Attack Detection in Cyber-Physical Production Systems using the Deterministic Dendritic Cell Algorithm

Rui Pinto; Gil Gonçalves; Eduardo Tovar; Jerker Delsing

CISTER-TR-201011

Attack Detection in Cyber-Physical Production Systems using the Deterministic Dendritic Cell Algorithm

Rui Pinto; Gil Gonçalves; Eduardo Tovar; Jerker Delsing

CISTER Research Centre

Faculdade de Engenharia Universidade do Porto (FEUP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8321159

E-mail:

https://www.cister-labs.pt

Abstract

Cyber-Physical Production Systems (CPPS) are key enablers for industrial and economic growth. The introduction of the Internet of Things (IoT) in industrial processes represents a new revolution towards the Smart Manufacturing oncept and is usually designated as the 4 th Industrial Revolution. Despite the huge interest from the industry to innovate their production systems, in order to increase revenues at lower costs, the IoT concept is still immature and fuzzy, which increases security related risks in industrial systems. Facing this paradigm and, since CPPS have reached a level of complexity, where the human intervention for operation and control is becoming increasingly difficult, Smart Factories require autonomic methodologies for security management and self-healing. This paper presents an Intrusion Detection System (IDS) approach for CPPS, based on the deterministic Dendritic Cell Algorithm (dDCA). To evaluate the dDCA effectiveness, a testing dataset was generated, by implementing and injecting various attacks on a OPC UA based CPPS testbed. The results show that these attacks can be successfully detected using the dDCA.

Attack Detection in Cyber-Physical Production Systems using the Deterministic Dendritic Cell Algorithm

1st Rui Pinto
Research Center for Systems and Technologies
Faculty of Engineering, University of Porto
Porto, Portugal
rpinto@fe.up.pt

2nd Gil Gonçalves
Research Center for Systems and Technologies
Faculty of Engineering, University of Porto
Porto, Portugal
gil@fe.up.pt

3rd Eduardo Tovar
Research Centre in Real-Time and Embedded Computing Systems
Polytechnic of Porto - School of Engineering
Porto, Portugal
emt@isep.ipp.pt

4th Jerker Delsing
EISLAB
Luleå University of Technology
Luleå, Sweden
jerker.delsing@ltu.se

***Abstract*—Cyber-Physical Production Systems (CPPS) are key enablers for industrial and economic growth. The introduction of the Internet of Things (IoT) in industrial processes represents a new revolution towards the Smart Manufacturing oncept and is usually designated as the 4th Industrial Revolution. Despite the huge interest from the industry to innovate their production systems, in order to increase revenues at lower costs, the IoT concept is still immature and fuzzy, which increases security related risks in industrial systems. Facing this paradigm and, since CPPS have reached a level of complexity, where the human intervention for operation and control is becoming increasingly difficult, Smart Factories require autonomic methodologies for security management and self-healing. This paper presents an Intrusion Detection System (IDS) approach for CPPS, based on the deterministic Dendritic Cell Algorithm (dDCA). To evaluate the dDCA effectiveness, a testing dataset was generated, by implementing and injecting various attacks on a OPC UA based CPPS testbed. The results show that these attacks can be successfully detected using the dDCA.**

***Index Terms*—CPPS, OPC UA, IDS, DCA, AIS, Network Attacks**

I. Introduction

The proliferation of machine-to-machine (M2M) communication devices is leading to an enhanced role played by smart products and smart services, and to the emergence of global plant floors and autonomous factories. *Industry 4.0* is based on the idea of converging the real and the virtual worlds, by connecting every physical object to each other, in order to identify themselves to other devices and be able to communicate with each other. Also, based on Artificial Intelligent (AI) principles, it was emphasized the creation of intelligent machines, which make decisions and react to changes like human operators. All these ideas paved the way for the Smart Manufacturing concept.

