International Journal of Information Security (2025) 24:157
https://doi.org/10.1007/s10207-025-01076-2

REGULAR CONTRIBUTION

Defending industrial internet of things against Modbus/TCP threats: A combined AI-based detection and SDN-based mitigation solution

Thanasis Kotsiopoulos[1] · Panagiotis Radoglou-Grammatikis[1] · Zacharenia Lekka[2] · Valeri Mladenov[3] · Panagiotis Sarigiannidis[1]

Abstract
Industrial Internet of Things (IIoT) environments are ushering in new avenues for connectivity and intelligent control, yet their integration with legacy systems poses substantial security challenges. Present cybersecurity frameworks are insufficient for safeguarding protocols like Modbus/TCP, widely employed in critical infrastructures such as smart grids and healthcare. This protocol's inherent vulnerabilities-specifically, the lack of robust authentication and authorisation mechanisms-render industrial networks susceptible to a spectrum of cyberattacks with potentially cascading effects. The research motivation stems from the urgent need for an adaptive, robust security solution that bridges this gap. To address these issues, we propose an integrated approach that combines advanced threat modeling with state-of-the-art detection and mitigation techniques. First, we develop a comprehensive Modbus/TCP threat model by integrating STRIDE-per-element analysis, Attack Defence Trees (ADT), and risk assessment frameworks (CVSS and OWASP-RR) to quantitatively and qualitatively evaluate 14 distinct cyber threats. Next, we introduce a novel Intrusion Detection and Prevention System (IDPS) that leverages an Active ResNet50-based Convolutional Neural Network enhanced with Transfer Learning and Active Learning. This enables automated detection and classification of cyberattacks through continuous re-training based on human verification. Finally, our system employs a Software Defined Networking (SDN)-based mitigation strategy, using Thompson Sampling for adaptive, cost-effective decision-making. Experimental evaluation on a custom Modbus/TCP dataset demonstrates improved accuracy, higher True Positive Rates, and reduced False Positive Rates compared to conventional methods. These outcomes substantiate that integrating AI-driven detection with SDN-based mitigation offers a viable and robust framework to minimize cyberattack impacts on critical IIoT infrastructures.

Keywords Active learning · Intrusion detection and prevention · Modbus · Software-defined networking · Thompson sampling · Threat modelling · Transfer learning

1 Introduction

The technological leap of the Industrial Internet of Things (IIoT) leads the Critical Infrastructures (CIs) and, in general, the industrial environments into a new digital era with multiple benefits, such as self-monitoring, self-healing and pervasive control. In particular, the smart electrical grid will constitute the biggest IIoT application, offering advantageous services for energy consumers and utility companies

✉ Panagiotis Radoglou-Grammatikis
pradoglou@uowm.gr; pradoglou@k3y.bg

Thanasis Kotsiopoulos
akotsiopoulos@uowm.gr

Zacharenia Lekka
zlekka@k3y.bg

Valeri Mladenov
valerim@tu-sofia.bg

Panagiotis Sarigiannidis
psarigiannidis@uowm.gr

[1] Department of Electrical and Computer Engineering, University of Western Macedonia, Campus ZEP Kozani, 50100 Kozani, Greece

[2] K3Y Ltd, Studentski District, Vitosha Quarter, Bl. 9, 1700 Sofia, Bulgaria

[3] Department of Theoretical Electrical Engineering, Technical University of Sofia, 8, Kliment Ohridski Blvd., Block 12, Floors 4 and 5, Sofia, Bulgaria

Published online: 13 June 2025

[1]. Nevertheless, this evolution raises severe cybersecurity and privacy issues due to the heterogeneous nature of smart and legacy IIoT entities. In particular, the operation of legacy IIoT systems, such as Supervisory Control and Data Acquisition (SCADA)/Industrial Control Systems (ICS), rely on insecure communication protocols [2, 3]. On the other hand, the vast amount of data generated by smart IIoT devices, such as sensors and actuators, makes the security and information management of the various entities harder. A cybersecurity incident against an IIoT environment can result in disastrous consequences [4]. A characteristic example was the Advanced Persistent Threat (APT) [5, 6] against a Ukrainian substation, leading to a power outage for more than 225, 000 people [7]. Other relevant critical cases were Stuxnet, Duqu, Flame, Gaus, DragonFly, WannaCry and TRITON.

Therefore, the presence of reliable intrusion detection and mitigation mechanisms is necessary. Although significant advancements have been made in securing communication protocols in non-industrial environments [8] and in protocols other than Modbus/TCP [9, 10], industrial infrastructures remain exposed to critical cybersecurity challenges. Legacy protocols such as Modbus/TCP are still widely used in IIoT settings; however, they inherently lack robust authentication and authorization mechanisms. As a result, these protocols are particularly vulnerable to cyberattacks, which can trigger cascading failures across interconnected systems.

In [11], P. Kotzanikolaou et al. study the interdependencies among the CIs and the risk of cascading effects. Similarly, G. Mendes et al. in [12] provide a regional analysis related to the economic impact of power outages in the healthcare facilities of the US. Consequently, a Modbus/TCP threat against the components of a distribution substation related to a healthcare centre can affect the operation of the latter. Although Machine Learning (ML) and Deep Learning (DL) solutions [13, 14] have already demonstrated their efficiency for detecting intrusions, the rarely available Modbus/TCP intrusion detection datasets complicate their adoption. Based on the aforementioned remarks, in this paper, we first present a Modbus/TCP threat model, evaluating quantitatively and qualitatively 14 Modbus/TCP threats supported by the existing Modbus/TCP-related penetration testing tools. Next, we provide an Intrusion Detection and Prevention System (IDPS), which combines Transfer Learning, Active Learning, Reinforcement Learning (RL) [15, 16] and Software-Defined Networking (SDN) [17, 18] in order to detect, discriminate and mitigate the Modbus/TCP threats defined by the proposed Modbus/TCP threat model. In particular, Transfer Learning [19] and Active Learning [20] are adopted for the detection and classification process, while the Thompson Sampling (TS) RL method and SDN are utilised to mitigate the Modbus/TCP threats. Consequently, the contributions of this paper are summarised in the following key points.

- **Providing a Modbus/TCP threat model which assesses quantitatively and qualitatively the severity of the Modbus/TCP threats supported by the existing Modbus/-TCP-related penetration testing tools**: The proposed Modbus/TCP threat model combines (a) STRIDE-per-element, (b) Attack Defence Tree (ADT) and (c) one of the Common Vulnerability Scoring System (CVSS) or OWASP Risk Rating (OWASP-RR) methodology.
- **Providing an Active `ResNet50`-based Convolutional Neural Network (CNN) capable of detecting and discriminating** 14 **Modbus/TCP threats**: The proposed IDPS uses an Active `ResNet50`-based CNN, which combines Transfer Learning and Active Learning. Through Transfer Learning, the proposed IDPS takes full advantage of strong pre-trained CNNs, such as `ResNet50`. On the other side, Active Learning allows IDPS to re-train itself dynamically, thus optimising its detection performance. It is also noteworthy that in the context of the Active `ResNet50`-based CNN implementation, we created a Modbus/TCP intrusion detection dataset, which is provided publicly through this work.
- **Mitigating Modbus/TCP threats combining TS and SDN**: The proposed IDPS can mitigate the Modbus/-TCP threats recognised successfully by combining TS and SDN, taking into account the special IIoT characteristics.

The rest of this paper is organised as follows. Section 2 discusses similar works, highlighting our contribution. Section 5 is devoted to the Modbus/TCP threat model. Section 4 presents the architecture of the proposed IDPS. Section 5 focuses on the detection of the Modbus/TCP threats, analysing two detection layers. Next, section 6 details the mitigation process. Finally, section 7 is devoted to the evaluation analysis, while section 8 concludes this paper.

2 Related work

Several works investigate the security issues of IIoT. In this section, we focus on similar works regarding (a) threat modelling in IIoT, (b) intrusion detection for IIoT and (c) mitigating or even preventing cyberattacks through SDN. Finally, based on this brief literature review, we discuss how our paper is differentiated, highlighting the relevant contributions.

In [21], E. Li et al. provide a threat model combining an Attack Tree (AT) and CVSS in order to identify and evaluate the potential intrusions against a Distribution Automation System (DAS) [22]. First, the authors present the DAS architecture, discussing the components, the functional characteristics and the individual security requirements. Next,

the proposed DAS AT model is introduced, describing how it is formed and adapted appropriately with respect to the DAS architecture. Subsequently, the CVSS [23] standard is analysed with respect to the proposed DAS AT model. In particular, the authors show how the CVSS score is calculated for each leaf node of AT and how this score is propagated to the upper internal nodes. The leaf nodes correspond to particular Common Vulnerabilities and Exposures (CVE), whose CVSS score is computed by the US National Vulnerability Database (NVD). On the other side, the internal nodes represent more conceptual threats. For each case, the respective equations and algorithms are provided. Consequently, CVSS is applied to the overall AT and the most threatening path is identified. The authors compare the proposed threat model with a similar one relying on CVSS and the Bayes method. Although both models provide similar results, the proposed threat model provides higher attack probabilities.

In [24], P. Husting et al. provide a detailed attack taxonomy for the Modbus protocols. First, the authors discuss the two versions of Modbus: (a) Modbus/Serial and (b) Modbus/TCP. Next, they introduce an attack identification methodology, which is composed of three groups: (a) attacks exploiting the Modbus protocol specifications, (b) attacks taking full advantage of the vendors' implementation and (c) attacks targeting the infrastructure, which comprises various assets' types. Based on this identification, the authors pay special attention to the first category, enumerating and discussing the possible cyberattacks for Modbus/Serial and Modbus/RTU. Each Modbus cyberattack is classified into one of the following four categories: (a) Interception, b) Interruption, (c) Modification and (d) Fabrication. Moreover, the attacks' targets are identified. Finally, the impact of each attack is evaluated and discussed. The authors identified and evaluated 20 attacks for Modbus/Serial and 28 attacks for Modbus/TCP. However, it is noteworthy that most of them are theorised without providing implementation details or tools. Furthermore, many of them refer to the transport layer instead of the application layer where Modbus operates.

In [25], I. Baptista et al. present a novel malware detection system that relies on binary visualisation and Self-Organising Incremental Neural Networks (SOINN). Regarding binary visualisation, the authors adopt `Binvis` in order to transform network packet files (i.e., pcap files) into two-dimensional images. Next, the feature extraction phase takes place, identifying the images' Region of Interest (RoI). In particular, the images are divided into four parts and converted into an entire histogram, which serves as a feature vector. Finally, SOINN receives the previous feature vector and is responsible for detecting the malicious patterns. The authors investigate the efficiency of the proposed IDS against multiple malware types originating from the `VirusShare` website. In particular, it is validated against (a) Virus, (b) Worm, (c) Backdoor, (d) Trojan, (e) Rootkit and (e) other malware types that are incorporated into a variety of binary files, such as .exe, .doc, .pdf, .txt and .htm. The evaluation results demonstrate the efficacy of the proposed detection mechanism since the maximum detection accuracy equals 94.1%.

In [26], J. A. Perez-Diaz et al. present an SDN-based architecture for detecting and mitigating low-rate DDoS attacks against Hypertext Transfer Protocol (HTTP). The proposed architecture consists of two main components: (a) IPS and (b) IDS. On the one hand, IPS is responsible for gathering the network flows and mitigating them based on the detection outcome of IDS. In particular, IPS is composed of three modules: (a) Flow Management Module, (b) Suspicious Attackers Management and (c) Mitigation Management Module. The Flow Management Module gathers the HTTP flows from the SDN switches. These flows will be further processed to detect a potential low-rate Distributed DoS (DDoS) attack. HTTP flow statistics are generated through Flowtbag and transmitted to IDS. The Suspicious Attackers Management module handles a blacklist of potential cyberattackers. Finally, the Mitigation Management module follows a mitigation strategy and generates appropriate rules to mitigate the malicious flows. These rules are transmitted to SDN-C. In this work, the `Open Network Operating System (ONOS)` is utilised as SDN-C [27]. On the other hand, IDS comprises three modules: (a) Identification API, (b) ML Model Selection and (c) Identification. The Identification API manages the communication with the Flow Management Module of IPS. The ML Model Selection Module represents a set of pre-trained ML models. Finally, the Identification module selects one of the pre-trained ML models to analyse the HTTP flow each time. To evaluate their work, the authors use `Mininet`, `SlowHTTPTest` and the 2017 CIC DoS dataset. The experimental results confirm the efficiency of the proposed method.

Although the previous works provide valuable insights and methodologies, they are characterised by remarkable limitations. First, P. Husting et al. in [24] provide a comprehensive survey about the Modbus/Serial and Modbus/TCP threats. However, they do not estimate quantitatively the severity of them. Moreover, they conduct a theoretical analysis without taking into account the Modbus-related cyberattacks supported by relevant penetration testing tools. Regarding the detection of the Modbus/TCP threats, none of the existing works can discriminate efficiently the type of the various Modbus/TCP threats supported by the Modbus-related penetration testing tools. Finally, it is noteworthy that the current solutions cannot mitigate these cyberattacks in an efficient manner, taking into account the sensitive IIoT characteristics. In this paper, we present an entire solution, which solves the limitations mentioned above. First, we provide a Modbus/TCP threat model, which assesses quantitatively and qualitatively the severity of 14 Modbus/TCP threats that

can be executed by existing Modbus/TCP-related penetration testing tools. In addition, we present an IDPS that can detect, classify and mitigate 14 Modbus/TCP cyberattacks defined by the previous threat model. The Table 1 provides a summary regarding the main highlights and the opportunities for future work regarding each paper.

