This is a preprint version of the paper entitled "Attacking and Defending DNP3 ICS/SCADA Systems" which was presented in 2022 18th International Conference on Distributed Computing in Sensor Systems (DCOSS). The published version is available in IEEE Xplore: https://ieeexplore.ieee.org/document/9881726. Citation: V. Kelli, P. Radoglou-Grammatikis, A. Sesis, T. Lagkas, E. Fountoukidis, E. Kafetzakis, I. Giannoulakis and P. Sarigiannidis, "Attacking and Defending DNP3 ICS/SCADA Systems", 2022 18th International Conference on Distributed Computing in Sensor Systems (DCOSS), 2022, pp. 183-190, doi: 10.1109/DCOSS54816.2022.00041.

Attacking and Defending DNP3 ICS/SCADA Systems

Vasiliki Kelli†, Panagiotis Radoglou-Grammatikis†, Achilleas Sesis‡, Thomas Lagkas§, Eleftherios Fountoukidis¶, Emmanouil Kafetzakis‖, Ioannis Giannoulakis‖ and Panagiotis Sarigiannidis†

***Abstract*—The highly beneficial contribution of intelligent systems in the industrial domain is undeniable. Automation, supervision, remote control, and fault reduction are some of the various advantages new technologies offer. A protocol demonstrating high utility in industrial settings, and specifically, in smart grids, is Distributed Network Protocol 3 (DNP3), a multi-tier, application layer protocol. Notably, multiple industrial protocols are not as securely designed as expected, considering the highly critical operations occurring in their application domain. In this paper, we explore the internal vulnerabilities-by-design of DNP3, and proceed with the implementation of the attacks discovered, demonstrated through 8 DNP3 attack scenarios. Finally, we design and demonstrate a Deep Neural Network (DNN)-based, multi-model Intrusion Detection Systems (IDS), trained with our experimental network flow cyberattack dataset, and compare our solution with multiple machine learning algorithms used for classification. Our solution demonstrates a high efficiency in the classification of DNP3 cyberattacks, showing an accuracy of 99.0%.**

***Index Terms*—cyberattack, DNP3, ICS, Intrusion Detection, SCADA**

I. Introduction

Industrial Control Systems (ICS)/Supervisory Control And Data Acquisition (SCADA) systems' utility and benefit in the industrial domain is undoubtful [1] [2] [3]. Such solutions boost automation in contemporary, Internet of Things (IoT)-enabled industrial processes, by supervising, monitoring, and controlling the manufacturing procedures [4] [5]. Nowadays, an ever-increasing amount of industries are utilizing ICS/SCADA solutions for automation, with this phenomenon being especially highlighted in the energy sector [6] [7]. However, as the applications of intelligent monitoring systems increase, the attack surface against such critical infrastructures (CI) expands simultaneously [8] [9] [10]. The rate of cyberattacks is at an all-time high, and the industrial domain has not remained unscathed from this issue [11] [12] [13].

*This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No. 833955.

† V. Kelli, P. Radoglou-Grammatikis and P. Sarigiannidis are with the Department of Electrical and Computer Engineering, University of Western Macedonia, Kozani 50100, Greece - E-Mail: {vkelly, pradoglou, psarigiannidis}@uowm.gr

‡ A. Sesis is with the 0infinity Limited, Imperial Offices, London, UK, E6 2JG - E-Mail: achilleas@0infinity.net

§ T. Lagkas is with the Department of Computer Science, International Hellenic University, Kavala Campus, 65404, Greece - E-Mail: tlagkas@cs.ihu.gr

¶ E. Fountoukidis is with Sidroco Holdings Ltd, Petraki Giallourou 22, Office 11, 1077 Nicosia, Cyprus - E-Mail: efountoukidis@sidroco.com

‖ E. Kafetzakis and I. Giannoulakis are with Eight Bells Ltd, Agias paraskevis 23, P.C. 2002, Strovolos, Nicosia, Cyprus - E-Mail: {giannoul, mkafetz}@8bellsresearch.com

Attacks against CI can have disastrous consequences as past incidents have indicated [14] [15] [16]. Specifically, a cyber incident in 2015 targeting Ukraine's power grid resulted in a complete black-out for almost 225,000 people [17] [18]. Furthermore, STUXNET is considered to be the biggest cybersecurity incident to have ever occurred, since it targeted and succeeded in causing malfunctions in Iran's nuclear facilities [19] [20] [21]. STUXNET's success in nuclear-focused infrastructures raises significant concerns regarding the security measures adopted in CIs. A further concern is the fact that a lot of ICS/SCADA-oriented protocols, such as DNP3, a protocol heavily used in the manufacturing process and especially in smart grids, are not designed with security in mind [22], and have to rely on different protocols or software to perform that task [23].

As insinuated in the previous paragraphs, the rapid and accurate detection of threats, especially in the industrial domain and smart grids, is essential for mitigation purposes. Machine Learning approaches have nearly global application in all IoT domains [24] [25] [26]. The integration of such technology for the protection of ICS/SCADA systems is a common approach with machine learning contributing mostly to the creation of IDS, able to perform a variety of tasks, such as anomaly detection and classification of network traffic, intending to add a layer of protection against cyberattacks in the procedures of ICS/SCADA systems.

With the above topics into consideration, we are presenting our research centered around attacking and defending the industrial protocol DNP3, against malicious attempts. As such, the contribution of this paper is two-fold:

- Investigate, describe and implement 8 DNP3 protocol-specific cyberattacks and malicious attempts
- Design and implement a machine learning-based, multi-model cyberattack classification IDS, trained to recognize all of the aforementioned DNP3 attacks

The paper is organized as follows. In Section II, current research on cybersecurity regarding DNP3, and IDS is presented. Section III aims in providing a background on the DNP3 protocol, how it operates, and the various layers it is composed of. Next, the detailed description of the attacks conducted in a simulated DNP3-enabled infrastructure, are explained in Section IV. Following, our multi-model DNP3 cyberattack detection and classification method is presented in Section V, and the evaluation of the aforementioned approach is conducted in Section VI, followed by the conclusions of our research in Section VII.