The premises of the IoT applied in manufacturing processes is the connectivity of everything. Wireless communication technologies together with ubiquitous Internet access provide increasing interaction between machines, devices, sensors and people. Interconnection of physical devices and people in an industrial facility is the basis of joint collaboration, in order to reach common goals when performing tasks. In such complex systems, although openness in information exchange is desired, not every entity should learn every piece of information regarding others. There are several levels of entity clearance to access different types of information, which imply information privacy and entity access control. Also, trust and identification are very important, since malicious attacks may take over or impersonate entities, in order to access confidential information, driven by monetary and political interests. Privacy, access control, trust, identification and authentication are some aspects to be considered in an IoT cyber security solution.

Cyber-Physical Production Systems (CPPS) are becoming increasingly more susceptible to security vulnerabilities, specially with the introduction of IoT principles. Industrial control systems and operations should ensure increased availability, reliability and safety [1]. But, because of current security weaknesses, IoT may increase the vulnerability of these systems. The Maroochy Shire water plant control system was hacked in 2000, flooding the hotel grounds with raw sewage [2]. In 2008, a CIA report reveals that hackers have penetrated power systems in several regions outside the United States [3]. According to an Inspector General report sent to the Federal Aviation Administration in 2009 [4], hackers broken several times into the air traffic control mission-support systems of the U.S. Federal Aviation Administration. An Siemens plant-control system was also hacked by a virus in 2010 [5]. In 2011, hackers disrupt the Iran's nuclear system using the Stuxnet worm [6]. Hackers grounded ten LOT Polish Airlines flights at Warsaw's Chopin Airport in 2015, by attacking ground computer systems, affecting about 1,400 passengers [7]. More recently, in 2016 the Ukrainian power grid was attacked, where 30 power substations were taken down for six hours, affecting around 80,000 people [8].

The security of CPPS is crucial to the development and acceptance of the IoT technology [9]. One of the key issues nowadays that affects the full transformation of traditional manufacturing systems in CPPS is the lack of mature and complete security models and standards. M2M communication in CPPS is based on the Internet Protocol (IP), allowing for IP enabled devices to be monitored and controlled over the Internet. However, unlike the Internet, an IoT system has more complex communications, thus the resulting security issues represent more challenges than any other existing network system. Several M2M architectures were developed by different standard organizations, being the IEC 62541 a standard for Open Platform Communication Unified Architecture (OPC UA), which provides interoperability in process automation and service-oriented architecture (SOA) for industrial applications. While traditional Intrusion Detection Systems (IDS) can be modified and applied to CPPS intrusion detection, these techniques are limited when used to detect cyber-attacks on CPPS with high accuracy, since they usually lack sufficient capabilities to investigate network traffic based on unique proprietary protocols, such as OPC UA.

In the present paper, authors propose a deterministic Dendritic Cell Algorithm (dDCA) based technique for CPPS protection. This technique uses the danger theory model of the Human Immune System (HIS), which have been abstracted into an algorithm, which exists under a field of artificial intelligence called Artificial Immune Systems (AIS). This work intends to provide readers a description about an OPC UA attack detection technique based on the dDCA. Also, an evaluation of the dDCA is provided, by applying it to an OPC UA dataset. This dataset was generated while capturing OPC UA network traffic in a laboratory CPPS testbed, including the injection of security threats in the CPPS.

In the present paper, authors intend to provide readers a description about an OPC UA attack detection technique based on the deterministic Dendritic Cell Algorithm (dDCA). For this, a dataset was generated while capturing OPC UA network traffic in a laboratory CPPS testbed, including the injection of security threats in the CPPS. On the other side, the evaluation of the dDCA is provided, by applying it to an OPC UA dataset. This technique uses the danger theory model of the Human Immune System (HIS), which have been abstracted into an algorithm, which exists under a field of artificial intelligence called Artificial Immune Systems (AIS).

There is previous work, such as [10], regarding intrusion detection in industrial communication scenarios based on the dDCA. In this case the authors used the DNP3 protocol. However, there is a gap regarding using the same technique to analyse OPC UA traffic in related industrial scenarios. The main contributions of this work are: 1) generation of an additional CPPS network traffic dataset containing operational OPC UA traffic in an industrial M2M communication scenario; 2) evaluation of the dDCA as intrusion detection technique for anomaly-based detection of this industrial M2M network data.