3 Modbus/TCP threat modelling

The proposed Modbus/TCP threat model combines three methodologies: (a) STRIDE-per-element [28], (b) ADT [29] and (c) one of CVSS and OWASP-RR. The main goal is to identify the various Modbus/TCP cyberattacks [30] and prioritise their severity, considering their probability and impact as isolated and combined cases against the essential cybersecurity principles: Confidentiality, Integrity and Availability (CIA). First, STRIDE-per-element is adopted in order to define the primary cyberattack super-classes reflecting the target behind the various Modbus/TCP cyberattacks. Next, ADT is used to map and combine those Modbus/TCP cyberattacks with the STRIDE-per-element super-classes. Subsequently, both CVSS and OWASP-RR are utilised for calculating the severity for each Modbus/TCP threat. Finally, the logical relationships among the nodes of the ADT are used to estimate the severity of the STRIDE-per-element classes. Consequently, the proposed Modbus/TCP threat model combines the benefits of each methodology, thus determining the severity of the individual Modbus/TCP threats and their super-class. If we use only ADT, we could not provide an adequate quantitative analysis. On the other hand, if we adopt only CVSS or OWASP-RR, we could not calculate the severity of the different super-classes targeting CIA. Finally, the STRIDE-per-element allows us to distinguish the appropriate super-classes related to CIA and the individual Modbus/TCP threats. The following paragraphs provide a brief description for each of the aforementioned methodologies, while subsequently, the Modbus/TCP-related ADT and the respective CVSS and OWASP-RR scores are analysed.

First, STRIDE is an acronym that stands for Spoofing, Tampering, Repudiation, Information Disclosure, DoS, and Elevation of Privilege. In the context of this paper, the variant called STRIDE-per-element [31] is used to identify the Modbus/TCP threats supported by existing penetration testing tools. In particular, five penetration testing tools are investigated: `Smod`, `Metasploit`, `Nmap`, `mbtget`, `ModScan`. Hence, 14 cyberattacks are identified. These cyberattacks are considered as Data Flow elements. Thus, from the initial STRIDE attack families, only three families are taken into account: (a) Tampering, (b) Information Disclosure and (c) DoS.

Subsequently, ADT is used to structure and visualise the Modbus/TCP threats. In particular, an ADT consists of two opponent nodes: (a) attacking nodes and (b) defending nodes. The first category expresses the goal and the malicious activities that a cyberattacker may perform to violate the security of the target system. On the other side, the defending nodes indicate the countermeasures that the defender can adopt in order to mitigate or even prevent the cyberattacks. Each node can be expanded with one or more children of the same type, thus defining refinements that indicate sub-goals and actions. In addition, each node can have children of the opposite type, denoting threats or countermeasures, respectively. The refined nodes can be divided into two types (a) conjunctive and (b) disjunctive. In the first case, a conjunctively refined node carries out its goal, whether all of its children necessarily accomplish their goals. In contrast, the goal of a disjunctively refined node is achieved if at least one of its children carries out its goal. Therefore, the conjunctive and disjunctive refinements are represented by the AND and OR logical operators, respectively.

Finally, CVSS and the OWASP-RR are used to evaluate the severity of each Modbus/TCP threat quantitatively and qualitatively. Both of them operate independently and rely on different methodologies. In particular, CVSS is an open vulnerability assessment framework which quantifies the severity of each vulnerability or attack between 0 and 10. CVSS consists of three metric groups, namely (a) Base Group, (b) Temporal Group and (c) Environmental Group. The Base Group reflects the intrinsic features of the vulnerability/attack. These features cannot be affected over time or modified by compensating factors. The Temporal Group focuses on vulnerabilities/attacks that evolve or change over time, evaluating their exploitability as well as the availability of the respective security controls. Finally, the Environmental Group enables an organisation to adjust appropriately the values of the Base Group, taking into account its own security requirements. On the other side, the calculation of the OWASP-RR score is calculated by Equation 1. Both Likelihood and Impact depend on additional factors. In particular, Likelihood expresses the possibility of occurrence of each identified threat, and it is computed by averaging the values of the Threat Agent Factor and the Vulnerability Factor. The Threat Factor is calculated by summing the values of four factors: (a) Skill Level, (b) Motive, (c) Opportunity and (d) Size. Similarly, the Vulnerability Factor is computed by adding four factors: (a) Ease of Discovery, (b) Ease of Exploit, (c) Awareness and (d) Intrusion Detection. Accordingly, Impact represents the consequences if that threat eventuates, and it is determined by averaging the values of the Technical Impact Factor and the Business Impact Factor. In a similar manner, the Threat Impact Factor is calculated by summing the values of four factors: (a) Loss of Confidentiality, (b) Loss of Integrity, (c) Loss of Availability and (d) Loss of Accountability. On the other hand, the Business Impact Factor is also calculated by summing the values of four factors: (a) Finan-

Table 1 Review Summary of Related Works

Paper	Strengths / Highlights	Opportunities for Future Work
Li et al. [21]	Integrates Attack Tree methodology with CVSS to systematically identify and evaluate potential threats.	May benefit from real-time adaptation and integration with modern mitigation strategies in IIoT settings. In addition, regarding the detection of the Modbus/TCP threats, they cannot discriminate efficiently the type of the various Modbus/TCP threats supported by the Modbus-related penetration testing tools
Husting et al. [24]	Offers an extensive taxonomy for Modbus-related attacks, aiding in a thorough understanding of vulnerabilities.	The authors do not estimate quantitatively the severity of the security incidents. Moreover, they conduct a theoretical analysis without taking into account the Modbus-related cyberattacks supported by relevant penetration testing tools. Regarding the detection of the Modbus/TCP threats, they cannot discriminate efficiently the type of the various Modbus/TCP threats supported by the Modbus-related penetration testing tools. Last but not least, they cannot mitigate these cyberattacks in an efficient manner, taking into account the sensitive IIoT characteristics.
Baptista et al. [25]	Presents an innovative approach for malware detection using binary visualization and SOINN. The method effectively transforms network packet data into informative visual representations, enabling the identification of malicious patterns.	Regarding the detection of the Modbus/TCP threats, they cannot discriminate efficiently the type of the various Modbus/TCP threats supported by the Modbus-related penetration testing tools. Also, they cannot mitigate these cyberattacks in an efficient manner, taking into account the sensitive IIoT characteristics. Additionally, integrating adaptive or hybrid learning techniques may further improve detection accuracy and system scalability.
Perez-Diaz et al. [26]	Proposes a robust SDN-based architecture for the detection and mitigation of low-rate DDoS attacks. Their design successfully integrates flow management with real-time intrusion detection and mitigation, showcasing promising operational efficiency in managing HTTP traffic.	Current detection methods for Modbus/TCP threats lack the capability to effectively discriminate between different types of attacks generated by Modbus-specific penetration testing tools. Furthermore, these approaches fail to provide efficient mitigation strategies, particularly when considering the unique and sensitive operational requirements of Industrial Internet of Things (IIoT) environments. Further refinement of the integration between the IPS and IDS modules may also improve the system's overall resilience and real-world applicability.

cial Damage, (b) Reputation Damage, (c) Non-compliance and (d) Privacy Violation. The values of the aforementioned factors range between $0-9$.

$$OWASP-RR_{Risk} = Likelihood \times Impact \quad (1)$$

Fig. 1 illustrates the proposed ADT. The STRIDE elements represent the refined nodes, while 14 cyberattacks supported by the aforementioned Modbus/TCP-related penetration testing tools denote the non-refined nodes. Therefore, `Tampering` is related to the integrity principle and is composed of two disjunctive refinements: (a) `modbus/function/writeSingleCoils` and (b) `modbus/function/writeSingleRegister`. Similarly, `DoS` refers to the availability requirement and consists of six disjunctive refinements: (a) `modbus/dos/writeSingleCoils`, (b) `modbus/dos/writeSingleRegister`, (c) `modbus/function/readCoils (DoS)`, (d) `modbus/function/readCoils (DoS)`, (e) `modbus/function/readInputRegister (DoS)` and (f) `modbus/function/readDiscreteInput (DoS)`. Finally, `Information Disclosure` corresponds to the confidentiality principle and comprises six disjunctive refinements, namely (a) `modbus/function/readCoils`, (b) `modbus/scanner/getfunc`, (c) `modbus/scanner/uid`, (d) `modbus/function/readInputRegister`, (e) `modbus/function/readHoldingRegister` and (f) `modbus/function/readDiscreteInput`. The aforementioned cyberattacks take full advantage of the fact that Modbus/TCP does not include any authentication and authorisation mechanism, thus allowing a cyberattacker to use the Modbus/TCP function codes for malicious purposes. The names of the non-refined nodes originate from the corresponding modules of the aforementioned penetration testing tools. For each non-refined node, CVSS and OWASP-RR are applied individually, calculating the corresponding severity scores.

Next, these scores are propagated to the upper nodes based on Equation 2 and Equation 3. In particular, Equation 2 is

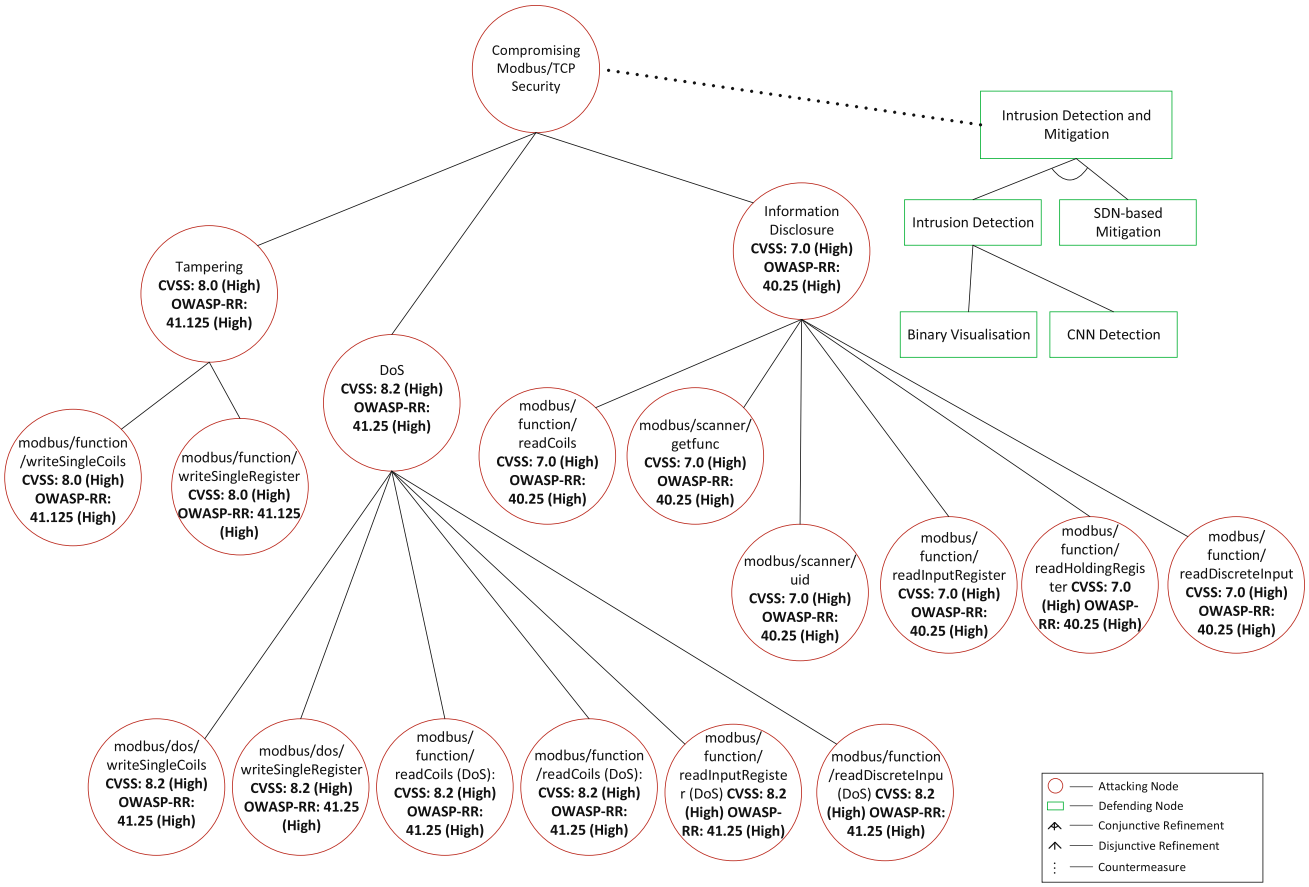

Fig. 1 Modbus/TCP Threat Model

applied when the refined node comprises conjunctive refinements since the parent's goal is achieved whether all children accomplish their goal. Therefore, the severity score of a conjunctively refined node is equal to the product of the childrens' severity scores. The product indicates the probability behind the severity score of each child.

In contrast, Equation 3 is utilised when the refined node includes disjunctive refinements since the respective goal is achieved whether a child will accomplish its goal. Consequently, the severity score of the disjunctively refined node equates with the maximum severity score of the various children. Based on these computations, both CVSS and OWASP-RR estimate the severity of each Modbus/TCP threat as "high". Fig. 1 presents the quantitative scores.