II. Related Work

Undoubtedly, the topic of securing the extremely critical industrial sector is a top research priority, given the continually expanding attack surface and the potentially disastrous effect of cyberattacks in such domains. As such, the aim of this chapter is to present similar research works that have been conducted in the field of cybersecurity in ICS.

Radoglou-Grammatikis et.al. [27] suggest a network flow-based intrusion and anomaly detection and prevention technique, named DIDEROT, using both supervised and unsupervised machine learning. Network flows are classified with a Decision Tree model trained on various attacks according to the Rodofile dataset, as well as normal behavior collected from real-world substation traffic. If the flow is classified as non-malicious, the autoencoder gets activated and searches for possible anomalies. If a malicious activity or anomaly is discovered, the Response module informs the SDN controller to drop the specific flow. According to the evaluation results, both the autoencoder and decision tree were able to achieve very high accuracy and F1 score, namely 99.7 percent and 99.1 percent respectively for the decision tree, and 95.1 percent and 95.3 percent respectively for the autoencoder.

Another important work is presented in the field of cybersecurity in [28] by Radoglou-Grammatikis et.al., where the authors are introducing the Secure and PrivatE smArt gRid (SPEAR) Security Information and Event Management (SIEM) system, able to directly address the needs and peculiarities of smart grid ecosystems. Their SPEAR SIEM tool is able to detect attacks targeting multiple application layer protocols used in smart grids, while the proposed solution's efficiency is demonstrated through 4 real-life smart grid infrastructures.

A work addressing the DNP3 and its vulnerabilities is presented in [29] by Darwish et.al., where the authors highlight the weaknesses of DNP3 and continue by implementing two attack scenarios against the protocol, where the former aims in intercepting and stopping an unsolicited message from the slave, and the latter modifies the content of the TCP payload to manipulate and redirect the DNP3 traffic. In addition, the authors propose a round trip time delay (RTTD) and a pass-and-drop algorithm-based attack detection and mitigation strategy.

Lin et. al. [30] propose a technique for protecting SCADA systems based on the Bro network traffic analyzer. Specifically, the authors introduce a custom DNP3 analyzer based on Bro, to get a concrete view of SCADA-related events. In addition, they constructed a protocol validation policy to verify that the DNP3 network traffic adheres to the predefined communication patterns.

Irvene et.al. argue with the fact that machine learning options constitute the best solution for creating efficient IDS. Specifically, with their work in [31], they demonstrate a lightweight security mechanism for substations utilizing DNP3. Furthermore, data were collected and analyzed through the span of 2.5 years from four real energy plants, concluding to the observation that master's and outstations' communication does not present the variety in application layer function codes assumed in many works. Taking into consideration the monotone communication patterns in regular traffic, a rule-based system for detecting abnormalities was realized, based on characteristics including the frequency of appearance of a specific function code and the interarrival time of packets presenting the same function code.

Amoah et.al. [32] emphasize the lack of security solutions for the case in which the master broadcasts messages to the outstations in large-scale infrastructures. Thus, the authors present a novel hash-chain-based lightweight security measure for broadcast mode communications, entitled DNP3 Secure Authentication for Broadcast (DNP3-SAB). The solution of DNP3-SAB is tested and verified through the utilization of Coloured Petri Nets (CPN), against a variety of common DNP3 cyberattacks.

Further addressing the security issues of DNP3, Bagaria et.al. [33] address the protection of existing and future infrastructures utilizing the protocol for their internal communications. Specifically, they designed an encryption bump-in-the-wire prototype, able to blend in with legacy systems without the need of modifying them.

Undoubtedly, a lot of great research has been conducted in the previous years, focusing on the protection of the highly critical SCADA systems and the industrial domain as a whole. To aid in the previous research efforts, we are presenting in the following sections an in-depth view on vulnerabilities-by-design and exploitable points of DNP3, and our multi-model machine learning tool for the detection of DNP3 cyberattacks.

III. DNP3 Background

DNP3 is a multi-tier application layer protocol utilized in the industrial domain, with its application being highlighted in the energy sector and smart grids [34]. DNP3 usually runs on top of Transmission Control Protocol (TCP) and occupies port 20000. DNP3 follows a client-server model, according to which there are two entities, the master with client functions, and the slave or outstation with server functions, whose aim is to provide responses to the master's requests [35]. This model gives the master the opportunity to supervise, control, and acquire data from slaves, thus fully controlling the production processes. Regarding the protocol's architecture, it is divided into 3 layers, namely the Data Link layer, the Transport layer, and the Application layer.

The Link layer is responsible for sending and receiving frames, and contains header information such as source DNP3 address and destination DNP3 address, while it is also responsible for calculating errors through Cyclic Redundancy Check (CRC), and checking the link's status. The Transport layer's purpose lies in the fragmentation of large packets received by the Application layer, while its header contains the information required to reassemble the aforementioned fragments. Finally, the Application layer creates the message to be communicated; however, this layer's header differs, depending on whether the message creator is a master or a slave, as the latter's header contains the Internal Indications field to better describe the node's status. Still, in both cases, the function code field describes the way the received message should be handled. Figure 1 represents the information flow through the multi-layered architecture of DNP3, during message creation.

