI-based IDPS with a self-learning mechanism. The proposed IDPS is composed of three modules: (a) Network Flow Monitoring and Collection Module, (b) Intrusion Detection Engine and (c) Notification and Response Module. The first module is responsible for collecting the network traffic data and generating flow statistics. The second module uses pre-trained AI models in order to detect potential attacks against Hypertext Transfer Protocol (HTTP) and Modbus/Transmission Control Protocol (TCP). Moreover, this module includes an active learner that allows the re-training process in a dynamic manner. Finally, the Notification and Response Module generates the corresponding security events based on the detection outcomes of the Intrusion Detection Engine.

In [17], T. Xing et al. introduce a Software-Defined Networking (SDN) based Intrusion Prevention System (IPS), SDNIPS, which is comprised of four key components: a Snort module, an SDNIPS daemon, an alert interpreter, and a rule generator. This system identifies potential cyber threats and transforms this information into JSON format for the SDN controller. The alert interpreter then parses this data for relevant information, while the rule generator creates OpenFlow entries for integration into Open vSwitch flow tables. The researchers establish the effectiveness of SDNIPS by simulating two Denial of Service (DoS) attacks and comparing it against a traditional IPS that relies on iptables, demonstrating that SDNIPS outperforms the conventional system under high network traffic conditions.

H. Lin [18] proposes an SDN-based in-network honeypot that mitigates cyberattacks using two methods. First, it isolates attackers by disrupting their communication with legitimate nodes. Second, it employs network communication spoofing via phantom nodes to deceive attackers and gather information. Initially, the SDN controller identifies and isolates malicious nodes by disrupting their communication. Then, it establishes communication with the attacker using spoofed IP addresses. Network packet content is modified at the network and application layers using statistical and physical models.

Undoubtedly, the previous works provide useful solutions and methodologies. In particular, it is obvious that many researchers try to take full advantage of AI in terms of ML and Deep Learning (DL) methods in order to detect potential attacks. It is worth mentioning that despite the fact that both ML and DL-based intrusion detection mechanisms can be characterised by potential alarms, the authors continue investigating and actively working on such solutions with the goal to minimise the generalisation error by optimising the training procedure. On the other hand, there are not a lot of papers that investigate how AI can be used to identify the optimal mitigation strategy in a cyberdefence game. In this work, we aim to cover this gap by combining SDN and Q-Learning based on the detection outcomes of a pre-trained DNN.