Finally, the proposed ADT includes a countermeasure called Intrusion Detection and Mitigation. This countermeasure comprises two conjunctive refinements: (a) Intrusion Detection and (b) SDN-based mitigation. The first one is responsible for the timely detection of the Modbus/TCP threats and includes two disjunctive refinements: (a) Binary Visualisation and (b) CNN detection. More details about them are given in section 5. On the other side, SDN-based mitigation refers to the mitigation of the Modbus/TCP threats, taking full advantage of the SDN technology. Section 6 provides more information on this aspect.

$$CVSS(\text{or OWASP-RR})_{RefinedNode} = \prod_{i=1}^{n} CVSS(\text{or OWASP-RR})_{Refinement_i} \tag{2}$$

$$\begin{aligned} CVSS(\text{or OWASP-RR})_{RefinedNode} = \max\{ & \\ & (CVSS(\text{or OWASP-RR})_{Refinement_1}) \\ & , (CVSS(\text{or OWASP-RR})_{Refinement_2}) \\ & , ..., (CVSS(\text{or OWASP-RR})_{Refinement_n})\} \end{aligned} \tag{3}$$

4 Architectural model and implementation details

Based on the SDN architectural design, Fig. 2 illustrates the architecture of the proposed IDPS. The goal behind the proposed IDPS is to detect and mitigate timely the Modbus/TCP threats discussed in the previous section, taking full advantage of the SDN technology. In particular, the Active

ResNet50-based CNN is utilised for the detection process, while SDN plays the role of a mitigation mechanism that can drop or re-arrange the malicious Modbus/TCP network flows. In this paper, we focus only on the first case (i.e., dropping malicious Modbus/TCP network flows). In contrast to typical IPS and traditional firewall systems, SDN represents a more reliable mitigation mechanism with respect to a massive amount of alerts. In our case, instead of corrupting the malicious Modbus/TCP network flows directly like other works, we consider whether this action (i.e., dropping malicious Modbus/TCP network flows) could generate more destructive effects, taking into account the sensitive nature of an IIoT environment.

The SDN architectural model consists of three main planes: (a) data plane, (b) control plane and (c) application plane. The data plane includes the physical and virtual assets connected to the SDN switches. These assets are called Network Elements (NE) and, in our case, represent IIoT devices, such as sensors, actuators, Programmable Logic Controllers (PLCs) and Remote Terminal Units (RTUs). The SDN switches can be physical or virtual. We adopt the Open vSwitch (OVS). Next, the control plane is characterised by the presence of one or more SDN Controllers (SDN-C) responsible for configuring the SDN switches and orchestrating the overall SDN network. In this paper, we use the Ryu controller [32]. SDN-C communicates with SDN switches through a South-Bound Interface (SBI). To this end, various SBI protocols have been implemented. We utilise OpenFlow v.1.3. Finally, the Application Plane includes SDN applications that instruct SDN-C to execute specific network policies. The communication between SDN-C and SDN applications is conducted through a North-Bound Interface (NBI). Similarly to SBI, various NBI protocols have been designed, such as Representational State Transfer (REST). In the context of the proposed IDPS, we use the Ryu REST Application Programming Interface (API).

The architecture of the proposed IDPS consists of five modules: (a) Network Traffic Monitoring and Capturing Module (NTMCM), (b) Network Flow Extraction and Clustering Module (NFECM), (c) Visual Representation Generation Module (VRGM), (d) Intrusion Detection Engine (IDE) and (e) Notification and Mitigation Module (NMM). The first module is responsible for monitoring and capturing the entire Modbus/TCP network traffic. To this end, Switched Port Analyser (SPAN) and Tcpdump are utilised. NFECM receives the overall Modbus/TCP network traffic as an overall pcap file and discriminates the bidirectional Modbus/TCP network flows, generating the corresponding pcap files.

A network flow is characterised by four elements: (a) source Internet protocol (IP) address, (b) destination IP address, (c) source TCP/User Datagram Protocol (UDP) port and (d) destination TCP/UDP port. Thus, each pcap file generated by NFECM includes the Modbus/TCP packets of a specific Modbus/TCP network flow. For this purpose, the PcapPlusPlus-PcapSplitter is used. Next, VRGM uses Binvis in order to convert each pcap file related to the Modbus/TCP network flows into visual representations. More details about this conversion are provided in subsection 5.1. Subsequently, IDE adopts an Active ResNet50-based CNN, which receives the visual representations and classifies them into the aforementioned Modbus/TCP threats. Accordingly, more insights about the operation of the proposed Active ResNet50-based CNN is given in subsection 5.2. Finally, NMM informs the security administrator about the security events and applies TS in order to mitigate them. The mitigation process is further analysed in subsection 6.

If NMM takes a decision to drop automatically the malicious network flows related to the aforementioned Modbus/TCP cyberattacks, then NMM does not use OpenFlow directly, but, it takes full advantage of the Ryu REST API in order to guide Ryu on how to insert the appropriate rules to the flow tables of OVS. In particular, two rules are added; thus, two REST requests are sent by NMM to Ryu.

Next, the appropriate OpenFlow commands are transmitted automatically by Ryu in order to insert the new rules to the OVS flow tables. The REST requests include the following fields: table_id, actions, hard_timeout, idle_timeout, priority, dpid and match. The last field includes also seven extra sub-fields: in_port, eth_type, ip_proto, ipv4_src, tcp_src, ipv4_dst and tcp_dst.

First, table_id expresses the identifier of the table where the new rules will be inserted. actions defines a set of instructions such as for example to allow, drop or forward the Modbus/TCP packets specified by the rule. hard_timeout denotes the maximum time before discarding. idle_timeout implies the idle time prior to discarding. Next, priority defines the priority of this rule, while dpid denotes the identifier of the corresponding SDN switch (i.e., OVS). Finally, match defines the criteria utilised for identifying the Modbus/TCP packets that will be managed by this rule. in_port indicates the input port of OVS. eth_type determines the Ethernet frame type according to Internet Assigned Numbers Authority (IANA). ip_proto defines the protocol attribute of Internet Protocol version 4 (IPv4 or IP) based on IANA. ipv4_src, ipv4_dst, tcp_src and tcp_dst are used to identify the network flows controlled by this rule.

In particular, the first two attributes define the source IP address and the source TCP/UDP port, while the latest ones specify the destination IP address and the destination TCP/UDP port, respectively. The first REST request uses the ipv4_src and the tcp_src, while the second uses the ipv4_dst and the tcp_dst. Both ipv4_src and ipv4_dst refer to the same IP address. Similarly,

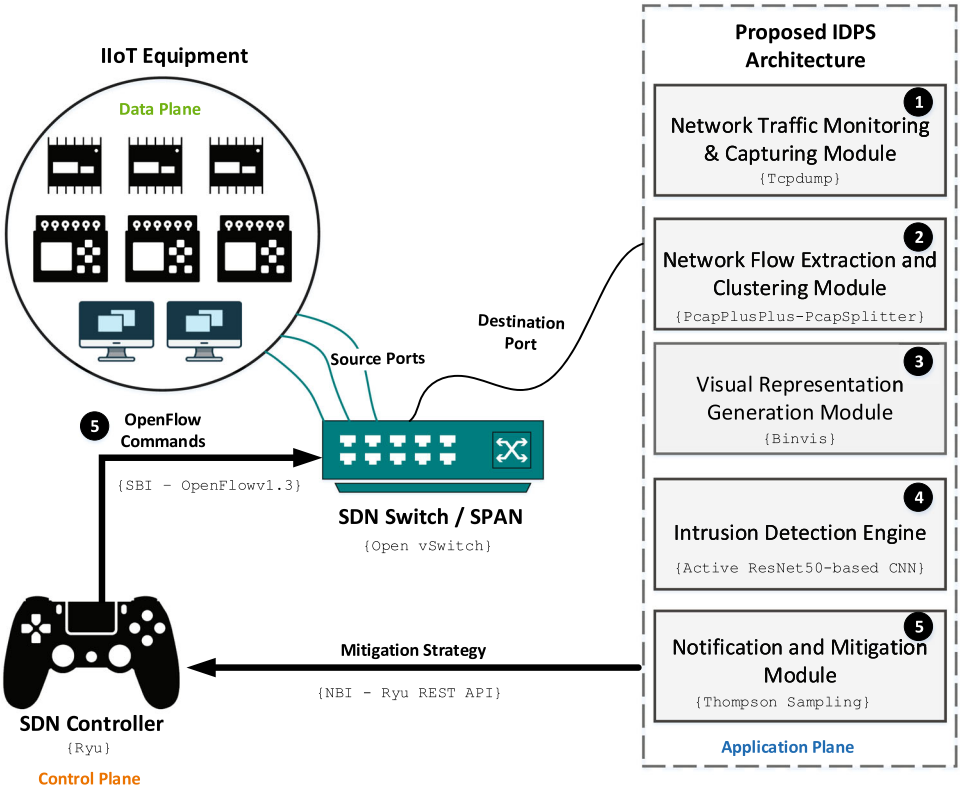

Fig. 2 Architecture of the proposed IDPS

`tcp_src` and `tcp_src` are assigned to 502 which is the default TCP port for the Modbus/TCP protocol. Finally, with respect to the installation of the proposed IDPS two different Virtual Machines (VMs) are utilised. The first VM is used by `Ryu`, while the second VM is used by the proposed IDPS.

5 Modbus/TCP threat detection

The IDE combines two detection layers that work in a complementary manner. The first layer constitutes a binary visualisation mechanism that supports the security administrator to distinguish manually the Modbus/TCP threats. On the other side, the second layer applies an Active `ResNet50`-based CNN in order to classify the Modbus/TCP network flows automatically. Both layers work together for the accurate detection of the Modbus/TCP threats. In particular, the first layer constitutes a verification method through which the security administrator can oversee the detection results of the second layer. Moreover, it is noteworthy that the first layer contributes to the re-training process of the Active `ResNet50`-based CNN. The following subsections provide more details for each detection layer, respectively.

5.1 Binary visualisation

The proposed IDPS adopts `Binvis` [33] in order to transform the pcap files reflecting the corresponding Modbus/TCP network flows into understandable visual representations (i.e., images) utilised by the security administrator to discriminate the aforementioned Modbus/TCP threats. `Binvis` relies on the Python library `scurve`, which transforms binary files into various curve representations. In particular, each byte of the pcap files is translated into a pixel, utilising the following colour scheme of `scurve`: (a) `Black: 00`, (b) `White: FF`, (c) `Blue: printable characters` and `(d) Red: everything else`. Thus, each pixel is placed on the two-dimensional visual representation, taking into account the locality of the binary elements.

The binary elements being close in the pcap files should be placed as near as possible on the two-dimensional representation. To this end, `Hilbert Curve` is used to arrange the pixels in the image. The `Hilbert Curve` belongs to the family of the recursive Space-Filling Curves (SFCs) that divide a space into several segments, visiting the segments with a particular order. SFCs, also known as Peano curves, project the data from one-dimensional space into an n-dimensional space by preserving the properties of the original data. The range of SFC covers the two-dimensional unit square and, in general, an n-dimensional unit hypercube; however, in this paper, we focus on the two-dimensional

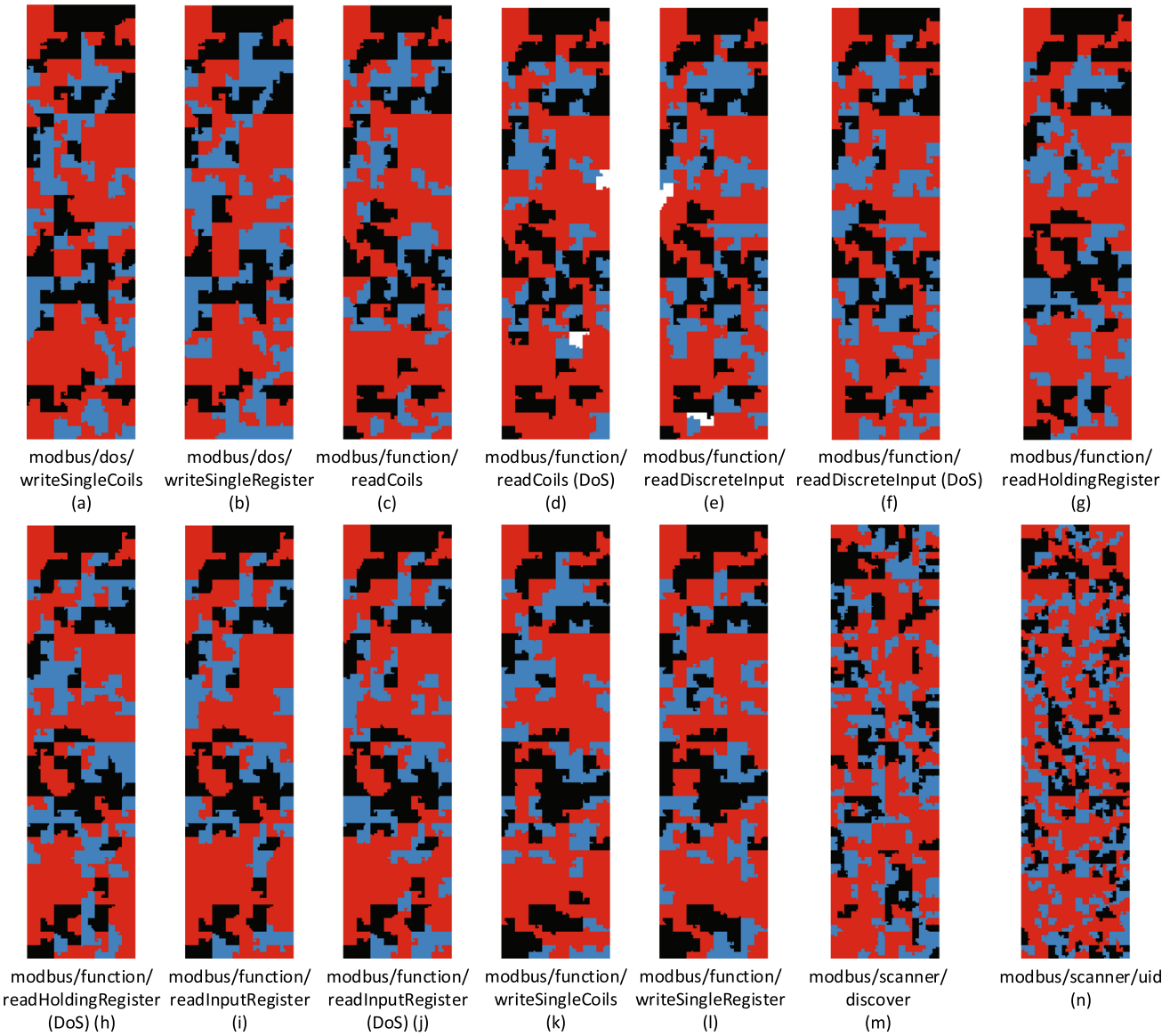

Fig. 3 Visual representation of the pcap files corresponding to the malicious network flows of the Modbus/TCP threats