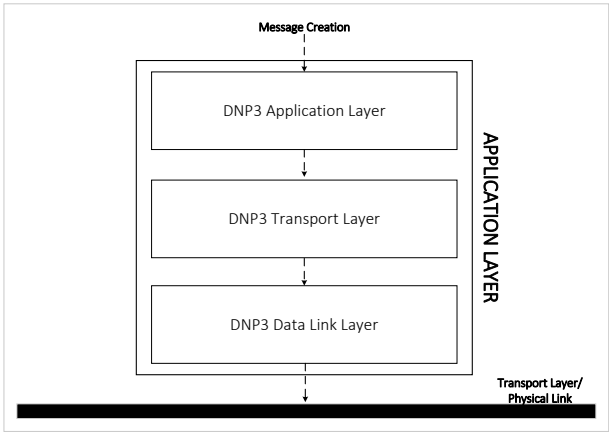

Fig. 1. Flow of data within DNP3

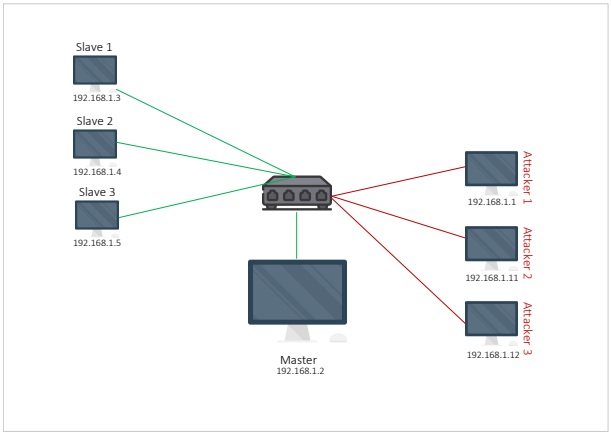

Fig. 3. Topology 2

IV. Attacking DNP3 SCADA/ICS and Data Collection

DNP3 is a protocol widely used in ICS for automating the control and supervision of the production process, in the electricity, oil, and water industries. Notably, over 75% of North America's electrical plants utilize DNP3 as a SCADA protocol, to allow communication between field devices and master stations [36]. Therefore, attacks against a protocol of such extreme utilization in highly critical domains can have disastrous consequences. The aim of this section is to provide a summary of the attacks which can have the highest impact on DNP3-enabled SCADA environments, and describe the training dataset generated through the execution of the aforementioned attacks.

Our experimental setup included two different topologies of master and slave instances, which can be observed in the Figures 2 and 3 below. All DNP3 communications were simulated with opendnp3 [37], which is an open-source application of the DNP3 protocol written in C++. A total of 8 attack scenarios were executed against the simulated DNP3 environments, capturing the network traffic generated through this process. Through all the attacks, we assume that the attacker has already obtained access to the DNP3 network. In the following subsections, an overview of the attacks performed is given.

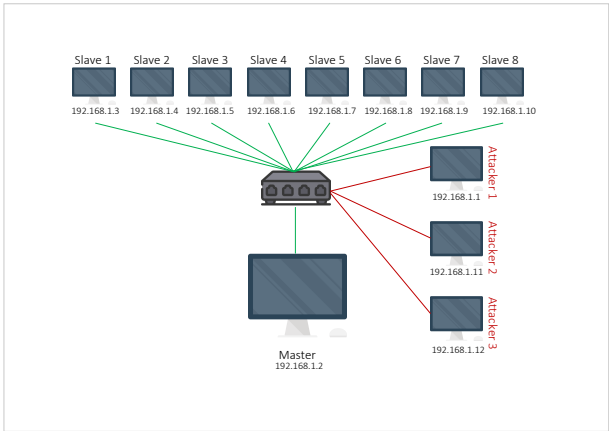

Fig. 2. Topology 1

A. Disable Unsolicited Messages

The first attack conducted, had the objective of ordering the slave to disable the functionality of sending unsolicited messages to the master. As the slave can use this function to rapidly notify the master of possible faults, disabling it may cause the slave to delay in communicating the abnormality through an unsolicited response. This specific attack took advantage of the fact that slaves conform to the order regardless of whether the node sending the order is a legitimate master. As such, we had our attacking nodes of topology 1, as seen in Figure 2 connect as masters to each slave and request them to disable their spontaneous messages, by sending a DNP3 packet with an application layer function code 21. This attack was repeated in the [20, 30] second interval by each attacking node, with the aim to create a large enough training dataset.

B. Cold Restart

The second attack executed, had the objective to force the slave to fully restart and go through all the self-check processes, through a cold restart packet. This specific attack takes advantage of regular DNP3 functions to possibly send the slave offline for some time and cause a Denial of Service (DoS) as the slave fails to respond to the masters' requests. As such, similarly to the implementation of the first attack, we took advantage of the fact that slaves reply to any node sending master requests, and had each attacking node of topology 1 send DNP3 requests with application layer function code 13 to each slave, repeatedly, in a [20, 30] second interval.

C. Warm Restart

Following the idea of the second attack, our third attack has the aim of causing the slave to restart DNP3 applications only, but not perform a full restart like the second attack. As a result, the slave becomes unresponsive for some time, until the application's restart process is completed. To that aim, we send a warm restart request packet containing the DNP3 application layer function code 14 by each attacker impersonating a master, according to topology 1, similarly to the previous attack scenarios. Similar to the previous scenarios, the attack is repeated in the [20, 30] second interval, for the training dataset generation process.

D. Slave Discovery

Our fourth and fifth malicious actions against DNP3 involved launching two Nmap Scripting Engine (NSE) against our DNP3 slave nodes, whose aim is to recognize whether the given Internet Protocol (IP) address belongs to a DNP3 slave. The former NSE sends to the target IP a packet to request the status of the link, with data link layer function code 9, to the first 100 DNP3 data link addresses in order to check whether a response is given. Similarly, the latter NSE sends requests to the first 100 DNP3 data link addresses of the given IP address and awaits for a response. In both cases, if a response is given, then the NSE successfully recognized a DNP3 slave and its data link address, running on the target IP. The scripts were run against our second topology of DNP3, as seen in Figure 3, while the scanning was executed in a repeated manner, randomly every [20, 30] seconds.

E. Initialize Data

On the same logic as the attacks aiming to exploit DNP3 by leveraging the protocol's application layer function codes, our sixth attack aimed to command the slave to initialize its data. As such, the objective was to request the slave to reset its data to the initial values, thus, any updates sent from the point of the attack, will not reflect the system's actual status. This attack was achieved by intercepting a legitimate packet from the master via Man-In-The-Middle (MiTM) approaches, and modifying it with scapy [38] to set the function code of the application layer to 15 and handle all the necessary changes to the payload accordingly. The modified packet is then injected back into the traffic. This attack was run against the setup of the second topology as seen in Figure 3, while it is also repeated in the [20, 30] second interval.