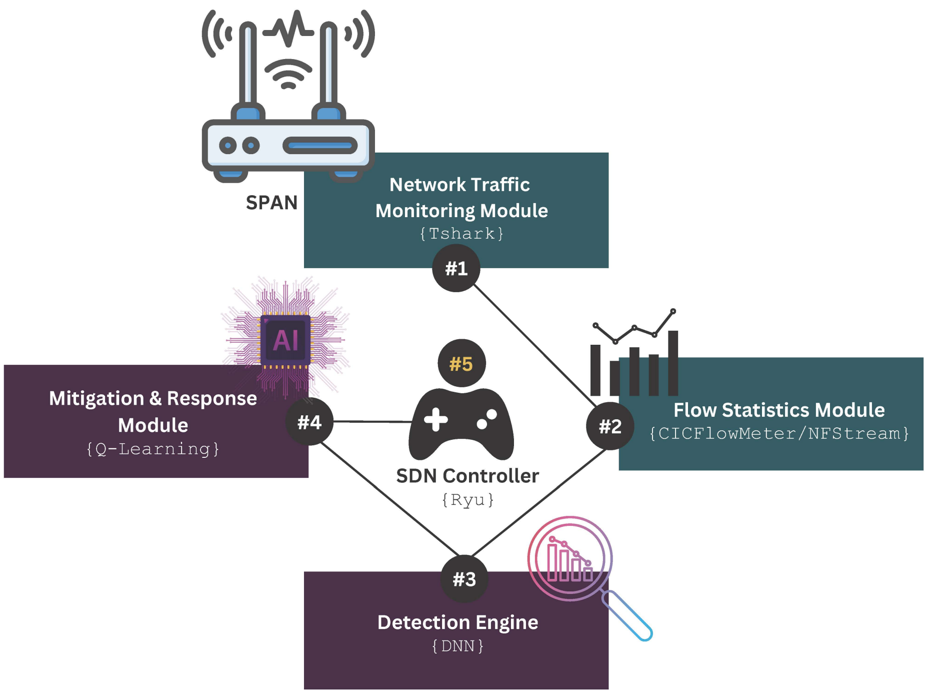

Fig. 1. Architecture of the Proposed IDPS

III. Architecture of the Proposed Intrusion Detection and Prevention System

As illustrated in Fig. III, the architecture of the proposed IDPS consists of four main modules, namely: (a) Network Traffic Monitoring Module, (b) Flow Statistics Module, (c) Detection Engine and (d) Mitigation and Response Module. The first module uses `Tshark` in order to capture the network traffic data. To this end, this module receives the overall network traffic through a Switched Port Analyser (SPAN) (i.e., port mirroring). SPAN is a feature in network switches that allows monitoring and analysis of network traffic. It duplicates specific switch ports or Virtual Local Area Network (VLAN) traffic to a monitoring device for troubleshooting, security monitoring, and performance analysis without affecting normal network flow. Next, the Flow Statistics Module receive the network traffic data (i.e., pcap files) and generates TCP/Internet Protocol (IP) bidirectional flow statistics, utilising both `CICFlowMeter` and `NFStream`. `CICFlowMeter`, previously referred to as `ISCXFlowMeter`, is a tool used for generating and analysing Ethernet traffic bidirectional flows to detect abnormalities. It has been widely utilised in various cybersecurity datasets. On the other hand, `NFStream` is a Python package that offers efficient and adaptable data structures for seamless handling of online or offline network flow analysis. Next, based on the flow statistics from `CICFlowMeter` and `NFStream`, the Detection Engine uses pre-trained MLPs in order to detect potential attacks. In particular, MLP is a type of a neural network with multiple interconnected layers of nodes. It uses weights and activation functions to process input data and make predictions. Finally, the Mitigation and Response Module receives the detection outcomes and guides the SDN-C to execute the appropriate network-related mitigation action. In our case, `Ryu` is used as SDN-C. The decisions of the Mitigation and Response Module rely on a Q-Learning agent trained offline. More details about the detection and mitigation processes are provided in the following actions.

IV. Detection of IoT Cyberattacks

As mentioned, for the detection process, MLP networks are used. MLP is a feedforward neural network, meaning that information flows in one direction, from the input layer through the hidden layers to the output layer. Each neuron in an MLP receives inputs, which are multiplied by corresponding weights and passed through an activation function. The activation function introduces non-linearity into the network and helps the MLP model complex relationships in the data. The weighted inputs are summed at each neuron, and the result is passed through the activation function to produce the output. MLPs are typically organised into an input layer, one or more hidden layers, and an output layer. The hidden layers contain one or more layers of neurons between the input and output layers. Each neuron in the hidden layers receives inputs from the previous layer and produces outputs for the next layer. The training process of an MLP involves adjusting the weights of the connections between neurons to minimise the error between the predicted output and the desired output. This is typically done using backpropagation, an iterative optimisation algorithm that updates the weights based on the calculated error. As illustrated in Fig. 2-Fig. 4, three MLPs are investigated in our case with two, three and four hidden layers.

In the first case, with two hidden layers, each hidden layer has 30 neurons. In the second case, with three hidden layers, each hidden layer has 60 neurons. Finally, in the last case, with four hidden layers, each hidden layer has 80 neurons. Based on several experiments, ReLu (Equation 1) is used between the hidden layers, while Softmax (Equation 2) is used in each output layer. Finally, all MLPs are trained with the `CIC IoT Dataset 2022` in terms of discriminating three classes: (a) Normal, (b) Flood and (c) RSTP bruteforce. For each of them, the Mean Squared Error (MSE) (Equation 3) is used as a loss function for the training procedure.

$$\text{ReLU}(x) = \max(0, x) \tag{1}$$

$$\text{Softmax}(x_i) = \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \tag{2}$$

where: $\text{Softmax}(x_i)$ represents the softmax value for the i-th element of the vector. x_i is the input value for the i-th element of the vector. n is the total number of elements in the vector.