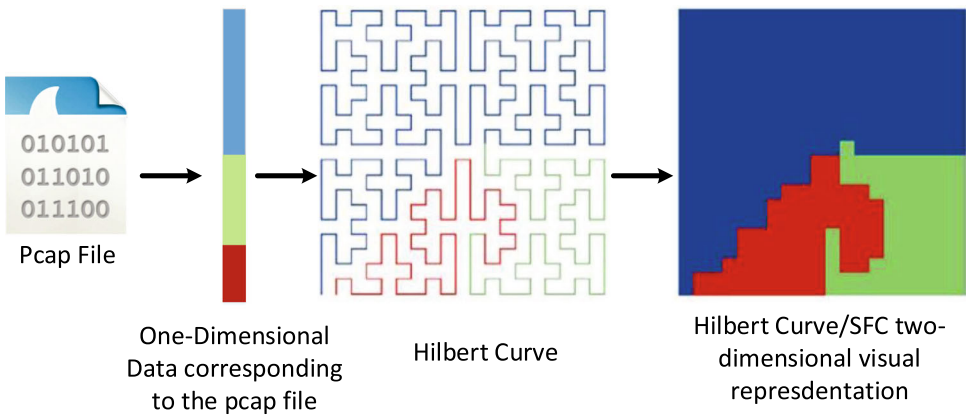

Fig. 4 Transformation of a binary pcap file into a Hilbert curve two-dimensional visual representation

space since the output of `Binvis` is a two-dimensional visual representation. Thus, a two-dimensional unit square refers to a visual representation of $n \times n$ pixels, and the Hilbert curve represents a continuous curve for each unit square (i.e., pixel of the image).

Although G. Peano was the first who defined and discovered the first SFC, D. Hilbert was the one who identified a geometrical process that allows the generation of an entire class of SFCs. D. Hilbert defined that each t belonging to an interval $I = [0, 1]$ is determined by a sequence of nested closed intervals that are generated by a successive partitioning. This sequence corresponds to a sequence of nested closed squares whose diagonals shrink into a point, determining a unique point in $Q = [0, 1]^2$ which is the image $f_h(t)$ of t. $f_{h*}(I)$ is called Hilbert Curve.

Fig 4 depicts how the Hilbert curve is utilised for transforming one-dimensional data (i.e., pcap binary file) into a two-dimensional visual representation. First, each byte of the binary pcap file is transformed into a particular colour based on the colour scheme of `scurve`. Then, the Hilbert curve is applied in order to map the one-dimensional data into a two-dimensional visual representation. Similarly, Fig 3 shows the `Binvis` visualisations for each pcap file corresponding to the malicious network flows of the aforementioned Modbus/TCP threats. Although the `Binvis` visualisations are similar to each other, a granular inspection can distinguish the differences, thus identifying the Modbus/TCP threats discussed in section 3.

5.2 Active ResNet50-based CNN detection

Although the first detection layer provides an adequate manner for discriminating the Modbus/TCP threats, it constitutes a manual solution, not applicable for a large number of Modbus/TCP network flows. The binary visualisation can be utilised only as an additional detection mechanism verifying or correcting the outcomes of automatic means. The second layer of the proposed IDPS adopts a CNN, which combines Transfer Learning [34] and Active Learning [35] in order to classify the pcap visual representations of the Modbus/TCP network flows into the Modbus/TCP threats automatically.

Both Transfer Learning and Active Learning are adopted when there are not available datasets or a sufficient amount of data, as in our case, since IIoT environments like CIs cannot disclose and share their sensitive data. On the one side, Transfer Learning refers to when an ML/DL model pre-trained for another task is used to solve a problem from another domain. This approach is applied widely to the CNN models. In particular, the new CNN uses some weights of a pre-trained CNN, which has been trained on a large-scale dataset like `ImageNet`. Usually, from the pre-trained CNN, the final fully-connected layers are removed. Next, a concise training process follows to adjust the remaining parts of

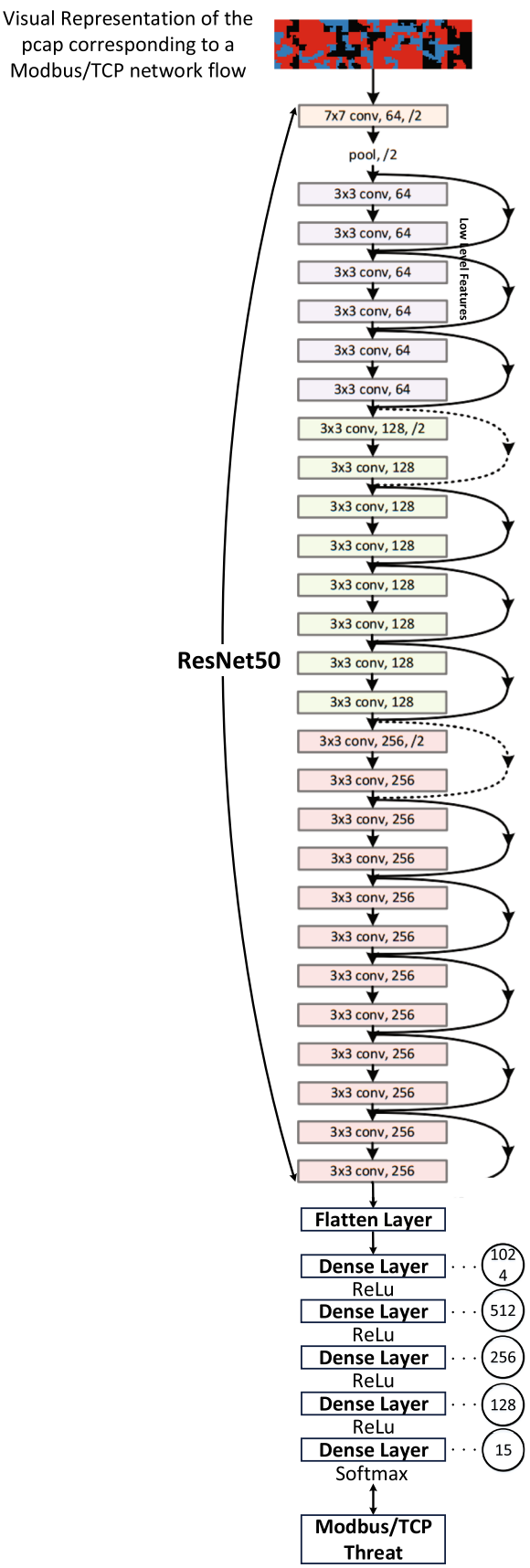

Fig. 5 Active ResNet-based CNN architecture

the new CNN corresponding to the fully connected layers. Multiple pre-trained CNNs have already demonstrated their efficiency, using the ImageNet dataset, which involves 1.2 million images. Characteristic examples are `VGG16`, `VGG19`, `ResNet50`, `Xception`, `MobileNet`, `DenseNet121` and `EfficientNetB0`. Based on a comparative analysis described in section 7, the proposed IDPS uses `ResNet50`.

More specifically, Fig. 5 shows the CNN architecture behind the second detection layer of the proposed IDPS. First, `ResNet50` is utilised, and then a sequence of a `Flatten` layer and 5 `Dense` layers follow with 1024, 512, 256, 128 and 15 neurons, respectively. Apart from the last `Dense` layer, the remaining ones use the `ReLu` activation function given by Equation 4. The last `Dense` layer uses the `Softmax` function, given by Equation 5. `ResNet50` is inspired by `VGG19`, utilising 34-layer plain network architecture in which shortcut connections are added, thus leading to the residual network illustrated by Fig. 5. The colour scheme denotes the number of the filters with respect to the convolutional layers. The training process uses the `Categorical Cross-Entropy` function (Equation 6) and the `Adam` optimiser.

$$f(x) = \begin{cases} 0, & \text{for } x \leq 0 \\ x, & \text{for } x \geq 0 \end{cases} \tag{4}$$

$$softmax(z)_i = \frac{e^{z_i}}{\sum_{j,n} e^{z_j}} \tag{5}$$

$$L_{cc}(r, p) = -\sum_{j=0}^{M}\sum_{i=0}^{N}(r_{ij} \times log(p_{ij})) \tag{6}$$

Although the `ResNet50`-based CNN constitutes an initial and efficient model for detecting and classifying the Modbus/TCP threats, its performance relies on the available training data (i.e., pcap files reflecting malicious Modbus/TCP threats.) However, such data is rarely available. Even if there are some synthesised datasets, the Modbus/TCP threats and their consequences can differ from one IIoT environment to another IIoT environment.

Therefore, the proposed IDPS adopts an Active Learning approach, which makes IDE capable of re-training itself. Active Learning composes a functional framework, which allows the selection of the most informative data samples from an unlabelled dataset, thus creating or enhancing the training dataset, leading, in our case, to a more accurate multiclass classification model. In Active Learning, the classifier is called $Hypothesis$. Unlike Passive Learning, which selects the data samples randomly, Active Learning follows particular criteria, leading to represented and representative data samples providing more accurate results. Usually, an external factor called $Oracle$ assesses and annotates the data samples selected by the Active Learning methods.

In our case, IDE and particularly the `ResNet50` CNN represents the $Hypothesis$, while the system administrator plays the role of the $Oracle$, utilising the Binvis representations. Fig. 6 illustrates the Active Learning procedure behind the proposed IDPS. In the first step, the pooling-based sampling method is adopted in order to create a pool with the unlabelled data. Next, a query strategy is used to decide which data samples from the pool will be labelled by $Oracle$ and added to the new training dataset. With respect to the query strategy, we utilise Uncertainty Sampling, which relies on the uncertainty of the $Hypothesis$.

In other words, the Uncertainty Sampling selects those binary representations for which the Active `ResNet50`-based CNN is less confident. Subsequently, the $Hypothesis$ is fed with the unlabelled data selected in the previous step. Next, the $Hypothesis$ predicts the labels of this data. The prediction outcome of the `ResNet50`-based CNN can be assessed by the security administrator based on the binary visualisation of the first detection layer. Suppose the security administrator agrees with the decision of the `ResNet50`-based CNN. In that case, this data sample (i.e., the visual representation corresponding to the pcap file of the malicious Modbus/TCP network flow) is added to the new training dataset. Otherwise, $Oracle$ will correct the decision of $Hypothesis$, and the data sample is added to the new training dataset. Finally, the new training dataset is used to re-train the `ResNet50`-based CNN, thus converting it into an Active `ResNet50`-based CNN.

Suppose the visual representations corresponding to the Mobuds/TCP network flows from an IIoT environment are generated continuously. Let x be an unlabelled visual representation from the input space X and y the respective label related to the Modbus/TCP threats discussed earlier, also comprising the normal state. Furthermore, U denotes a set of unlabelled visual representations within the pool, while L indicates the new training dataset, which will be used to re-train IDE. Therefore, on the one hand, the function $f(x) = y$ is the target function that discriminates and classifies the visual representations accurately without any functional error. On the other hand, the function $h(x) = y'$ represents the Active `ResNet50`-based CNN predicting the label of the visual representation. Consequently, the goal is to minimise the generalisation error defined by Equation 8. More precisely, the squared error loss function quantifies the deviation between the predicted output of the Active ResNet50-based CNN and the ground truth label for a given network flow. In the context of the proposed mechanism, this loss function plays a main role in both the initial training and the active learning process. It allows the model to iteratively adjust its parameters to minimise prediction errors and refine its ability to distinguish between 14 distinct Modbus/TCP threats. In addition, during the active learning process, the squared error function helps identify samples with high