F. Stop Application

Taking a system offline by deliberately causing DoS is an extremely common cyber incident. DNP3 function codes can play an important part as previously explained, in silently making the slaves unresponsive. Specifically, DNP3 features a master request function code to order the slave to immediately cease the function of the DNP3 application and thus no longer respond to DNP3 requests. Our seventh attack is accomplished by capturing a request packet via MiTM, accordingly modifying its fields via scapy, setting the application layer function code to 18, and injecting it back in the traffic. Similar to the attack above, this scenario was conducted in the second topology and involved sending modified packets every [20, 30] seconds.

G. Replay

Our eighth and final attack scenario involved the repetition or the delay in transmission of a non-malicious packet. Although this specific attack was not centered around DNP3 and its functions, it is still an effective measure of causing disruptions and obstructing communications between DNP3 nodes. To implement this scenario, MiTM techniques, specifically, Address Resolution Protocol (ARP) Poisoning, was utilized to capture traffic of the second topology, save the packets and then delay their transmission in a random time, in the [5, 10] second interval.

During the execution of the aforementioned attack scenarios, all traffic was being continually captured by each node in the corresponding topology. As a result, we have curated a set of datasets to train our DNNs, able to capture and represent the attacks conducted in TCP/IP flows, and DNP3-specific flows. In detail, we captured the network traffic generated by each node during each attack's execution, and transformed the packet capture files into TCP/IP flows, through the utilization of CICFlowMeter [39], and flows centered around DNP3 attributes, through a custom flow generator. Finally, we proceeded by labeling the datasets with the corresponding attack occurring during each flow, in order to train the classifiers.

V. Detecting DNP3 Attacks

In the previous section, a description of cyberattacks exploiting DNP3-related vulnerabilities was given. Evidently, the successful execution of the aforementioned attacks in a DNP3-based SCADA environment can result in the disruption of services, alteration of messages, and avert the delivery of critical notifications. As such, solutions aiming to rapidly and accurately detect malicious attempts are necessary to avoid cyberattack-related consequences. In this section, we are presenting our multi-layered Deep Learning-based IDS, which aims to classify potential threats by leveraging network flows related to DNP3 traffic.

Our proposed flow-based IDS solution is divided into 4 components, layered to operate in collaboration to timely recognize attacks targeting DNP3-based SCADA infrastructure. Our first component is responsible for sniffing network traffic in the form of packets; next, the captured traffic is transformed into network flows in our second component. The aforementioned flows are generated through the utilization of 2 different tools, the one being CICFlowMeter, which handles the transformation of traffic into TCP/IP flows, and the other one being a custom flow generation tool able to extract flows based on DNP3-specific attributes. After the finalization of the network traffic analysis through our second component, our third machine learning-based component handles the classification of the extracted flows. Specifically, our DNN models are trained to recognize the 8 DNP3-based attacks demonstrated in Section IV, as well as normal traffic, while CICFlowMeter additionally recognizes ARP poisoning flows. As a result, the network flows extracted in our second component, get labeled through the inference mode of our third component, resulting in the creation of the fully labeled file. Our fourth and final component is responsible for creating alerts in case a cyberattack is detected by the DNNs, concluding a circle of operation of our IDS. The components making up our DNP3 IDS solution, as well as their interplay, can be observed in Figure 4 below in a step-by-step manner, while their operation is explained in detail through the following subsections.

A. Sniffing and Flow Generation

Capturing the traffic generated through the communication of ICS/SCADA systems is a crucial first step when creating an IDS. This ensures that communications from all parties are