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \bar{y})^2 \tag{3}$$

where: n is the total number of data points. y_i denotes the actual value of the i-th data point. $\bar{y}$ represents the predicted value of the i-th data point.

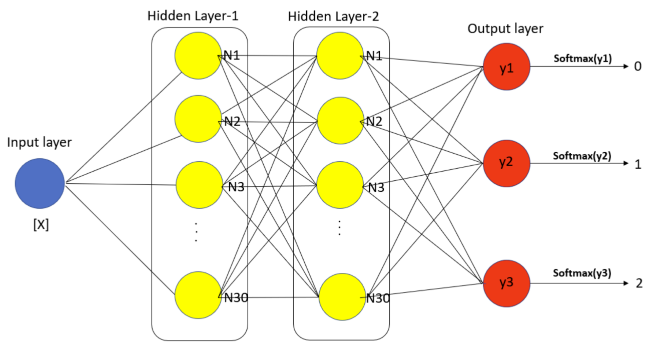

Fig. 2. MLP with Two Hidden Layers

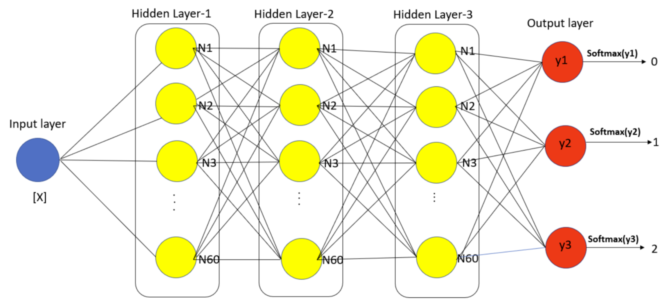

Fig. 3. MLP with Three Hidden Layers

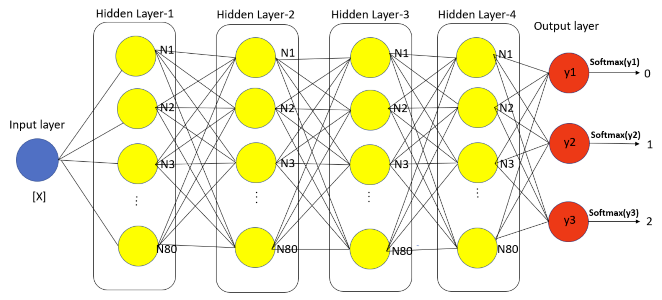

Fig. 4. MLP with Four Hidden Layers

V. Mitigation of IoT Cyberattacks

The Mitigation and Response Module relies on Q-Learning, which is a Reinforcement Learning (RL) algorithm that enables an agent to learn an optimal policy in a Markov Decision Process (MDP) by iteratively updating its action-value function, known as the Q-function. In Q-Learning, the agent aims to maximise its long-term cumulative reward by iteratively updating the Q-values associated with state-action pairs. The Q-value represents the discounted expected future reward the agent will receive by taking a particular action in a specific state. The Q-function is typically represented as a lookup table (Q-table), where each entry corresponds to a state-action pair and its associated Q-value. In our case, based on the detection outcomes, the state space consists of three states: (a) Normal State, (b) Flood State and (c) Bruteforce State, while the action space is composed of four actions: (a) Rerouting, (b) Rate Limiting, (c) Network Isolation and (d) Notification. Initially, the Q-table (Tabel I) is initialised (in our case with zeros) arbitrarily, and the agent takes actions in the environment based on an exploration-exploitation strategy (in our case, epsilon-greedy is used) to balance between exploring new actions and exploiting the current knowledge. As the agent interacts with the environment, it updates the Q-values in the Q-table based on the observed rewards and the maximum Q-value achievable in the next state. The Q-values are updated with Equation 4.