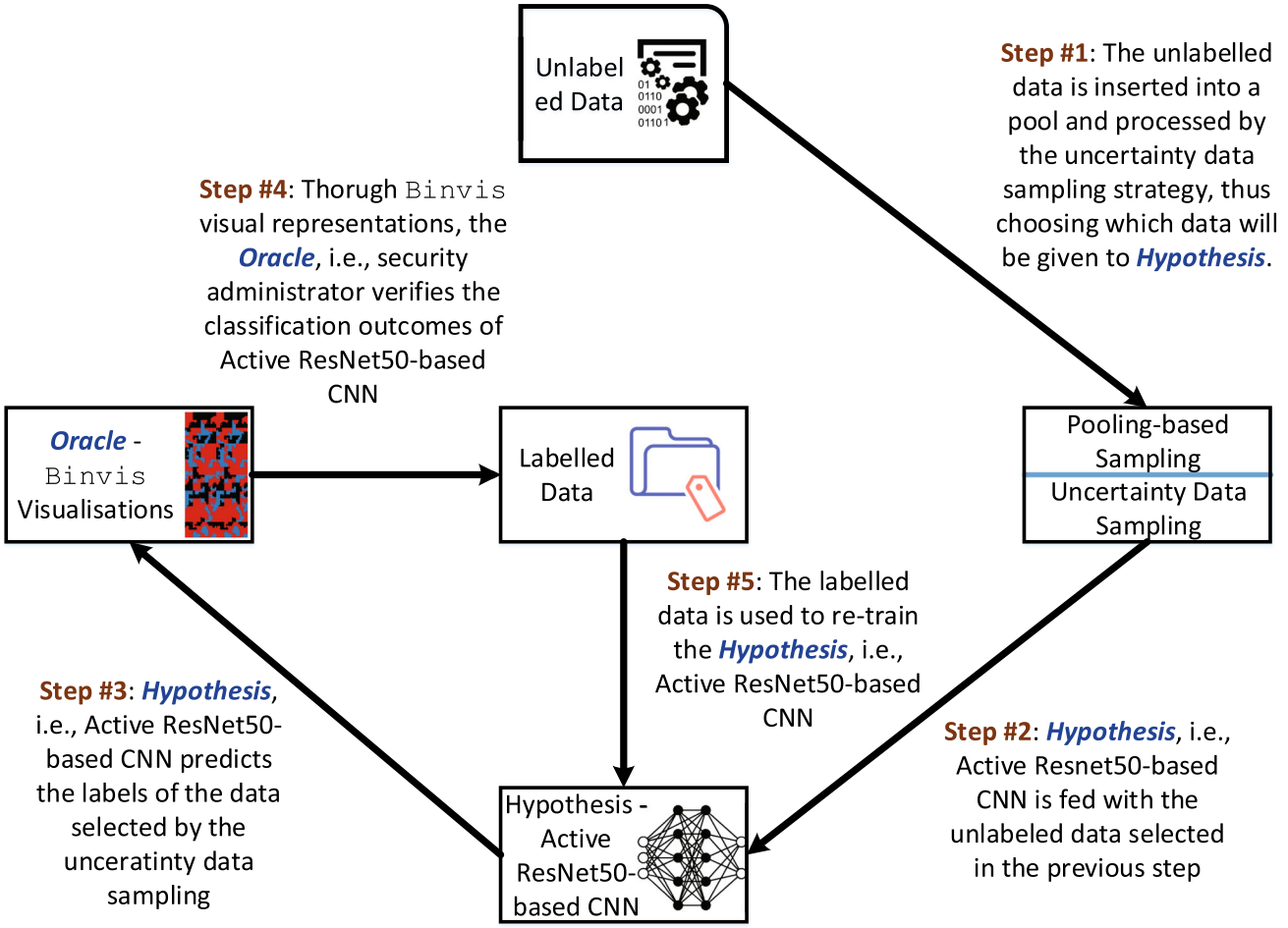

Fig. 6 Proposed Active Learning Procedure

prediction uncertainty-those with large deviations from the expected output-which are then reviewed and annotated by a human oracle. This facilitates targeted re-training, ensuring that the model evolves efficiently by focusing on informative and hard-to-classify examples. The squared error's convexity and differentiability make it ideal for gradient-based optimisation, while its sensitivity to larger errors ensures that critical misclassifications in sensitive IIoT environments are strongly penalised.

$$\begin{aligned}\mathcal{E}(h) &= \mathbb{E}_{x\sim\mathcal{D}}\left[\ell\left(h(x), f(x)\right)\right] \\ &= \int_{\mathcal{X}} \ell\left(h(x), f(x)\right)\ p(x)\,dx \\ &= \int_{\mathcal{X}} \left(h(x) - f(x)\right)^2\ p(x)\,dx\end{aligned} \tag{7}$$

where:

- $\mathcal{E}(h)$ is the expected generalization error of the hypothesis h.
- $x \in \mathcal{X}$ is an input sample from the input space $\mathcal{X}$.
- $h(x)$ is the predicted output (label) from the hypothesis function h.
- $f(x)$ is the true label from the target function f (ground truth or oracle).
- $\ell(h(x), f(x))$ is the pointwise loss function (see Equation 8).
- $p(x)$ is the probability density function over the input space $\mathcal{X}$.
- $\mathcal{D}$ is the underlying data distribution.

$$l(h(x), f(x)) = (h(x) - f(x))^2 \tag{8}$$

where:

- $h(x) \in \mathbb{R}$ is the predicted value (hypothesis output),
- $f(x) \in \mathbb{R}$ is the ground truth (oracle or target function),
- $\ell(h(x), f(x)) \in \mathbb{R}_{\geq 0}$ is the loss for a single input x.

Therefore, the Active Learning problem lies in labelling correctly and selecting the appropriate visual representations from U, thus composing and enhancing a new training dataset L that will re-train the Active `ResNet50` CNN ($Hypothesis$) and will optimise its detection efficiency. The labelling process is conducted by the $Hypothesis$ itself and is validated by the security administrator through the binary visualisation. To identify the suitable visual representations in U, Uncertainty Sampling is used. The $Hypothesis$' uncertainty can be calculated with various criteria: (a) entropy, (b) least confidence of prediction and (c) least margin. In this work, we use entropy defined by Equation 9.

$$H = -\sum_{i=1}^{m} p_\theta(y_i|x)\log_2(p_\theta(y_i|x)) \tag{9}$$

where p_θ denotes the probability of class i for the visual representation x, while θ implies the parameters of the $Hypothesis$. Therefore, the entropy criterion chooses the visual representations x^* from U that fulfil the Equation 10. In this paper, δ is determined experimentally.

$$x^* = argmax(x) + H > \delta \tag{10}$$

Based on the above remarks, Algorithm 1 illustrates the Active Learning process of the Active `ResNet50`-based CNN. First, the $Hypothesis$ $h(X)$ is trained with an initial dataset L comprising a few data samples. To this end, a Modbus/TCP intrusion detection dataset was constructed by emulating the aforementioned Modbus/TCP threats. Next, U is filled in continuously with new visual representations. While the size of U is greater than 0, $h(x)$ classifies each visual representation within U. The security administrator verifies this process through the visual representations. As depicted in Fig. 3, although the visual representations of the Modbus/TCP threats present common characteristics, they constitute an adequate manner for discriminating the Modbus/TCP threats manually. Next, the uncertainty of $h(x)$ is calculated. If the entropy criterion is satisfied, then the corresponding visual representation of U is moved in L. Next, when the size of L reaches a new threshold t, the re-training process is applied.

Algorithm 1: Active ResNet50-based CNN: Pooling-based Sampling and Uncertainty Sampling Strategy

```
Data: U, L, h
Result: Retrain h
Train h;
while size(U) > 0 do
    if uncertainty(h(U(i))) > δ then
        h predicts y(i);
        The security administrator verifies the prediction of h;
        Add U(i) and y(i) to L;
        Retrain h;
    end
    if size(L) == t then
        Retrain h;
        Clear U;
    end
end
```

6 SDN-based mitigation: A reinforcement learning approach

After detecting the Modbus/TCP threats, the mitigation phase follows, taking full advantage of the network programmability provided by SDN. In particular, NMM takes a decision whether the assets (IIoT physical or virtual devices) related to the security events will be isolated or not by SDN-C. The continuous operation of the IIoT, such as CIs, is critical since possible disturbances can lead to more devastating consequences, cascading effects or even fatal accidents. Therefore, the NMM cannot instruct arbitrarily the SDN-C to drop the possibly malicious Modbus/TCP network flows. Such an irresponsible action by SDN-C could lead to a more severe impact than an actual Modbus/TCP cyberattack. For example, the impact of a Modbus/TCP reconnaissance cyberattack, such as *modbus/scanner/uid* and *modbus/scanner/getfunc* is less significant than a legitimate action targeting the availability of the relevant IIoT assets.

Moreover, the presence of a false positive alarm can result in the wrong decision. As presented in section 3, both CVSS and OWASP-RR can estimate the severity of the Modbus/TCP threats. However, the decision about isolating the assets affected by the security events cannot exclusively rely on these scores since (a) the sensitive nature of IIoT environments comprises extensive risks that are hard to estimate, (b) both CVSS and OWASP-RR do not consider the special peculiarities of an IIoT environment and (c) they cannot calculate the actual cost, which can be different for each organisation.

Based on the aforementioned remarks, NMM utilises an RL methodology to mitigate or even prevent the potential Modbus/TCP threats. In particular, for each security event, the response of NMM relies on three strategies: s_1: NMM will instruct SDN-C to isolate the assets affected by the security events, thus corrupting entirely the corresponding Modbus/TCP network flows, s_2: NMM will instruct SDN-C to drop some of the malicious Modbus/TCP network flows with a probability p_c, thus trying to thwart the cyberattackers' plans and s_3: NMM will wait for the security administrator to decide. The probability p_c in s_2 can be associated with parameters of the IIoT environment or the number of the security events.

Each strategy is characterised by a respective cost that can be related to financial damages, monetary claims, reputation damage, privacy violation or, in general, unit costs. In this paper, we use the general case of unit costs since we do not examine a particular IIoT case study. Moreover, we assume that the unit costs follow the normal distribution $N(\mu, \tau^{-}1)$. The goal is to train NMM to decide for each security event the appropriate strategy with the maximum expected reward, which corresponds to the minimum unit cost. The unit cost for each strategy is called *Return* and symbolised by x_i.

$$
\begin{aligned}
&p(\mu \mid X) \propto p(X \mid \mu)p(\mu) \\
&= \left(\prod_{i=1}^{N} \sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau}{2}(x_i-\mu)^2\right)\right) \cdot \\
&\left(\sqrt{\frac{\lambda_0}{2\pi}} \exp\left(-\frac{\lambda_0}{2}(\mu-\mu_0)^2\right)\right) \\
&= \left(\sqrt{\frac{\tau}{2\pi}}\right)^{N} \exp\left(-\frac{\tau}{2}\sum_{i=1}^{N}(x_i-\mu)^2\right)\sqrt{\frac{\lambda_0}{2\pi}} \cdot \\
&\exp\left(-\frac{\lambda_0}{2}(\mu-\mu_0)^2\right) \\
&\propto \exp\left(-\frac{\tau}{2}\sum_{i=1}^{N}(x_i-\mu)^2 - \frac{\lambda_0}{2}(\mu-\mu_0)^2\right)
\end{aligned}
\tag{11}
$$

$$
\begin{aligned}
p(\mu|X) &= \sqrt{\frac{\lambda}{2\pi} exp(-\frac{\lambda}{2}(\mu-m)^2)} \\
&= \sqrt{\frac{\lambda}{2\pi} exp(-\frac{\lambda}{2}(\mu^2-2m\mu+m^2))} \\
&\propto exp(-\frac{\lambda}{2}(\mu^2-2m\mu)) \\
&= exp(-\frac{\lambda}{2}\mu^2 + m\lambda\mu)
\end{aligned}
\tag{12}
$$

$$
\begin{aligned}
\lambda &= \tau N + \lambda_0 \\
m &= \frac{1}{\tau N + \lambda_0}(\tau \sum_{i=1}^{N} x_i + \lambda_0 m_0)
\end{aligned}
\tag{13}
$$

Equations (11) to (13) describe the Bayesian inference process that underpins the use of Thompson Sampling (TS) for decision-making in the SDN-based mitigation module of the proposed IDPS. These equations formally derive the posterior distribution of the expected cost μ for each available mitigation strategy, given observed historical data $X = \{x_1, x_2, \ldots, x_N\}$. Equation (11) combines the likelihood of observing the costs under a normal distribution assumption with a conjugate prior, resulting in a posterior probability distribution $p(\mu \mid X)$ for the strategy's expected cost. Equation (12) expresses the posterior as another normal distribution, simplifying the sampling procedure, and Equation (13) defines the updated parameters of this posterior-namely, the new precision λ and mean m, which are functions of the observed cost values and prior assumptions. In the proposed system, this Bayesian framework allows the mitigation module to balance exploration (trying underused strategies to learn more about their impact) and exploitation (favoring strategies that are likely to minimize disruption or cost). By sampling from the posterior rather than always choosing the strategy with the lowest average cost, the system can intelligently adapt to evolving IIoT conditions and security threats. This is especially crucial in environments where aggressive mitigation actions, like dropping network flows, could cause unintended outages or cascade effects. Thus, these equations enable cost-aware, probabilistic decision-making that reflects both learned experience and uncertainty.

Our decision problem can be considered as a MAB problem, where the NMM plays the role of the gambler and the various mitigation strategies correspond to slot machines. In a typical MAB problem, the gambler aims to maximize profit by choosing, at each time step, the slot machine offering the maximum payout. Since only one slot machine can be chosen at a time, the gambler faces an exploration-exploitation dilemma: exploration involves identifying the machine that yields the maximum profit, while exploitation focuses on maximizing the overall gain.

Unlike the typical MAB scenario, our goal is to minimize the possible cost associated with these mitigation strategies. To address this, we adopt the TS method. In our context, exploration means discovering more information about the cost of the different strategies, and exploitation means choosing the strategy that minimizes the cost of mitigating a Modbus/TCP threat. TS is a Bayesian method that leverages conjugate priors to compute the posterior probability $p(\mu \mid X)$.

In particular, given

$$X = x_1, x_2, \ldots, x_N,$$

the likelihood is defined as

$$
p(X \mid \mu, \tau) = \prod_{i=1}^{N} \sqrt{\frac{\tau}{2\pi}} \exp\left(-\frac{\tau}{2}(x_i-\mu)^2\right), \quad with \quad x_i \sim N\left(\mu, \tau^{-1}\right).
$$

Assuming that τ is known and that the prior for μ is given by

$$\mu \sim N(m_0, \lambda_0^{-1}),$$

the prior probability is

$$p(\mu) = \sqrt{\frac{\lambda_0}{2\pi}} \exp\left(-\frac{\lambda_0}{2}(\mu - m_0)^2\right).$$

Thus, the posterior probability is computed as

$$p(\mu \mid X) \propto p(X \mid \mu)\, p(\mu),$$

which leads to

$$\mu \mid X \sim N(m, \lambda^{-1}).$$

This Bayesian formulation allows TS to balance exploration and exploitation when deciding which mitigation strategy to apply. Our goal is to define the parameters m and λ of the posterior probability $p(\mu|X)$ as a function of the data X and the prior parameters m_0 and λ_0. Thus, based on Equations 11-13, $\lambda = \tau N + \lambda_0$ and $m = \frac{1}{\tau N + \lambda_0}(\tau \sum_{i=1}^{N} x_i + \lambda_0 m_0)$. Suppose μ follows the standard normal distribution, (i.e., $m_0 = 0$ and $\lambda_0 = 1$), for each security event, TS takes a sample from the posterior probability for each strategy: $N(m, \lambda^{-1}) \rightarrow N(0, 1)\sqrt{\frac{1}{\tau}} + m$, selecting the minimum value. Next m and λ are updated based on Equation 13. Algorithm 2 shows how the TS method is applied. The matrices: x_Matrix, sum_x_Matrix, λ_Matrix, and m_Matrix are used to store x_i, $\sum_{i=1}^{N} x_i$, λ and m for each strategy. N denotes the corresponding number of the latest security event, while S indicates a set of the three strategies: s_1, s_2 and s_3 described earlier.