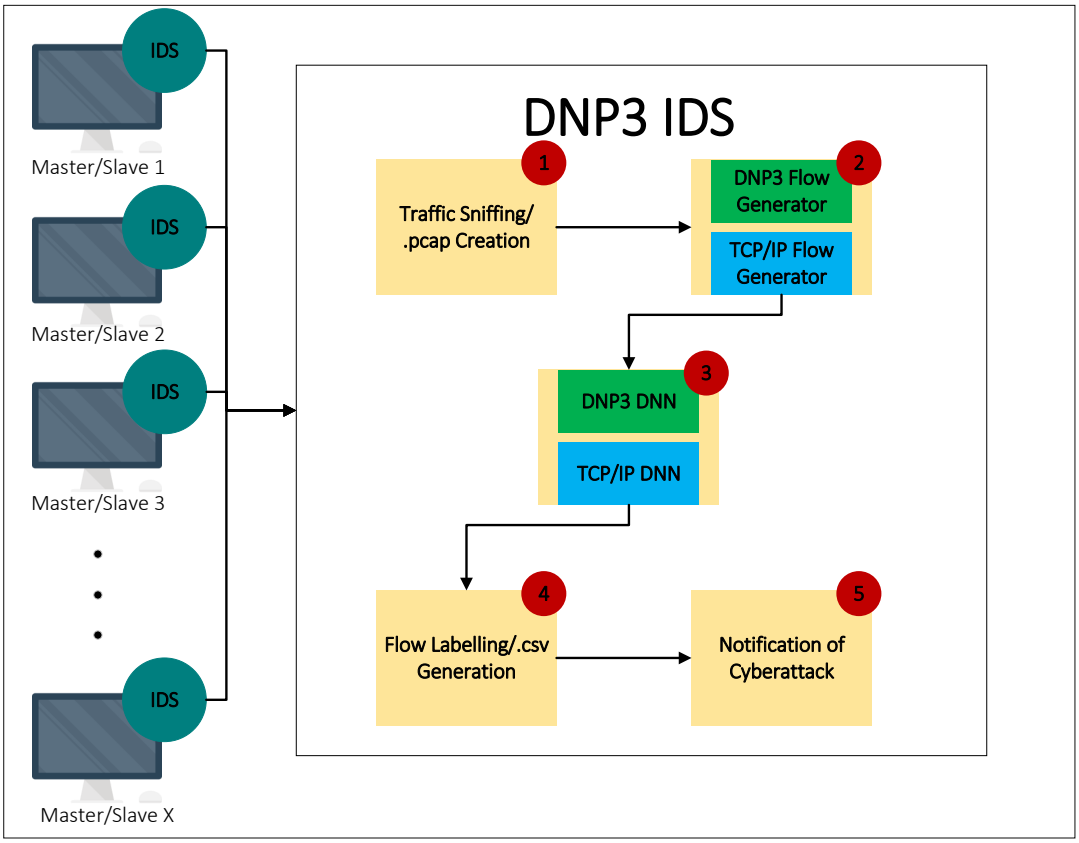

Fig. 4. ICS/SCADA DNP3 IDS

captured and registered in a common format, to facilitate the successful execution of the next step, which is the analysis of the communications. Our IDS solution captures communications in a network traffic format, meaning, it sniffs packets from each system it is installed in. Specifically, it captures packets throughout a pre-defined time window in each entity, and stores the results in a packet capture (.pcap) file.

Next, the analysis of the .pcap files follows, in a flow-based manner. According to Request for Comments (RFC) 2722 [40], the operation of a network flow is comparable to a phone call action. There is a source, a destination, data exchanged throughout its duration, a start, and an end. Our purpose is to utilize network flows to structure and analyze the packets captured through the sniffing procedure.

To achieve that, we use two different flow-generating software tools to thoroughly analyze the .pcap files. The first one, is CICFlowMeter, which takes as an input the packets and extracts the network flows in a Comma Separated Values (.csv) format, taking into account a predefined time duration for each flow. Each network flow instance is characterized by more than 80 features centered around the transport layer of the packets, as well as interarrival time-related data. The second software used to extract flows from .pcap is a custom-made DNP3-centered tool written in python. Our tool utilizes the well-known packet dissector scapy and its DNP3 library to extract essential features from all DNP3 packets, and form the flows. The analysis begins by taking the first DNP3 packet in the .pcap file and searching the rest of the file for packets captured in a pre-defined time window, with the same or reversed characteristics of the source and destination IP address and ports. In case of matches, then a flow is formed, consisting of over 100 features centered around attributes of all the layers of DNP3, and time-related data. This process is repeated until there are no packets left in the .pcap to analyze. Similar to CICFlowMeter, our DNP3-based tool stores the results in a .csv format to facilitate the training procedures of machine learning models.

B. Deep Neural Network and Attack Response

DNNs are heavily utilized to aid in resolving complex problems that would have been extremely difficult to address through different means. As such, our attack detection and classification models demonstrated in this paper, follow a DNN scheme. The above procedure of traffic analysis and categorization in network flows aims to prepare data to be used for training a DNN or to use it as a means for the extraction of results. As such, in the former case, the dataset is labeled depending on the attack that took place during each flow's duration. In the latter case, the unlabeled dataset is used as an input to the DNN which is considered to be the core

of the machine learning-based IDS, in order to have it label automatically each flow, according to the training it received.

We have developed a set of DNNs following a similar process and architecture as explained in the paragraphs below, which are at the center of the implemented DNP3 IDS, each one individually trained with data generated through the procedure mentioned in V-A. Moreover, one DNN is trained with CICFlowMeter-generated flows, while the second DNN is trained with flows extracted through our custom-made DNP3 parser; in both cases, the datasets were labeled accordingly. The structure of the cyberattack detection and classification DNNs can be observed in Figure 5. As a first dataset preprocessing step to prepare them for the training process, we applied feature normalization to turn data into values in the $[0, 1]$ range, where x_{sc} is the scaled feature value, $\mathbf{x}$ is the feature vector and x_i is the initial feature value;

$$x_{sc} = \frac{x_i - \min(\mathbf{x})}{\max(\mathbf{x}) - \min(\mathbf{x})} \tag{1}$$

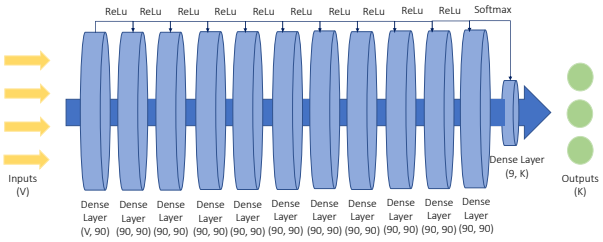

Fig. 5. Structure of Attack Detection and Classification DNNs

Furthermore, our DNNs' architectures consist of 10 layers, optimized through Adam with a learning rate of 0.0001. Both of the attack detection models are compiled with Categorical Crossentropy (2), as a loss function, where K indicates the number of the classes, b_{kc} which is a binary pointer, indicating whether the kth input belongs to the c class, and finally the output o_{kc}, which denotes the probability predicted by the model, that the kth input belongs to class c.

$$\mathcal{L}_{CCE} = -\sum_{c=1}^{K} b_{kc} log(o_{kc}) \tag{2}$$

Every Dense layer in our neural networks apart from the output one, is activated through the ReLu activation function (3), where x denotes the value inputted to the neuron:

$$f(x) = \begin{cases} 0 & \text{if } x \leq 0 \\ x & \text{if } x > 0 \end{cases} \tag{3}$$

Finally, the last or the output layer is activated through the Softmax activation function as seen in Equation (4) in both of our DNNs, which can transform the input values to probabilities. In detail, Softmax calculates a probability distribution of class membership for every output of the final layer. This is achieved by dividing the exponential value of output z_i with the summation of all exponentials:

$$SM(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \tag{4}$$

Our training process for both of the DNNs consumed about 715MB of memory for both cases, including the aforementioned data pre-processing steps. It consisted of 1000 epochs, and resulted in the generation of two different DNN-based models; one able to perform multi-class classification of attacks with flows generated through our custom DNP3 parser, and the other able to perform the same task but with flows created through CICFlowMeter. As such, during the operation of our IDS, two separate .csv files containing data from the two flow generation software are used as an input to each corresponding model; thus, the output of our IDS would be two fully labeled .csv files consisting of the network flows and their characterization, according to the underlying model. The timely and accurate detection of threats is of high importance in the mitigation of a cyberattack. To this end, in case a DNP3 cyberattack is detected by either model, an alert is generated with the malicious flows and the DNNs' classification of them, to help the handlers react promptly and address the issue, thus mitigating the impact of the cyberattack. The evaluation analysis and the performance comparison of both models is explained in detail in the following section.