TABLE I
Initial Visual Representation of Q-Table

State/Action	Rerouting	Rate Limiting	Isolation	Notification
Normal	0	0	0	0
Flood	0	0	0	0
Bruteforce	0	0	0	0

$$Q(s,a) \leftarrow Q(s,a) + \alpha\left[R(s,a) + \gamma \max_{a'} Q(s',a') - Q(s,a)\right] \tag{4}$$

where $Q(s,a)$ is the Q-value for state s and action a. $R(s,a)$ is the immediate reward obtained from taking action a in state s. γ is the discount factor. Finally, the term $max_{a'}Q(s',a')$ represents the maximum Q-value over all possible actions a' in the next state s'. In the previous equation, the reward $R(s,a)$ is calculated by Equation 5 in a non-linear manner.

$$reward = w1 \cdot (\mathrm{L})^2 - w2 \cdot e^{\mathrm{CA}} - w3 \cdot \log(1 + \mathrm{N}) \quad (5)$$

where L is the network latency, CA denotes the cost of each action and N indicates the number of security events that are related to this state. Finally, $w1$, $w2$ and $w3$ denote the hyperparameters that can affect each of the previous factors. Based on the aforementioned remarks, Algorithm 1 provides the overall Q-Learning for the mitigation process in our case.

Algorithm 1 Q-Learning Algorithm

```
1: Initialize Q-table with arbitrary or zero values for all state-action pairs
2: Set learning rate $\alpha$, discount factor $\gamma$, and exploration rate $\epsilon$
3: while not converged do
4:     Observe current state $s$
5:     Choose action $a$ based on an $\epsilon$-greedy policy
6:     Execute action $a$, observe reward $r$ and next state $s'$
7:     Calculate dynamic reward using the reward formula
8:     Update Q-value for current state-action pair:
9:         $Q(s,a) \leftarrow Q(s,a) + \alpha\,[r + \gamma \max(Q(s',a')) - Q(s,a)]$
10:    Update current state: $s \leftarrow s'$
11: end while
```

Finally, it is worth mentioning that, in our case, an offline dataset based on domain knowledge is utilised for the training procedure (offline Q-Learning). In particular, this dataset includes tuples consisting of a state, an action, a reward, and the next state. A visual representation of this dataset is provided in Table II.

TABLE II
VISUAL REPRESENTATION OF Q-LEARNING DATASET

State	Action	Reward	Next State
Normal State	Rerouting	-1	Normal State
Normal State	Blacklisting	-5	Normal State
Flood	Rate Limiting	10	Normal State
Flood	Rerouting	0	Flood
Bruteforce	Isolation	10	Normal State
⋮	⋮	⋮	⋮

VI. Evaluation Analysis

Before delving into the evaluation results pertaining to detection, it's vital to understand the key terminologies. True Positives (TP) are the instances where the cyberattacks are correctly identified as intrusions. Conversely, True Negatives (TN) are cases where normal network packets are accurately identified as such. In contrast, False Negatives (FN) represent incorrect classifications where cyberattacks are misidentified as normal, and False Positives (FP) are instances where standard network behaviours are erroneously labelled as intrusions. Considering these terms, several evaluation metrics are applied. It should be noted that the comparative study utilises a range of Machine Learning (ML) techniques including Decision Tree, K-Nearest Neighbour (KNN), Random Forest, Naive Bayes, Support Vector Machine (SVM) with different kernels, AdaBoost, Linear Discriminant Analysis (LDA), and Stochastic Gradient Descent (SGD) classifier.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

$$TPR = \frac{TP}{TP + FN} \quad (7)$$

$$FPR = \frac{FP}{FP + TN} \quad (8)$$

$$F1 = \frac{2 \times TP}{2 \times TP + FP + FN} \quad (9)$$

Table III summarises the evaluation results, utilising the `CICFlowMeter` flow statistics. Based on them, the MLP with the four hidden layers (described) above achieves the best detection efficiency, in terms of $Accuracy = 0.998$, $TPR = 0.997$, $FPR = 0.001$ and $F1 = 0.997$. Fig 5 shows the confusion matrix of this model.

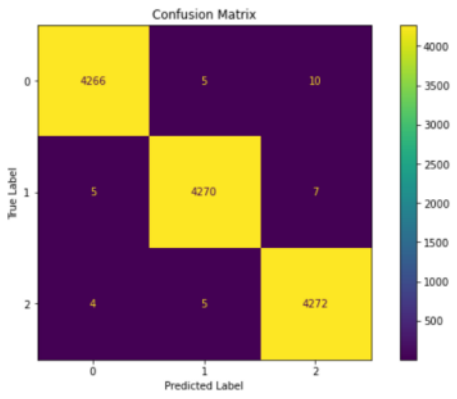

Fig. 5. Confusion Matrix of MLP with 4 Hidden Layers trained on CICFlowMeter Flow Statistics