For better understading of Algorithm 2, we include a simplified overview of the SDN-based mitigation approach with Thompson Sampling (TS). Upon detection of a Modbus/TCP threat, the proposed solution chooses between three response methods: fully isolate the attacked asset, partially drop malicious traffic, or wait for manual action. There is a hidden cost associated with each strategy, say, disruption of business or ineffectiveness in halting the threat. Rather than merely choosing the strategy with the lowest average past cost each time, the system models each strategy's cost as a probability distribution that is updated over time as events are experienced. At each decision, the proposed mechanism samples from these distributions and selects the strategy with the lowest sampled cost. In this way, the system is able to balance trying new strategies and exploiting known good ones. As it gets more data, the system becomes more confident in its decision and converges to optimal behaviour. This makes the proposed mitigation method adaptive, cost-aware, and well-suited to the dynamic and sensitive nature of IIoT environments.

Algorithm 2: SDN-based Mitigation - TS with Normal Distribution

Data: $S, \tau, m_0, \lambda_0, m, \lambda, x\_Matrix, sum\_x\_Matrix, \lambda\_Matrix, m\_Matrix$
Result: selectedStrategy
$securityEventCounter = 0$;
$\tau = 1, m_0 = 0, \lambda_0 = 1, m = 0$;
$x\_Matrix = [], sum\_x\_Matrix = [], \lambda\_Matrix = [], m\_Matrix = []$;
while *True* **do**
  Receive a security event;
  securityEventCounter = securityEventCounter +1;
  selectedStrategy = 0;
  min = ∞;
  for $strategy \leftarrow 0$ **to** S **by** 1 **do**
    posteriorProbabilitySample = $N(0,1)\sqrt{\frac{1}{\tau}} + m\_Matrix[selectedStrategy]$;
    if *posteriorProbabilitySample* $<$ *min* **then**
      min = posteriorProbabilitySample;
      selectedStrategy = strategy;
    end
  end
  SDN controller executes selectedStrategy;
  $x\_Matrix[selectedStrategy] = N(0,1)\sqrt{\frac{1}{\tau}} + \mu$;
  $sum\_x\_Matrix[selectedStrategy] = sum\_x\_Matrix[selectedStrategy] + x\_Matrix[selectedStrategy]$;
  $\lambda\_Matrix[selectedStrategy] = \lambda\_Matrix[selectedStrategy] + \tau$;
  $m\_Matrix[selectedStrategy] = \tau \times sum\_x\_Matrix = [selectedStrategy]/\lambda\_Matrix[selectedStrategy]$;
end

7 Evaluation analysis

Before presenting and discussing the evaluation results, we have to introduce the evaluation environment, the dataset used for this purpose and the respective evaluation metrics. In particular, we evaluate the efficiency of the proposed IDPS in terms of (a) Modbus/TCP threat detection and (b) mitigation performance. To this end, we used an Ubuntu 18.04.5 Long Term Support (LTS), 64-bit computing system with Intel Core i7-6700 Central Processing Unit (CPU), GeForce GTX 960 Graphics Processing Unit (GPU) and a Solid State Drive (SSD) with 245,1 GB. Towards the Modbus/TCP threat detection, we created a Modbus/TCP intrusion detection dataset, which is provided publicly through this work. This dataset is composed of pcap files and visual representations for the Modbus/TCP threats discussed earlier in section 3. Moreover, it includes Comma-Separated Values (CSV) files related to Modbus/TCP bidirectional network flow statistics generated by `CICFlowMeter`.

It is worth mentioning that active learning is also used to address the issue of dataset imbalance where specific kinds of Modbus/TCP malicious flows might be undersampled. Rather than using a static, potentially imbalanced dataset, the

proposed solution actively learns and selects the most informative and uncertain samples from an unlabelled repository of Modbus/TCP network traffic flows. A human oracle then validates these samples, creating a dynamically balanced and growing training set over time. Although standard techniques of data augmentation, such as converting images or creating synthetic data-were not used directly, converting network flow data into binary visual representations via Hilbert Curve mappings (using Binvis) naturally introduces variability in visual space. This method alleviates class imbalance effects by using general features learned from large-scale image datasets and Transfer Learning from a pre-trained ResNet50 model. Synthetic oversampling (e.g., SMOTE for tabular features) or adversarial data generation could be used in future work to better handle input-level imbalance.

To assess the evaluation performance, four evaluation metrics are adopted: (a) Accuracy (Equation 14), (b) TPR (Equation 15), (c) FPR (Equation 17) and (d) F1 score (Equation 18). Before discussing each of them, we need to introduce first some essential terms. TP denotes the number of the classifications that recognise the cyberattacks correctly. Similarly, TN expresses the amount of the correct classifications about the normal instances. In contrast, FP indicates the number of the mistaken classifications that categorise the normal instances as intrusions. Finally, FN denotes the wrong classifications that classify the cyberattacks as normal behaviours.

Accuracy represents the proportion of the correct classifications and the overall instances. It is a fair evaluation metric when the training dataset consists of an equivalent number of instances for all classes.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{14}$$

TPR or Recall expresses what ratio of the original malicious instances were detected as intrusions. TPR is calculated by dividing TP by the sum of TP and FN.

$$TPR = \frac{TP}{TP + FN} \tag{15}$$

Precision (Equation 16) measures how many of the samples predicted as positive (e.g., attacks) are actually positive. It's a measure of exactness or purity of positive predictions.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{16}$$

FPR denotes the ratio of the normal instances detected as malicious. FPR is computed by dividing FP by the sum of TN and FP.

$$FPR = \frac{FP}{FP + TN} \tag{17}$$

The F1 score represents the golden ratio between the TPR and Precision, considering both FN and FP. Precision is another evaluation metric, which computes the proportion of those data samples classified as cyberattacks. In particular, precision is calculated by dividing TP by the sum of TP and FP.

$$F1 = \frac{2 \times TP}{2 \times TP + FP + FN} = \frac{2 \times Precision \times TPR}{Precision + TPR} \tag{18}$$

The Area Under the Curve (AUC) (Equation]refauc) is another performance metric to assess the quality of a classification. It is the area under the Receiver Operating Characteristic (ROC) curve, i.e., the graph of the True Positive Rate (TPR) versus the False Positive Rate (FPR) at different classification thresholds.

$$\text{AUC} = \int_0^1 \text{TPR}(\text{FPR})\, d(\text{FPR}) \tag{19}$$

Several pre-trained CNNs are utilised in a comparative analysis with the aforementioned evlaution metrics, including (a) *DenseNet121*, (b) *DenseNet169*, (c) *DenseNet201*, (d) *EfficientNetB0*, (e) *EfficientNetB7*, (f) *MobileNet*, (g) *MobileNetV2*, (h) *NASNetLarge*, (i) *NASNetMobile*, (j) *ResNet50*, (k) *ResNet50V2*, (l) *ResNet101*, (m) *ResNet101V2*, (n) *ResNet152*, (o) *ResNet152V2*, (p) *VGG16*, (r) *VGG19* and (s) *Xception*. Furthermore, we include a comparative analysis with typical ML solutions, such as (a) *Logistic Regression*, (b) *Linear Discriminant Analysis (LDA)*, (c) *Decision Tree Classifier*, (d) *Naive Bayes*, (e) *SVM Linear*, (f) *SVM Radial Basis Function (RBF)*, (g) *Multi-Layer Perceptron (MLP)*, (h) *Random Forest*, (i) *Adaboost* and (j) *Quadratic Discriminant Analysis*. In addition, two custom Deep Neural Networks (DNNS) called (a) *Deep Dense Relu* [36] and *Deep Dense Tanh* [36] are used. The last DNNs originate from our previous work in [36]. The aforementioned ML and DL methods were trained with the bidirectional network flow statistics originating from *CICFlowMeter*. In the second case, regarding the mitigation performance, we examine how the posterior probability ranges with respect to the various number of security events for each strategy. For this purpose, we ran a simulation based on the Modbus/TCP intrusion detection dataset. The cost for each strategy was defined by IIoT security experts. Finally, we assess and compare the accuracy of the proposed TS method with a relevant method called Upper Confident Bound (UCB) with respect to choosing the optimal mitigation strategy.

Fig. 7 shows how the accuracy of the Active `ResNet50`-based CNN increases based on the updates of the new training

dataset. In particular, the x-axis denotes the time when a new training dataset is created and used, following Algorithm 1.

On the other hand, the y-axis indicates the new classification accuracy of the Active `ResNet50`-based CNN after each re-training process with the new training dataset. Consequently, each training process of the Active `ResNet50`-based CNN with a new training dataset corresponds to an accuracy value. Moreover, Table 2 summarises the evaluation metrics related to the pre-trained CNN models mentioned earlier after the last training process.

The pre-trained CNN models of Table 2 were re-trained under the same conditions based on the Modbus/TCP intrusion detection dataset provided by this work. The best detection performance is accomplished by `ResNet50`: $Accuracy = 0.984$, $TPR = 0.885$, $FPR = 0.008$ and $F1score = 0.885$. In addition, Fig. 8 illustrates how the loss function related to Active `ResNet50`-based CNN ranges per epoch. Totally, 200 epochs were used.

On the other side, `NASNetMobile` achieves the worst performance: $Accuracy = 0.961$, $TPR = 0.704$, $FPR = 0.020$ and $F1score = 0.709$. Table 3 depicts the evaluation results of typical ML solutions and two custom DNNs. Similarly, the Modbus/TCP intrusion detection dataset was utilised for the training process. The best performance is achieved by `Decision Tree Classifier`: $Accuracy = 0.964$, $TPR = 0.749$, $FPR = 0.019$ and $F1score = 0.749$, while `Adaboost` achieves the lowest efficiency: $Accuracy = 0.887$, $TPR = 0.214$, $FPR = 0.060$ and $F1score = 0.214$.

It is worth mentioning that Table 3 comprises the evaluation results of `Suricata`, which is a widely known signature-based IDPS. In particular, we utilised the Quickdraw ICS signatures [37]. The $Accuracy$, TPR, FPR and the $F1score$ related to the Suricata detection capacity are calculated at 0.787, 0.613, 0.000 and 0.578, respectively. In general, the efficiency of the pre-trained CNNs with the visual representations overcome the typical ML solutions, `Suricata` and the DNNs: (a) `Dense DNN Relu` and (b) `Dense DNN Tanh` that use `CiCFlowMeter` network flow statistics.

Regarding the mitigation performance, Fig. 9-17 illustrate how the posterior probability $p(\mu|X, \tau)$ ranges based on the number of 5, 10, 20, 50, 100, 200, 500, 1000, 1500 and 2000 security events. In particular, we observe that the more security events, the taller and skinnier Probability Density Function (PDF) for each strategy is, thus increasing our belief for the proper action. In our experiments, s_1 seems to be the appropriate strategy, where NMM will instruct SDN-C to corrupt all the malicious Modbus/TCP network flows. However, the choice differs from an IIoT environment to another IIoT environment since the related costs for each strategy are different. Moreover, Fig. 19 shows the distribution variance of the mean for each strategy based on the various security events. It is obvious that when the security events increase, the distribution variance of each strategy decreases. Furthermore, the first strategy presents the smallest variance for each number of security events. Finally, Fig. 20 compares TS and UCB with respect to selecting the optimal strategy. In general, TS overcomes UCB though the accuracy values are relatively close to each other.