VI. Evaluation Analysis

In order to fully and thoroughly evaluate the performance of the two flow generation tools in terms of model generation, we have conducted experiments featuring multiple supervised machine learning classifiers such as Decision Tree, DNN, K-Nearest Neighbor (k-NN), Naive Bayes, and Random Forest. Specifically, we have trained the aforementioned classifiers with datasets extracted through the two network flow generation tools, namely the DNP3-based flow tool and CICFlowMeter, to locate the solution that provides the best means to resolve the problem of creating a DNP3 cyberattack detection software.

It is worth noting that, since our custom DNP3 flow generation tool focuses solely on DNP3 attributes, it is not designed to target or categorize in flows packets of Open Systems Interconnection (OSI)'s data link layer; thus, any ARP poisoning attempts are not detected through the custom tool. Instead, CICFlowMeter is used as a means to additionally detect this cyberattack, and provide the relevant flows and statistics correlated with ARP poisoning scenarios. Hence, the models trained based on CICFlowMeter flows, are inclusive of ARP poisoning flows.

The metrics used to measure the performance of the classifiers, are the accuracy, precision, F1-score and recall, described in Equations 5, 6, 7 and 8 respectively. In detail, TP denotes the number of true positives, TN denotes the number of true negatives, FP denotes the number of false positives and finally, FN denotes the number of false negatives classified by the model. The results can be observed in Tables I for the DNP3 flow-based training and II for the CICFlowMeter-based training.

$$accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{5}$$

$$precision = \frac{TP}{TP + FP} \tag{6}$$

$$F1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)} \tag{7}$$

$$recall = \frac{TP}{TP + FN} \tag{8}$$

TABLE I
DNP3 FLOW-BASED CLASSIFIER EVALUATION

	Decision Tree	DNN	K-NN	Naive Bayes	Random Forest
Accuracy	0.9305	0.9900	0.9494	0.6827	0.905
Precision	0.9306	0.9580	0.9497	0.7222	0.9053
F1	0.9305	0.9565	0.9494	0.6490	0.9049
Recall	0.9305	0.9549	0.9494	0.6827	0.9049

TABLE II
CICFLOWMETER-BASED CLASSIFIER EVALUATION

	Decision Tree	DNN	K-NN	Naive Bayes	Random Forest
Accuracy	0.8818	0.9763	0.8794	0.6377	0.8690
Precision	0.8818	0.8845	0.8797	0.6746	0.8694
F1	0.8818	0.8832	0.8794	0.6022	0.8690
Recall	0.8818	0.8819	0.8794	0.6377	0.8690

As it can be observed from the tables above, the DNN was able to yield better performance metrics in both flow generation cases in comparison with the rest of the classifiers. This indicates that our choice of utilizing DNNs for attack detection and classification, is supported by the resulting performance evaluation, as following a DNN architecture seems to increase the model's reliability in categorizing network flows according to the underlying traffic, in both flow generation cases.

Regarding the comparison of the two DNN models resulting from training a similar DNN architecture with two different datasets, one DNP3 flow-based and one based on TCP/IP flows as generated through CICFlowMeter, the above performance results in Tables I and II respectively, indicate increased metrics for our DNP3 flow-based training. Furthermore, Figure 6 and Figure 7 represent the resulting confusion matrices for both the DNN models.

According to the results above, the DNN trained on DNP3 flows is able to recognize more accurately and reliably the vast majority of cyberattacks against DNP3. In contrast, the DNN trained on CICFlowMeter flows, shows a decrease in evaluation metrics and thus is not able to recognize DNP3 cyberattacks as accurately as its competitive DNN; it is however able to classify correctly all samples of the ARP Poisoning class, according to the confusion matrix in Figure 7. Hence, the TCP/IP DNN complements the operation of the DNP3-targeted DNN, by providing highly accurate detection of ARP Poisoning instances.

VII. CONCLUSIONS

As discussed in the previous sections, protocols intended for industrial applications are inherently insecure and present vulnerabilities by design. The purpose of this paper included a provision of a detailed overview of the security soft spots of