The paper is organized in six more different sections. In Section II, a description of the main technologies used is provided, namely OPC UA as protocol for communications within CPPS and AIS as field with algorithms inspired on the HIS for intrusion detection. Also, it gives an overview of the state-of-the art regarding CPPS security approaches for intrusion detection and main datasets used to evaluate algorithms. Section III provides a detailed characterization of the dDCA algorithm. Section IV describes the CPPS tesbed experimental setup and the methodology for the OPC UA dataset creation. Section V provides discussion regarding the experimental results of using dDCA for intrusion detection in the generated dataset. Finally, Section VI concludes the paper, stating final remarks about the generated dataset and the used intrusion detection algorithm, and provides orientations for future work.

II. Background & Related Work

Considering the industrial context, there have been significant progress in the development of IDS, specially since the transformation of classical manufacturing to CPPS and 'openess' of communications and data access increases the potential attack surfaces for hackers and other cyber criminals. Classical manufacturing systems are already lacking proper security measures [11], leading to huge negative impacts on the performance of these systems. So, implementing complex new architectures based on CPPS expose it to much more vulnerabilities compared to the existing ones.

An IDS is a malicious activity monitoring software application, whose goal is simply to detect intrusions by collecting and analyzing system data. In a CPPS, currently, the OPC UA is one of the preferred protocols used. Actually, IDS do not detect intrusions but the evidence of an intrusion presence, after it occurs or while it is in progress. Also, the nature of an IDS is classified based on the used intrusion detection approach, e.g., anomaly-based or signature-based intrusion detection techniques.

Regarding CPPS, the best approach is using an anomaly-based intrusion detection technique, since these models are used to identify a bad behavior by comparison to a predefined intended behavior, which is considered to be the normal or baseline behavior. The assumption is that attacks are outliers in the data set, which differ always from the ordinary behavior of a regular data set. The main advantage is that previously unknown attacks can be detected. However, these models are characterized by their high false positive rate.

Next is provided detail regarding the OPC UA, as a standard protocol for communication in CPPS and AIS, as alternative algorithms within the AI field for anomaly-based intrusion detection.

A. OPC UA

OPC UA is an Industrial Automation protocol that integrates the functionality of the OPC Classic specification into one modular framework, whose specifications are being standardized as the IEC 62541 norm. It is widely used in industrial communication, offering a concept of secure data transfer and a multi-platform interconnection, which means that different equipments could have an OPC UA client and/or server to exchange data [12]. The OPC UA specification is a multi-part specification, organized in three groups, namely Core, Access Type and Utility specification parts.

Core capabilities define how data is modelled and exposed, along with the services used to access and manage it, ranging from Part 1 to Part 7, plus Part 14. Access type capabilities define how core capabilities and data models access different types of data, ranging from Part 8 to Part 11. Finally, utility capabilities describes discovery mechanisms and ways of aggregating data, including Parts 12 and 13.

Regarding Part 4 - Services, OPC UA follows the design paradigm of SOA, by defining generic services sets, such as discovery between Clients and Server's endpoints, secure communication channels, application-layer connection in the context of a session, node management in the address space, explore and filter the structure of the address space, readable/writable node's attributes, interfaces for invoking methods attached to an object, monitored attributes or events and subscription of messages for monitored items

In OPC UA everything is represented as a Node, which is composed of Attributes (properties with data values characterizing the Nodes) and References (relationships with other Nodes). Each Attribute has an integer id, name, description, data type and a mandatory/optional indicator. Clients can access Attribute values using the referred OPC UA services. A communication stack is used on client and server-side to encode and decode message requests and responses, which supports binary and XML formats. Regarding the transport layer, the OPC UA protocol can use both TCP and HTTP, being the TCP format the most widely adopted, since this format achieves high performance, as it offers less computational costs when it comes to coding/decoding.