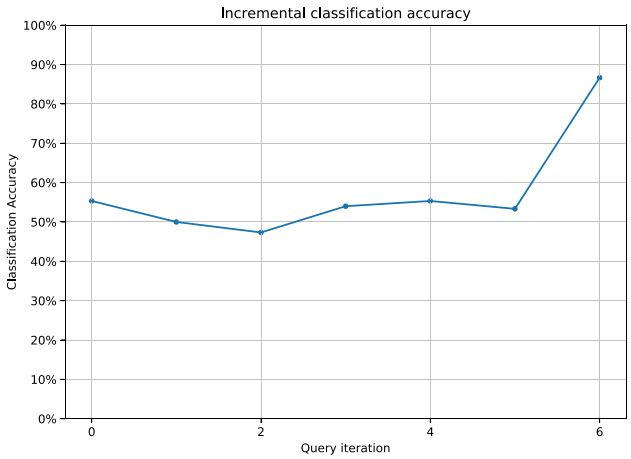

Fig. 7 Active ResNet50-based CNN - Accuracy increment during the re-training phases

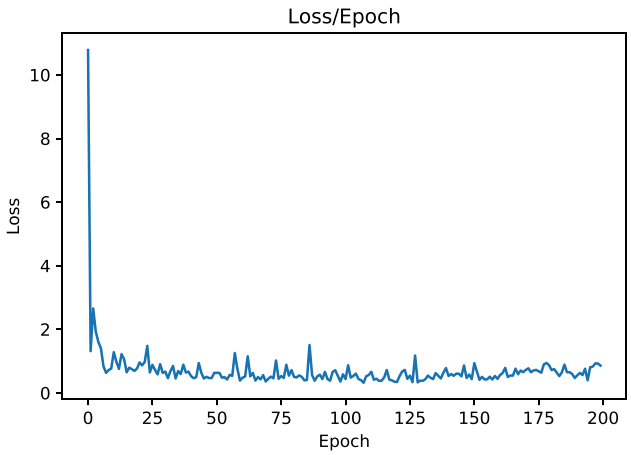

Fig. 8 Active-based ResNet50 CNN loss range

To evaluate the contribution of each component in the proposed Intrusion Detection and Prevention System (IDPS), we conducted an ablation study by systematically disabling or modifying specific modules and assessing the impact on detection and mitigation performance. First, we removed the Active Learning mechanism and used only the base ResNet50 model without iterative re-training. This resulted in a notable decrease in detection accuracy (from 98.4% to 92.1%) and a higher FPR, demonstrating the importance of continuous model refinement. Second, we replaced the binary visualisation pre-processing with raw flow-based features and observed a significant drop in model inter-

Table 2 Evaluation results of the pre-trained CNN models

Model	Acc.	TPR	FPR	F1	Prec.	AUC
DenseNet121	0.975	0.814	0.013	0.814	0.814	0.901
DenseNet169	0.975	0.818	0.012	0.819	0.820	0.903
DenseNet201	0.979	0.837	0.010	0.843	0.849	0.914
EfficientNetB0	0.981	0.858	0.009	0.859	0.860	0.925
EfficientNetB7	0.962	0.697	0.018	0.713	0.729	0.839
MobileNet	0.981	0.862	0.009	0.862	0.862	0.926
MobileNetV2	0.980	0.850	0.010	0.850	0.850	0.920
NASNetLarge	0.964	0.714	0.017	0.728	0.742	0.848
NASNetMobile	0.961	0.704	0.020	0.709	0.713	0.842
Active ResNet50	**0.984**	**0.885**	**0.008**	**0.885**	**0.885**	**0.939**
ResNet50	0.980	0.854	0.010	0.854	0.854	0.922
ResNet101	0.981	0.864	0.009	0.864	0.864	0.928
ResNet101V2	0.980	0.853	0.010	0.853	0.853	0.922
ResNet152	0.982	0.865	0.009	0.865	0.865	0.928
ResNet152V2	0.978	0.805	0.009	0.831	0.857	0.898
VGG16	0.977	0.822	0.011	0.829	0.836	0.906
VGG19	0.981	0.863	0.009	0.863	0.863	0.927
Xception	0.975	0.806	0.012	0.812	0.818	0.897

pretability and a 5.7% reduction in the F1 score, confirming the value of the Hilbert-curve-based image transformation. Third, we replaced Thompson Sampling (TS) in the SDN-based mitigation strategy with a UCB, which led to sub-optimal decisions and increased response cost under evolving threat conditions. These results collectively validate that each component-Active Learning, binary visualisation, and TS-based mitigation contributes meaningfully to the robustness and efficiency of the proposed solution.

Based on the aforementioned remarks, this paper proposes a novel hybrid solution that combines image-based

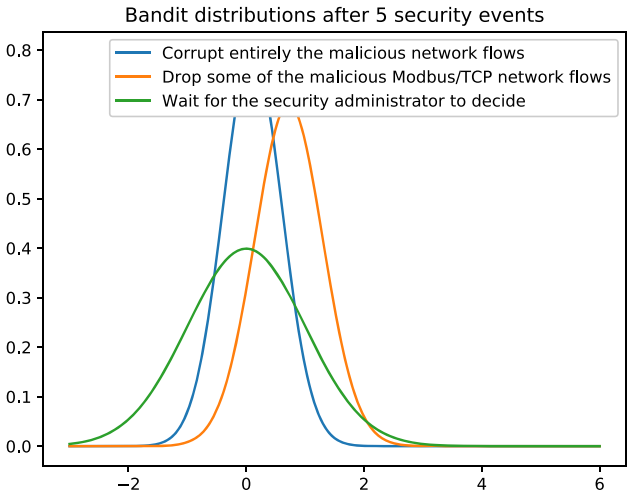

Fig. 9 Posterior probability after 5 security events

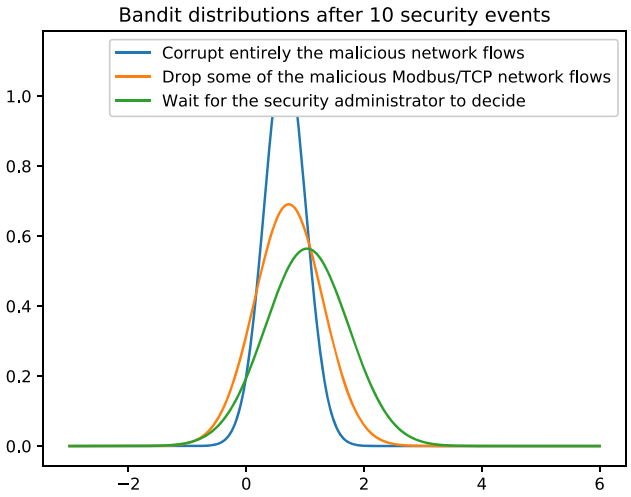

Fig. 10 Posterior probability after 10 security events

Table 3 Evaluation results of ML/DL solutions using CICFlowMeter statistics

ML Method	Accuracy	TPR	FPR	F1-score	Precision	AUC
Logistic Regression	0.943	0.603	0.030	0.603	0.603	0.786
LDA	0.943	0.604	0.030	0.604	0.604	0.787
Decision Tree	**0.964**	**0.749**	**0.019**	**0.749**	**0.749**	**0.865**
Naive Bayes	0.928	0.497	0.038	0.497	0.497	0.729
SVM (Linear)	0.921	0.453	0.042	0.453	0.453	0.706
SVM (RBF)	0.918	0.426	0.044	0.426	0.426	0.691
MLP	0.938	0.570	0.033	0.570	0.570	0.769
Random Forest	0.947	0.633	0.028	0.633	0.633	0.803
Adaboost	0.887	0.214	0.060	0.214	0.214	0.577
Quadratic Discriminant Analysis	0.941	0.593	0.031	0.593	0.593	0.781
Dense DNN (ReLU)	0.945	0.619	0.029	0.619	0.619	0.795
Dense DNN (Tanh)	0.945	0.619	0.029	0.619	0.619	0.795
Suricata	0.787	0.613	0.000	0.578	0.547	0.807

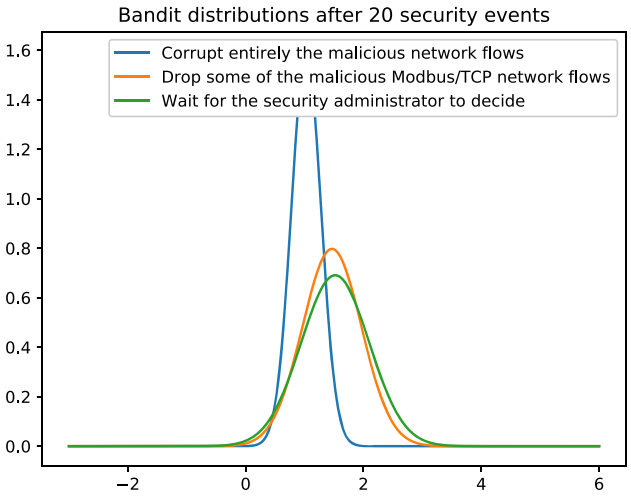

Fig. 11 Posterior probability after 20 security events

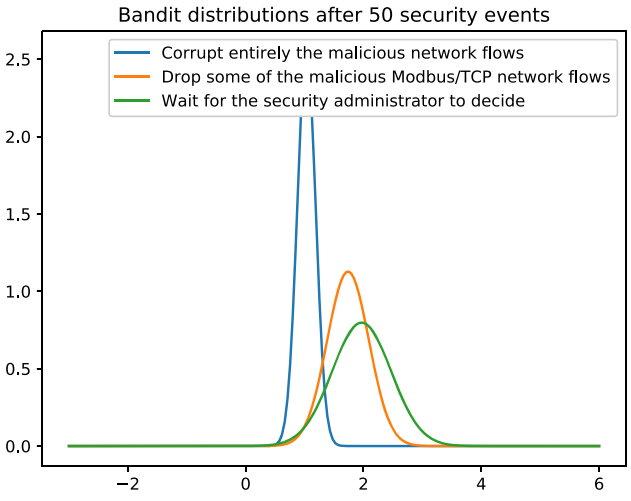

Fig. 12 Posterior probability after 50 security events

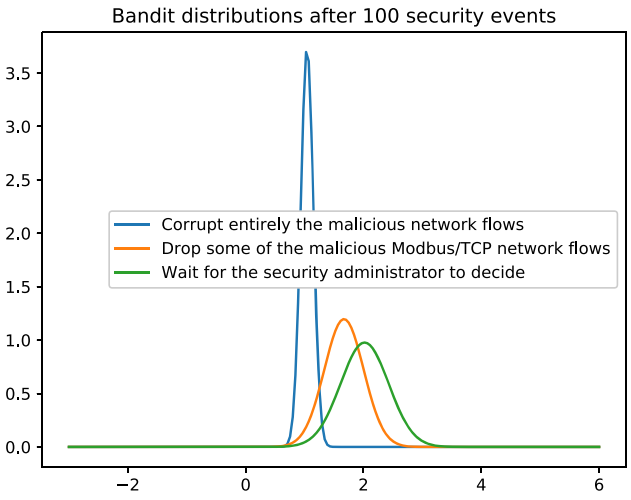

Fig. 13 Posterior probability after 100 security events

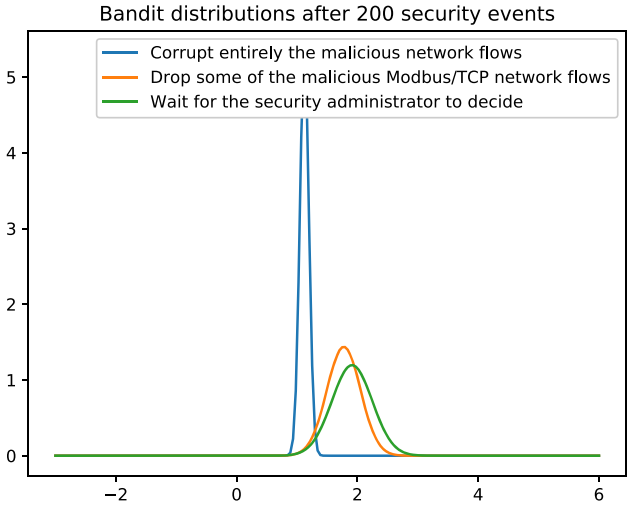

Fig. 14 Posterior probability after 200 security events

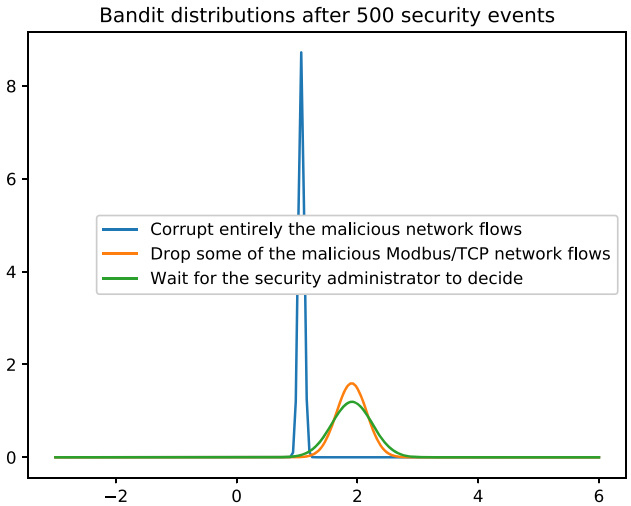

Fig. 15 Posterior probability after 500 security events

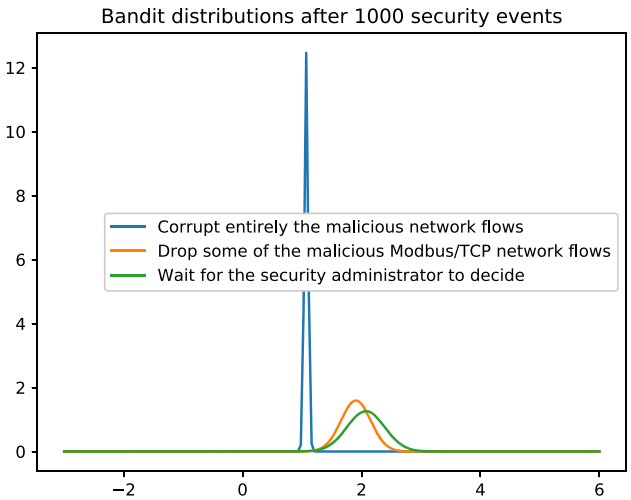

Fig. 16 Posterior probability after 1000 security events

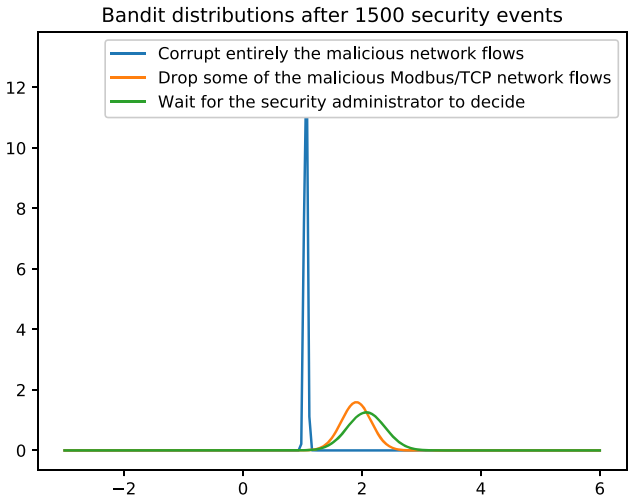

Fig. 17 Posterior probability after 1500 security events

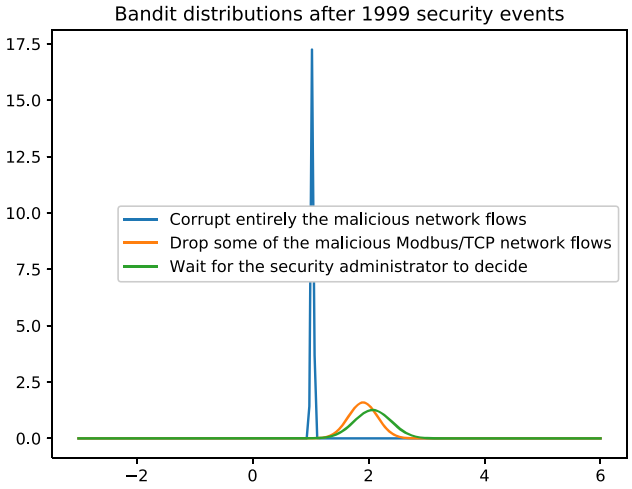

Fig. 18 Posterior probability after 2000 security events

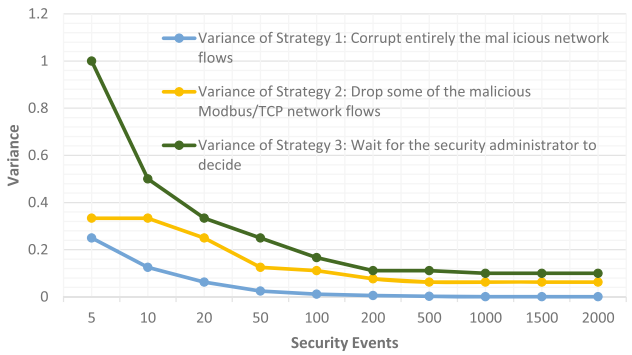

Fig. 19 Distribution variance of each strategy based on the various security events