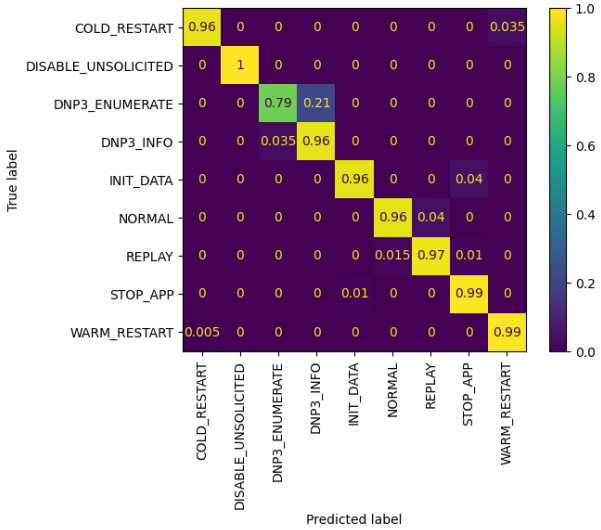

Fig. 6. Confusion Matrix of DNN trained with DNP3 flows

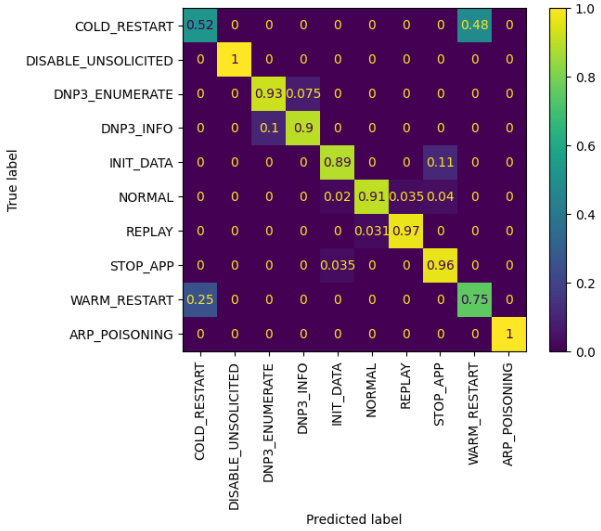

Fig. 7. Confusion Matrix of DNN trained with TCP/IP flows

DNP3. Specifically, we identified and implemented 8 DNP3-centered cyberattacks and malicious actions, namely Disable Unsolicited Messages, Cold and Warm Restart, two Slave Discovery nmap scripts, Initialize Data, Stop Application, and Replay. Utilizing the insightful outcomes of attacking DNP3, we have created a multi-model IDS able to perform cyberattack detection and classification tasks, presenting a DNN accuracy as high as 99.0%. Evidently, the utilization of network flows containing attributes specific to the protocol to be analyzed for machine learning and intrusion detection purposes, yields better metrics than using TCP/IP-based flows. Concluding, flow generators presenting protocol-specific attributes are the optimal solution for network flow-based IDS, as they target and describe directly the protocol's peculiarities and thus, present an attractive solution for enhancing the robustness against cyberattacks for ICS/SCADA.

VIII. Acknowledgement

The research leading to these results has received funding from the European Union's Horizon 2020 research and innovation program under grant agreement No 833955.

References

[1] V. R. Malik, K. Gobinath, S. Khadsare, A. Lakra, and S. V. Akulwar, "Security challenges in industry 4.0 scada systems – a digital forensic prospective," in *2021 International Conference on Artificial Intelligence and Computer Science Technology (ICAICST)*, pp. 229–233, 2021.

[2] A. T. S, B. C. Varghese, D. M. Baby, S. Antony, and R. Jose, "Plc automated water tap assembling and scrap recycling using scada representation," in *2021 Fourth International Conference on Microelectronics, Signals Systems (ICMSS)*, pp. 1–6, 2021.

[3] P. Bellini, D. Cenni, N. Mitolo, P. Nesi, G. Pantaleo, and M. Soderi, "High level control of chemical plant by industry 4.0 solutions," *Journal of Industrial Information Integration*, vol. 26, p. 100276, 2022.

[4] A. Üstündağ and C. Gencer, "Designing the clamp system with the emergency braking system in the trains by using plc and scada," in *2021 IEEE 11th Annual Computing and Communication Workshop and Conference (CCWC)*, pp. 1436–1441, 2021.

[5] C. Jayathilaka, "A literature review of cryptographic solutions used in scada to ensure its security," 04 2021.

[6] I. Erkek and E. Irmak, "Cyber security of internet connected ics/scada devices and services," in *2021 International Conference on Information Security and Cryptology (ISCTURKEY)*, pp. 75–80, 2021.

[7] M. Kermani, B. Adelmanesh, E. Shirdare, C. A. Sima, D. L. Carnì, and L. Martirano, "Intelligent energy management based on scada system in a real microgrid for smart building applications," *Renewable Energy*, vol. 171, pp. 1115–1127, 2021.

[8] V. L. Do, L. Fillatre, I. Nikiforov, and P. Willett, "Security of scada systems against cyber–physical attacks," *IEEE Aerospace and Electronic Systems Magazine*, vol. 32, no. 5, pp. 28–45, 2017.

[9] S. Ho, S. A. Jufout, K. Dajani, and M. Mozumdar, "A novel intrusion detection model for detecting known and innovative cyberattacks using convolutional neural network," *IEEE Open Journal of the Computer Society*, vol. 2, pp. 14–25, 2021.

[10] E. Sabev, G. Pavlova, R. Trifonov, K. Raynova, and G. Tsochev, "Analysis of practical cyberattack scenarios for wind farm scada systems," in *2021 International Conference Automatics and Informatics (ICAI)*, pp. 420–424, 2021.

[11] S. D. Antón, D. Fraunholz, C. Lipps, F. Pohl, M. Zimmermann, and H. D. Schotten, "Two decades of scada exploitation: A brief history," in *2017 IEEE Conference on Application, Information and Network Security (AINS)*, pp. 98–104, 2017.

[12] T. Akhtar and B. Gupta, "Analysing smart power grid against different cyber attacks on scada system," *International Journal of Innovative Computing and Applications*, vol. 12, no. 4, pp. 195–205, 2021.

[13] G. Yadav and K. Paul, "Architecture and security of scada systems: A review," *International Journal of Critical Infrastructure Protection*, vol. 34, p. 100433, 2021.

[14] V. Kelli, V. Argyriou, T. Lagkas, G. Fragulis, E. Grigoriou, and P. Sarigiannidis, "Ids for industrial applications: A federated learning approach with active personalization," *Sensors*, vol. 21, no. 20, 2021.

[15] A. Giani, G. Karsai, T. Roosta, A. Shah, B. Sinopoli, and J. Wiley, "A testbed for secure and robust scada systems," *SIGBED Rev.*, vol. 5, jul 2008.

[16] Q. S. Qassim, N. Jamil, M. N. Mahdi, Z. C. Cob, F. A. Rahim, and L. M. Sidek, "A short review: Issues and threats pertaining the security of scada systems," in *Advances in Cyber Security* (N. Abdullah, S. Manickam, and M. Anbar, eds.), (Singapore), pp. 230–247, Springer Singapore, 2021.

[17] D. Sebastian-Cardenas, S. Gourisetti, M. Mylrea, A. Moralez, G. Day, V. Tatireddy, C. Allwardt, R. Singh, R. Bishop, K. Kaur, J. Plummer, G. Raymond, B. Johnson, and A. Chawla, "Digital data provenance for the power grid based on a keyless infrastructure security solution," in *2021 Resilience Week (RWS)*, pp. 1–10, 2021.