Polge, Robert & Le Traon [13] presented the main OPC UA threats and the impact of those in OPC UA applications. The main identified threats are 1) Eavesdropping, 2) Session Hijacking, 3) Rogue Server, 4) Compromising User Credentials, 5) Message Spoofing, 6) Message Alteration, 7) Malformed Messages, 8) Message Flooding, 9) Message Replay, 10) Repudiation and 11) Server Profiling. As suggested previously, this standard has a security specification, so it implements satisfactory security mechanisms. However, according to the authors, it can not fill the lack of 'security-by-design' of legacy technologies. Also, Neu, Schiering & Zorzo [14] presented a detailed analysis of denial of service attacks that untrusted clients may perform against OPC UA servers.

B. Artificial Immune Systems

AIS are a category of biology-inspired computational methods that emerged in the 90s, bridging different areas such as immunology, computer science and engineering. AIS have been developed by computationally modeling immunologic processes know in biology, abstraction of the models into algorithms and implementation in the context of engineering.

The Human Immune System (HIS) is an immensely rich system and many immune processes are not well understood yet, generating discussion and little agreement among immunologists. This leads to a lack of clarity of the functioning of biologic immune processes and, regarding AIS practitioners, a difficulty to properly model a computational system. The vast majority of AIS have been inspired by the main innate and adaptive biologic immune mechanisms [15], which led to four main families of AIS algorithms: 1) Negative Selection algorithms [16]; 2) Artificial Immune Network Models [17], [18]; 3) Clonal Selection algorithms [19]; 4) Danger Theory based algorithms [20], where the most known algorithms are the Dendritic Cell algorithm [21] and the Tolk-like Receptor algorithm [22].

According to Dasgupta & Nino [23], there are three possible applications of AIS in computer security, namely virus detection, network security and intrusion detection systems. Detecting anomalies in computer systems filled with different virus resembles the problem of discrimination between self and non-self, tackled in the Negative Selection algorithms. Critical information is threatened when networks are exposed to malicious actions. Negative Selection algorithms can be used to monitor network traffic, in order to identify uncommon patterns caused by malicious attacks. Intrusion detection systems are not used to prevent attacks, but rather to detect attacks that already took place. These systems can be modeled using several immune algorithms, including the ones based on the Danger Theory.

Many AIS approaches have been used to develop IDS over the years. IDS based on AIS are generally used as anomaly based detection models. Rubio, Alcaraz, Roman & Lopez [24] characterize IDS for industrial ecosystems in the last years. These IDS include both commercially available and novel detection mechanisms and architectures developed in the academia. Bortoli [25] proposes in his thesis an IDS based solution to monitor malicious computer attacks on OPC UA, based on a plug-in for the dynamic Bro Network IDS to support OPC UA based protocol. However, there are very few AIS based approaches for IDS in industrial scenarios.

According to Antón, Gundall, Fraunholz & Schotten [26], being based on AIS or not, there are challenges in implementing industrial IDS, since there are hardly any datasets publicly available that can be used to evaluate intrusion detection algorithms. The KDD Cup99 [27], NSL-KDD [28] and UNSW-NB15 [29] are widely used datasets for benchmark in IDS. However, these datasets do not contain typical network traffic based on industrial protocols.

Antón, Gundall, Fraunholz & Schotten [26] propose three datasets containing operational OPC UA traffic of an industrial process, in which several attacks have been introduced. On a different work [30], they used the generated dataset to evaluate the performance of Support-vector machine and Random Forest algorithms. Igbe, Darwish & Saadawi [10] also propose a dataset for a Smart Grid scenario, containing Distributed Network Protocol 3 (DNP3) traffic. This dataset was used to evaluate the effectiveness of an implementation of the dDCA to detect cyber-attacks targeting Smart Grids that utilize DNP3.