TABLE III
DETECTION EVALUATION RESULTS OF THE PROPOSED IDPS WITH THE CICFLOWMETER FLOW STATISTICS

AI Models	Accuracy	TPR	FPR	F1-Score
MLP-2 hidden layer	0.92	0.916	0.041	0.919
MLP-3 hidden layers	0.98	0.987	0.006	0.98
MLP-4 hidden layers	**0.998**	**0.997**	**0.001**	**0.997**
Decicion Tree	**0.997**	**0.997**	**0.001**	**0.997**
k-NN	0.969	0.968	0.015	0.969
Random Forest	0.994	0.994	0.002	0.994
Naïve Bayes	0.812	0.811	0.094	0.812
SVM-Linear	0.964	0.964	0.017	0.964
SVM-RBF	0.966	0.966	0.016	0.966
SVM-Sigmoid	0.82	0.819	0.09	0.82
Logistic Regression	0.937	0.929	0.035	0.938
AdaBoost	0.911	0.91	0.04	0.911
LDA	0.874	0.89	0.05	0.875
SGD	0.938	0.938	0.03	0.938

Similarly, Table IV summarises the evaluation results, utilising the `NFStream` flow statistics. Based on them, the MLP with the four hidden layers (described) above achieves the best detection efficiency, in terms of $Accuracy = 0.999$, $TPR = 0.999$, $FPR = 0.001$ and $F1 = 0.999$. Fig 6 shows the confusion matrix of this model.

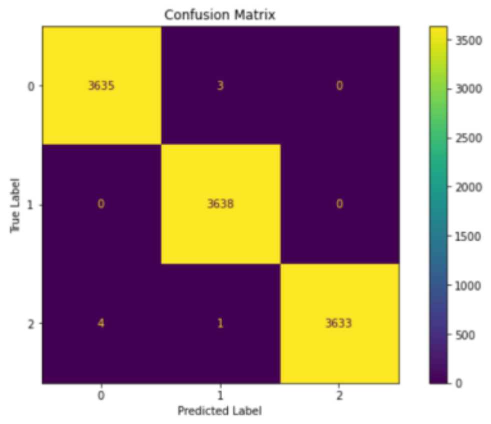

Fig. 6. Confusion Matrix of MLP with 4 Hidden Layers trained on NFStream Flow Statistics

TABLE IV
DETECTION EVALUATION RESULTS OF THE PROPOSED IDPS WITH THE NFSTREAM FLOW STATISTICS

AI Models	Accuracy	TPR	FPR	F1-Score
MLP-2 hidden layer	0.998	0.998	0.008	0.998
MLP-3 hidden layers	0.997	0.997	0.001	0.997
MLP-4 hidden layers	0.999	0.999	0.001	0.999
Decicion Tree	0.972	0.972	0.013	0.972
k-NN	0.997	0.997	0.012	0.997
Random Forest	0.98	0.98	0.01	0.98
Naïve Bayes	0.935	0.934	0.032	0.935
SVM-Linear	0.932	0.931	0.034	0.932
SVM-RBF	0.983	0.983	0.008	0.983
SVM-Sigmoid	0.8	0.798	0.01	0.8
Logistic Regression	0.897	0.897	0.05	0.897
AdaBoost	0.891	0.889	0.05	0.897
LDA	0.867	0.866	0.06	0.867
SGD	0.872	0.871	0.064	0.872

VII. CONCLUSIONS

The rapid growth of the IoT brings numerous benefits, like real-time decision-making and increased efficiency. However, it also increases the attack surface due to the insecure Internet model and weaknesses of the new technologies involved in the IoT. In this paper, we present an AI-powered IDPS for IoT environments. The proposed IDPS focuses on flood and bruteforce attacks, utilising an MLP with four hidden layers for the detection process, while SDN and Q-Learning are used for the mitigation. The evaluation results demonstrate the effectiveness of the proposed IDPS in detecting cyberattacks, while Q-Learning successfully determines the appropriate mitigation action.

ACKNOWLEDGMENT

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 101021936 (ELECTRON).

REFERENCES

[1] S. Kumar, P. Tiwari, and M. Zymbler, "Internet of things is a revolutionary approach for future technology enhancement: a review," *Journal of Big data*, vol. 6, no. 1, pp. 1–21, 2019.