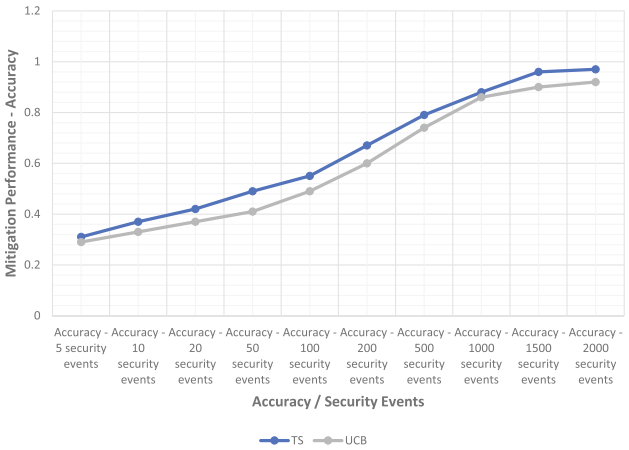

Fig. 20 Comparison between TS and UCB with respect to the mitigation accuracy

deep learning with a cost-sensitive, adaptive SDN-based mitigation mechanism, specifically for Modbus/TCP attacks in Industrial IoT networks. The scientific contribution is threefold: (1) the mapping of Modbus/TCP streams to binary visual representations via Hilbert curve mapping, which enables the application of powerful image-based CNNs for threat classification; (2) the incorporation of an active learning loop with human-in-the-loop feedback to enhance model flexibility and minimize false alarms over time; and (3) the application of Thompson Sampling-based decision-making in the SDN controller, which allows the system to trade off mitigation efficacy versus operational cost under uncertainty.

In comparison with other AI approaches and rule-based systems such as Suricata, the solution proposed is much more accurate and has better generalization with low false alarms. The approach does have limitations, though. The visual transformation pipeline creates computational overhead, and the active learning loop depends on occasional human feedback, which could be a bottleneck for full autonomous deployments. Furthermore, while the use of Thompson Sampling enables flexibility, its performance is contingent on the quality of cost feedback available to discover optimal actions. Despite such trade-offs, the system demonstrates improved performance in complex threat environments and presents a viable, interpretable, and adaptive solution for IIoT network security. Future work will focus on optimizing the inference efficiency of the model and generalizing the system to support encrypted traffic and zero-day attack scenarios.

8 Conclusions

The rise of IIoT provides multiple benefits, creating a new digital era in the industrial sector. Nevertheless, crucial cybersecurity concerns arise due to the insecure nature of the IIoT

communication protocols. In this paper, we focus on the Modbus/TCP threats. Modbus/TCP is an industrial protocol, which is usually adopted by IIoT environments. First, we introduce a Mobbus/TCP threat model, which calculates the severity score of the Modbus/TCP threats supported by relevant penetration testing tools. Next, we present an IDPS capable of detecting, distinguishing and mitigating the Modbus/TCP threats specified by the proposed Modbus/TCP threat model. The proposed IDPS uses an Active `ResNet5`-based CNN, which applies Active Learning and visual representations in order to re-train itself. Moreover, the proposed IDPS uses TS and SDN in order to mitigate the Modbus/TCP threats, taking into account the sensitive nature of the IIoT environments. The evaluation results demonstrate the efficiency of the proposed IDPS.

Acknowledgements This project has received funding from the European Union's Horizon Europe research and innovation programme under grant agreement No 101131292 (AIAS). Disclaimer: Funded by the European Union. Views and opinions expressed are, however, those of the authors only and do not necessarily reflect those of the European Union or European Commission. Neither the European Union nor the European Commission can be held responsible for them.

Funding Open access funding provided by HEAL-Link Greece.

References

1. Andronikidis, G., Eleftheriadis, C., Batzos, Z., Kyranou, K., Maropoulos, N., Sargsyan, G., Grammatikis, P.R., Sarigiannidis, P.: in *2024 IEEE International Conference on Cyber Security and Resilience (CSR)* (IEEE, 2024), 777–782
2. Kelli, V., Radoglou-Grammatikis, P., Lagkas, T., Markakis, E.K., Sarigiannidis, P.: in *2022 IEEE international conference on cyber security and resilience (CSR)* (IEEE, 2022), 351–356
3. Amponis, G., Radoglou-Grammatikis, P., Nakas, G., Goudos, S., Argyriou, V., Lagkas, T., Sarigiannidis, P.: in *2023 12th International Conference on Modern Circuits and Systems Technologies (MOCAST)* (IEEE, 2023), 1–4
4. Radoglou-Grammatikis, P., Liatifis, A., Dalamagkas, C., Lekidis, A., Voulgaridis, K., Lagkas, T., Fotos, N., Menesidou, S.A., Krousarlis, T., Alcazar, P.R. et al., in *Proceedings of the 18th International Conference on Availability, Reliability and Security*, 1–8 (2023)
5. Iturbe, E., Llorente-Vazquez, O., Rego, A., Rios, E., Toledo, N.: Unleashing offensive artificial intelligence: Automated attack technique code generation. Computers & Security **147**, 104077 (2024)
6. Alshamrani, A., Myneni, S., Chowdhary, A., Huang, D.: A survey on advanced persistent threats: Techniques, solutions, challenges, and research opportunities. IEEE Commun. Surv. & Tutor. **21**(2), 1851 (2019)
7. Sullivan, J.E., Kamensky, D.: How cyber-attacks in Ukraine show the vulnerability of the US power grid. Electr. J. **30**(3), 30 (2017)
8. Berguiga, A., Harchay, A.: An IoT-Based Intrusion Detection System Approach for TCP SYN Attacks. Comput., Mater. & Contin. **71**(2), 3839–3851 (2022). https://doi.org/10.32604/cmc.2022.023399
9. Massaoudi, A., Berguiga, A., Harchay, A.: Secure Irrigation System for Olive Orchards Using Internet of Things. Comput., Mater. & Contin. **72**(3), 4663–4673 (2022). https://doi.org/10.32604/cmc.2022.026972
10. Berguiga, A., Harchay, A., Massaoudi, A.: HIDS-RPL: A Hybrid Deep Learning-Based Intrusion Detection System for RPL in Internet of Medical Things Network. IEEE Access **13**, 38404 (2025). https://doi.org/10.1109/ACCESS.2025.3545918
11. Kotzanikolaou, P., Theoharidou, M., Gritzalis, D.: in *International Workshop on Critical Information Infrastructures Security* (Springer, 2011), 104–115
12. Mendes, G., Loew, A., Honkapuro, S.: in *2019 IEEE Power & Energy Society General Meeting (PESGM)* (IEEE, 2019), 1–5
13. Iturbe, E., Rios, E., Toledo, N.: in *2023 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)* (IEEE, 2023), 291–297
14. Asimopoulos, D.C., Radoglou-Grammatikis, P., Makris, I., Mladenov, V., Psannis, K.E., Goudos, S., Sarigiannidis, P.: in *Proceedings of the 18th International Conference on Availability, Reliability and Security* (2023), 1–8
15. Adawadkar, A.M.K., Kulkarni, N.: Cyber-security and reinforcement learning-a brief survey. Eng. Appl. Artif. Intell. **114**, 105116 (2022)
16. Grigoriadou, S., Radoglou-Grammatikis, P., Sarigiannidis, P., Makris, I., Lagkas, T., Argyriou, V., Lytos, A., Fountoukidis, E.: in *2023 IEEE International Conference on Cyber Security and Resilience (CSR)* (IEEE, 2023), 142–147
17. Liatifis, A., Dalamagkas, C., Radoglou-Grammatikis, P., Lagkas, T., Markakis, E., Mladenov, V., Sarigiannidis, P.: in *Proceedings of the 17th International Conference on Availability, Reliability and Security* (2022), 1–6
18. Radoglou-Grammatikis, P.: SecureCyber: an SDN-enabled SIEM for enhanced cybersecurity in the industrial internet of things. MMTC Commun.-Front. **18**(2), 16 (2023)
19. Zhuang, F., Qi, Z., Duan, K., Xi, D., Zhu, Y., Zhu, H., Xiong, H., He, Q.: A comprehensive survey on transfer learning. Proc. IEEE **109**(1), 43 (2020)
20. Ren, P., Xiao, Y., Chang, X., Huang, P.Y., Li, Z., Gupta, B.B., Chen, X., Wang, X.: A survey of deep active learning. ACM computing surveys (CSUR) **54**(9), 1 (2021)
21. Li, E., Kang, C., Huang, D., Hu, M., Chang, F., He, L., Li, X.: Quantitative Model of Attacks on Distribution Automation Systems Based on CVSS and Attack Trees. Information **10**(8), 251 (2019)
22. Radoglou-Grammatikis, P., Dalamagkas, C., Lagkas, T., Zafeiropoulou, M., Atanasova, M., Zlatev, P., Boulogeorgos, A.A.A., Argyriou, V., Markakis, E.K., Moscholios, I. et al., in *GLOBECOM 2022-2022 IEEE Global Communications Conference* (IEEE, 2022), 1856–1861
23. Houmb, S.H., Franqueira, V.N., Engum, E.A.: Quantifying security risk level from CVSS estimates of frequency and impact. J. Syst. Softw. **83**(9), 1622 (2010)

24. Huitsing, P., Chandia, R., Papa, M., Shenoi, S.: Attack taxonomies for the Modbus protocols. Int. J. Crit. Infrastruct. Prot. **1**, 37 (2008)
25. Baptista, I., Shiaeles, S., Kolokotronis, N.: in *2019 IEEE International Conference on Communications Workshops (ICC Workshops)* (IEEE, 2019), 1–6
26. Pérez-Díaz, J.A., Valdovinos, I.A., Choo, K.K.R., Zhu, D.: A flexible SDN-based architecture for identifying and mitigating low-rate DDoS attacks using machine learning. IEEE Access **8**, 155859 (2020)
27. Berde, P., Gerola, M., Hart, J., Higuchi, Y., Kobayashi, M., Koide, T., Lantz, B., O'Connor, B., Radoslavov, P., Snow, W., Parulkar, G.: (Association for Computing Machinery, New York, NY, USA, 2014), HotSDN '14, p. 1-6. https://doi.org/10.1145/2620728.2620744
28. Khan, R., McLaughlin, K., Laverty, D., Sezer, S.: in *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)* (IEEE, 2017), 1–6
29. Kordy, B., Pièvre-Cambacédès, L., Schweitzer, P.: DAG-based attack and defense modeling: Don't miss the forest for the attack trees. Computer science review **13**, 1 (2014)
30. Radoglou-Grammatikis, P., Siniosoglou, I., Liatifis, T., Kourouniadis, A., Rompolos, K., Sarigiannidis, P.: in *2020 9th International Conference on Modern Circuits and Systems Technologies (MOCAST)* (IEEE, 2020), 1–4
31. Shostack, A.: *Threat modeling: Designing for security* (John Wiley & Sons, 2014)
32. https://ryu-sdn.org/
33. Ong, B.L., Kiat Yeo, C.: in 9th IEEE Annual Ubiquitous Computing. Electronics & Mobile Communication Conference (UEMCON) **2018**, 412–417 (2018). https://doi.org/10.1109/UEMCON.2018.8796839
34. Pan, S.J., Yang, Q.: A survey on transfer learning. IEEE Trans. Knowl. Data Eng. **22**(10), 1345 (2009)
35. Li, D., Wang, Z., Chen, Y., Jiang, R., Ding, W., Okumura, M.: A survey on deep active learning: Recent advances and new frontiers (2024). https://arxiv.org/abs/2405.00334
36. Radoglou Grammatikis, P., Sarigiannidis, P., Efstathopoulos, G., Panaousis, E.: ARIES: a novel multivariate intrusion detection system for smart grid. Sensors **20**(18), 5305 (2020)
37. Wong, K., Dillabaugh, C., Seddigh, N., Nandy, B.: in *2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)* (IEEE, 2017), 1–5