III. The Dendritic Cell Algorithm

According to classical immune theories, the HIS is composed of a group of cells that main function is to detect foreign and potentially dangerous pathogens and respond adequately, while ignoring harmless substances and own body cells. Its main capability is to distinguish between self and non-self cells. The non-self cells are further categorized, in order to induce an appropriate type of response, while providing tolerance for self cells. Regarding the Danger Theory, Matzinger [31] explained that there are several flaws within the explanations of how an immune response is initiated in classical theories, such as Negative Selection algorithms. She states that, on contrary of those, the 'foreignness' of a pathogen is not the important feature that triggers a response, and 'selfness' in no guarantee of tolerance. The immune response is initiated by a co-stimulatory signal from specific dendritic cells, also known as Antigen Presenting Cells (APCs). Injured cells, such as those exposed to pathogens, emit danger/alarm signals that activate APCs. These signals should not be emitted by healthy cells or by cells undergoing normal physiological death. Once activated, they provide a co-stimulatory signal to exhibit an immune response in the danger zone around the injured cell. Fig. 1 illustrates the Danger Theory functionality.

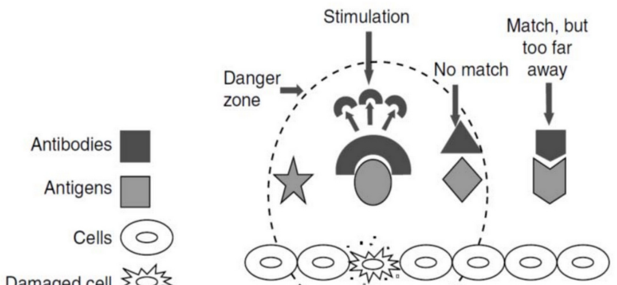

Fig. 1. Illustration of the Danger Theory. [23]

The Dendritic Cell Algorithm (DCA) is an abstraction of the Danger Theory, based on the known functionalities of the biologic dendritic cells of the inate immune system. The function of dendritic cells [32], first identified by Steinman & Cohn [33], in these type of algorithms is inspired by the innate immunity. Dendritic cells are derived from the bone marrow and circulate in the blood to be distributed in different tissues. In the tissue, when dendritic cells are stimulated by antigens and danger signals, they mature and differentiate accordingly to the specific tissue. After capturing the antigens, they migrate to local lymphoid nodes, where they present captured antigens to lymphocytes, inducing an immune response. Immature dendritic cells are not capable of inducing an immune response, which is influenced by the level of maturation of dendritic cells. The level of maturation of a dendritic cell is facilitated by the detection of signals within the tissue, namely danger signals (caused by damage to tissue cells), pathogenic associated molecular patterns (PAMPs), which can be seen as pre-defined bacterial signatures, safe signals (caused by regulated cell deaths) and inflammatory signals (general tissue distress).

The DCA is a population-based algorithm inspired by the interaction between dendritic cells (agents in the system) and their environment, i.e., the DCA has the ability to combine multiple input signals to assess the current context of the environment. The original implementation of the DCA was a highly stochastic algorithm, due to the number of random-based elements. Hence, in order to analyze better its performance, eliminating most of stochastic parameters in the algorithm, Greensmith et al. [34] propose the deterministic version of the DCA (dDCA). The correlation between the context and the antigen is used as the basis of anomaly detection in this algorithm. Input signals to the dDCA are pre-categorized based on their origin, and can be:

- Danger signals (Ds) - The presence of danger signals may indicate an anomalous situation. However, the probability of an anomaly is higher than under normal circumstances.
- Safe signals (Ss) - This signal increases in value in conjunction with observed normal behavior. Hence, the presence of safe signals almost certainly indicates that no anomalies are present.

Regarding the output signals, there are the co-stimulatory signal csm and the context output value k, which represents the concentration of semi-mature and mature signals. These output signal concentrations are derived based on the input signals, using (1) and (2).

$$csm = Ds + Ss \tag{1}$$

$$k = Ds - 2Ss \tag{2}$$

In (1), the csm output signal is used to determine when a DC has exceeded its lifespan and hence, ready for migration. The k output signal is used to determine the context of the DC. If the value of k is greater than 0, then the DC is assigned a context value of 1, meaning that its collected antigens may be anomalous. Else, if the value of k is less than 0, the DC is assigned a context of 0 indicating that its collected antigen is likely to be normal. Pseudocode for the dDCA is given in Algorithm 1.