[2] P. I. R. Grammatikis, P. G. Sarigiannidis, and I. D. Moscholios, "Securing the internet of things: Challenges, threats and solutions," *Internet of Things*, vol. 5, pp. 41–70, 2019.

[3] A. Alshamrani, S. Myneni, A. Chowdhary, and D. Huang, "A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1851–1877, 2019.

[4] D. E. Denning, "An intrusion-detection model," *IEEE Transactions on software engineering*, no. 2, pp. 222–232, 1987.

[5] P. I. Radoglou-Grammatikis and P. G. Sarigiannidis, "Securing the smart grid: A comprehensive compilation of intrusion detection and prevention systems," *IEEE Access*, vol. 7, pp. 46 595–46 620, 2019.

[6] A. Chowdhury, "Recent cyber security attacks and their mitigation approaches–an overview," in *Applications and Techniques in Information Security: 6th International Conference, ATIS 2016, Cairns, QLD, Australia, October 26-28, 2016, Proceedings 7*. Springer, 2016, pp. 54–65.

[7] S. Dadkhah, H. Mahdikhani, P. K. Danso, A. Zohourian, K. A. Truong, and A. A. Ghorbani, "Towards the development of a realistic multi-dimensional iot profiling dataset," in *2022 19th Annual International Conference on Privacy, Security & Trust (PST)*. IEEE, 2022, pp. 1–11.

[8] I. Makhdoom, M. Abolhasan, J. Lipman, R. P. Liu, and W. Ni, "Anatomy of threats to the internet of things," *IEEE communications surveys & tutorials*, vol. 21, no. 2, pp. 1636–1675, 2018.

[9] B. B. Zarpelão, R. S. Miani, C. T. Kawakani, and S. C. de Alvarenga, "A survey of intrusion detection in internet of things," *Journal of Network and Computer Applications*, vol. 84, pp. 25–37, 2017.

[10] K. A. Da Costa, J. P. Papa, C. O. Lisboa, R. Munoz, and V. H. C. de Albuquerque, "Internet of things: A survey on machine learning-based intrusion detection approaches," *Computer Networks*, vol. 151, pp. 147–157, 2019.

[11] E. F. Jesus, V. R. Chicarino, C. V. De Albuquerque, and A. A. d. A. Rocha, "A survey of how to use blockchain to secure internet of things and the stalker attack," *Security and communication networks*, vol. 2018, 2018.

[12] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A survey on emerging sdn and nfv security mechanisms for iot systems," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 812–837, 2018.

[13] J. Franco, A. Aris, B. Canberk, and A. S. Uluagac, "A survey of honeypots and honeynets for internet of things, industrial internet of things, and cyber-physical systems," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2351–2383, 2021.

[14] X. Liu, P. Zhu, Y. Zhang, and K. Chen, "A collaborative intrusion detection mechanism against false data injection attack in advanced metering infrastructure," *IEEE Transactions on Smart Grid*, vol. 6, no. 5, pp. 2435–2443, 2015.

[15] M. Attia, H. Sedjelmaci, S. M. Senouci, and E.-H. Aglzim, "A new intrusion detection approach against lethal attacks in the smart grid: temporal and spatial based detections," in *2015 Global Information Infrastructure and Networking Symposium (GIIS)*. Guadalajara, Mexico: IEEE, 2015, pp. 1–3.

[16] P. Radoglou-Grammatikis, P. Sarigiannidis, G. Efstathopoulos, T. Lagkas, G. Fragulis, and A. Sarigiannidis, "A self-learning approach for detecting intrusions in healthcare systems," in *ICC 2021-IEEE International Conference on Communications*. IEEE, 2021, pp. 1–6.

[17] T. Xing, Z. Xiong, D. Huang, and D. Medhi, "Sdnips: Enabling software-defined networking based intrusion prevention system in clouds," in *10th International Conference on Network and Service Management (CNSM) and Workshop*. IEEE, 2014, pp. 308–311.

[18] H. Lin, "Sdn-based in-network honeypot: Preemptively disrupt and mislead attacks in iot networks," *arXiv preprint arXiv:1905.13254*, 2019.