[18] Y.-C. Chen, V. Mooney, and S. Grijalva, "Electricity grid cyber-physical security risk assessment using simulation of attack stages and physical impact," in *2020 IEEE Kansas Power and Energy Conference (KPEC)*, pp. 1–6, 2020.

[19] J. Kaniewski, H. Jahankhani, and S. Kendzierskyj, *Usability of the CBEST Framework for Protection of Supervisory Control and Acquisition Data Systems (SCADA) in the Energy Sector*, pp. 1–20. Cham: Springer International Publishing, 2021.

[20] R. Langner, "Stuxnet: Dissecting a cyberwarfare weapon," *IEEE Security Privacy*, vol. 9, no. 3, pp. 49–51, 2011.

[21] T. M. Chen and S. Abu-Nimeh, "Lessons from stuxnet," *Computer*, vol. 44, no. 4, pp. 91–93, 2011.

[22] A. Kleinmann, O. Amichay, A. Wool, D. Tenenbaum, O. Bar, and L. Lev, "Stealthy deception attacks against scada systems," in *Computer Security* (S. K. Katsikas, F. Cuppens, N. Cuppens, C. Lambrinoudakis, C. Kalloniatis, J. Mylopoulos, A. Antón, and S. Gritzalis, eds.), (Cham), pp. 93–109, Springer International Publishing, 2018.

[23] J. A. Crain and S. Bratus, "Bolt-on security extensions for industrial control system protocols: A case study of dnp3 sav5," *IEEE Security Privacy*, vol. 13, no. 3, pp. 74–79, 2015.

[24] A. Verma and V. Ranga, "Machine learning based intrusion detection systems for iot applications," *Wireless Personal Communications*, vol. 111, no. 4, p. 2287–2310, 2019.

[25] S. Kiran, U. V. Kumar, and T. M. Kumar, "A review of machine learning algorithms on iot applications," in *2020 International Conference on Smart Electronics and Communication (ICOSEC)*, pp. 330–334, 2020.

[26] B. Qian, J. Su, Z. Wen, D. N. Jha, Y. Li, Y. Guan, D. Puthal, P. James, R. Yang, A. Y. Zomaya, O. Rana, L. Wang, M. Koutny, and R. Ranjan, "Orchestrating the development lifecycle of machine learning-based iot applications: A taxonomy and survey," *ACM Comput. Surv.*, vol. 53, Aug. 2020.

[27] P. Radoglou Grammatikis, P. Sarigiannidis, G. Efstathopoulos, P. Karipidis, and A. Sarigiannidis, "Diderot: an intrusion detection and prevention system for dnp3-based scada systems," in *Proceedings of the 15th International Conference on Availability, Reliability and Security*, pp. 1–8, 08 2020.

[28] P. Radoglou-Grammatikis, P. Sarigiannidis, E. Iturbe, E. Rios, S. Martinez, A. Sarigiannidis, G. Eftathopoulos, I. Spyridis, A. Sesis, N. Vakakis, D. Tzovaras, E. Kafetzakis, I. Giannoulakis, M. Tzifas, A. Giannakoulias, M. Angelopoulos, and F. Ramos, "Spear siem: A security information and event management system for the smart grid," *Computer Networks*, p. 108008, 2021.

[29] I. Darwish, O. Igbe, O. Celebi, T. Saadawi, and J. Soryal, "Smart grid dnp3 vulnerability analysis and experimentation," in *2015 IEEE 2nd International Conference on Cyber Security and Cloud Computing*, 11 2015.

[30] H. Lin, A. Slagell, C. Di Martino, and R. Iyer, "Adapting bro into scada: building a specification-based intrusion detection system for the dnp3 protocol," in *Proceedings of the Eighth Annual Cyber Security and Information Intelligence Research Workshop*, 01 2013.

[31] C. Irvene, T. Shekari, D. Formby, and R. Beyah, "If i knew then what i know now: On reevaluating dnp3 security using power substation traffic," in *Proceedings of the Fifth Annual Industrial Control System Security (ICSS) Workshop*, pp. 48–59, 12 2019.

[32] R. Amoah, S. Camtepe, and E. Foo, "Securing dnp3 broadcast communications in scada systems," *IEEE Transactions on Industrial Informatics*, vol. 12, no. 4, pp. 1474–1485, 2016.

[33] S. Bagaria, S. B. Prabhakar, and Z. Saquib, "Flexi-dnp3: Flexible distributed network protocol version 3 (dnp3) for scada security," in *2011 International Conference on Recent Trends in Information Systems*, pp. 293–296, 2011.

[34] T. R. de Toledo and N. M. Torrisi, "Encrypted dnp3 traffic classification using supervised machine learning algorithms," *Machine Learning and Knowledge Extraction*, vol. 1, no. 1, pp. 384–399, 2019.

[35] E. Y. Song, G. J. FitzPatrick, K. B. Lee, and E. Griffor, "A methodology for modeling interoperability of smart sensors in smart grids," *IEEE Transactions on Smart Grid*, vol. 13, no. 1, pp. 555–563, 2022.

[36] S. East, J. Butts, M. Papa, and S. Shenoi, "A taxonomy of attacks on the dnp3 protocol," in *Critical Infrastructure Protection III* (C. Palmer and S. Shenoi, eds.), (Berlin, Heidelberg), pp. 67–81, Springer Berlin Heidelberg, 2009.

[37] A. LLC, "OpenDNP3," 2010.

[38] P. Biondi, "Scapy," 2003.

[39] A. Habibi Lashkari, "Cicflowmeter-v4.0 (formerly known as iscxflowmeter) is a network traffic bi-flow generator and analyser for anomaly detection. https://github.com/iscx/cicflowmeter," 08 2018.

[40] N. Brownlee, C. Mills, and G. Ruth, "Rfc2722: Traffic flow measurement: Architecture," 1999.