Despite having less random-based variables, there are still some parameters to be specified for the dDCA, namely the

Algorithm 1 dDCA Pseudocode.

```
Input $numDCs, numIter, At$
Generate $Ags, Ds, Ss$
for $i \leq numDCs$ do
    Initialize dc()
end for
for $j \leq numIter$ do
    for all $Ags$ do
        Update cumulative $csm, k$
        for all $dc$ do
            Update dc profile()
        end for
        if $lifespan \leq 0$ then
            Migrate $dc$
            reset $dc$
            if $k \geq 0$ then
                $dc$ migrates as 'Mature'
            else
                $dc$ migrates as 'Semi-Mature'
            end if
        end if
    end for
end for
for all $Ags$ do
    Classify Ag()
    for all Migrated $dc$ do
        Count sampled Ags
        if $dc$ migrated as 'Mature' then
            Count cells
        end if
    end for
    $mcav \leftarrow numCells/numAntigen$
    if $mcav \geq At$ then
        $context \leftarrow 1$
    else
        $context \leftarrow 0$
    end if
end for
```

$numDCs$ - number of DCs in the population, $numIter$ - mumber of iterations of the algorithm, and At - anomaly threshold used in the classification phase.

The dDCA starts by generating the antigens and input signals Ds and Ss, according to the type of data available. The antigen refers to the data tuple to classify and the input signals represent the features of that data tuple. The dDCA proceeds by initializing every DC in an Imature state, which includes initializing the DC profile, such as the output signals csm and k as 0, the lifespan (expected time to live of the DC) as a random value according to a normal distribution, and an empty pool of sampled antigens.

Then, every antigen generated will be sampled by every DC, in a phase known as DC Migration. In this phase, the cumulative output signals are calculated, using (1) and (2), and every DC profile is updated, namely incrementing each own csm and k with the cumulative output signal results, the lifespan is decreased by csm and the pool of sampled antigens is updated accordingly.

During this process, if the lifespan of a given DC reaches 0, then the DC migrates, which means it leaves the Imature state. In order to know if the DC goes to a Mature or Semi-Mature state, its k value is analysed. If k is positive, then the DC migrates as Mature, otherwise as Semi-Mature. The DC Migration is repeated along all specified iterations.

In the end, the final phase of the dDCA is the actual antigen classification. For each antigen generated, it is counted the number of times the respective antigen was sampled by a migrated DC. Of those, it is also counted the number of DCs that are in a Mature state. Finally, the $mcav$ is calculated, where the number of Mature cells is divided by the number of sampled antigens. If the $mcav$ is greater then the specified anomaly threshold (At), then the antigen sampling context is 1, which means that the respective antigen is likely to be anomalous. Otherwise, the context is 0, which means the antigen is likely to be normal.

IV. Experimental Setup

The generation of the dataset containing OPC UA traffic was possible due to the setup and execution of a laboratory CPPS testbed. This CPPS uses OPC UA standard for horizontal and vertical communications, and was designed and deployed using the Dinasore tool [35]. Regarding the CPPS testbed setup, it consists on seven nodes in the network, as represented in Fig 2.

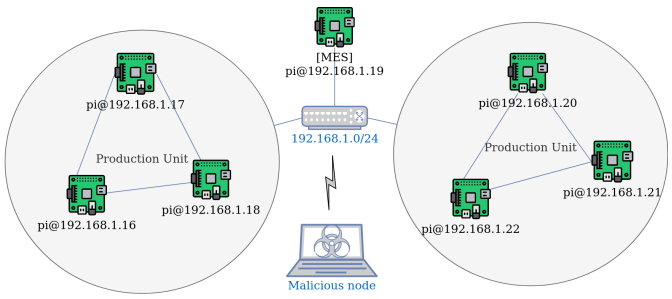

Fig. 2. CPPS Experimental Setup.

Each network node consist on a Raspberry Pi device, running the Python FreeOpcUa implementation [36]. In this configuration, there are two production units, each one containing three devices, and one node representing a Manufacturing Execution System (MES). Each device implements both OPC UA server and client, where the server publish to a OPC UA variable updates regarding sensor readings and the client subscribes all OPC UA variables from all other devices in the same production unit. On the other side, the MES only implements the OPC UA client, which subscribes all OPC UA variables from all devices in both production units. Also, connected to this network is an attack node, as it is assumed that the attacker already gained access to the CPPS network.

After setting up the CPPS testbed, a Python script that implements Tshark was used to capture OPC UA packets and export this traffic to a csv file format dataset. This traffic includes both normal and anomalous behaviour. Anomalous behaviour is achieved with the malicious node, which injects attacks into the CPPS network, targeting one or more legit device nodes and the MES. The attacks injected by the malicious activities are:

- Denial of Service(DoS), which have the purpose to eliminate device's capability to perform its expected function. This type of attacks may target not only at the physical level, but also the network level, by interfering with the network protocol, preventing measured data to arrive at the destination, or cause battery exhaustion, because of packet re-transmissions. DoS attacks are characterized by collisions and exhaustion at the link layer, black holes and misdirection at the network layer, malicious flooding and time synchronization disruption at the transport layer. This specific DoS occurred in a HELLO Flood attack fashion, where the malicious node broadcasts or send to a specific device HELLO messages at a high rate.
- Eavesdropping or Man-in-the-middle (MITM), which target the privacy of CPPS, by monitoring and gathering information, through remote access mechanisms. An attacker may monitor the information passively, by listening to the transmission medium and understanding unprotected messages, or can do an active traffic analysis, where the attacker initiates specific processes by sending control data as queries, in order to identify sensitive information. In this case, the malicious node eavesdrop unprotected messages exchanged between devices and MES.
- Impersonation or Spoofing, which consist on the impersonation of node, where the malicious node (bigger and powerful machine) is used as a server or gateway. The attacker may steal the received information and also create new messages or modify the received ones, in order to deliver different content compared to the one received in the first place.

A. Dataset Creation

To perform the attacks mentioned, a Python script is used, which implements the Scapy [37] module for packet sniffing, injection and modification. Regarding the dataset generation, another Python script, that implements Tshark (in this case Pyshark [38], was used to capture only OPC UA packets and export this traffic to a csv file format dataset. Actually, the OPC UA packets are converted to bidirectional communication flows, which are characterized by the following 32 features:

- src_ip - Source IP address;
- src_port - Source port;
- dst_ip - Destination IP address;
- dst_port - Destination port;
- $flags$ - TCP flag status;
- $pktTotalCount$ - Total packet count;
- $octetTotalCount$ - Total packet size;
- avg_ps - Average packet size;
- $proto$ - Protocol;
- $service$ - OPC UA service call type;
- $service_errors$ - Number of service errors in OPC UA request responses;
- $status_errors$ - Number of status errors in OPC UA request responses;
- msg_size - OPC UA message transport size;
- min_msg_size - Minimum OPC UA message size;
- $flowStart$ - Timestamp of flow start;
- $flowEnd$ - Timestamp of flow end;
- $flowDuration$ - Flow duration in seconds;
- $avg_flowDuration$ - Average flow duration in seconds;
- $flowInterval$ - Time interval between flows in seconds;
- $count$ - Number of connections to the same destination host as the current connection in the past two seconds;
- srv_count - Number of connections to the same port number as the current connection in the past two seconds;
- $same_srv_rate$ - The percentage of connections that were to the same port number, among the connections aggregated in $Count$;
- $dst_host_same_src_port_rate$ - The percentage of connections that were to the same source port, among the connections having the same port number;
- $f_pktTotalCount$ - Total forward packets count;
- $f_octetTotalCount$ - Total forward packets size;
- $f_flowStart$ - Timestamp of first forward packet start;
- f_rate - Rate at which forward packets are transmitted;
- $b_pktTotalCount$ - Total backwards packets count;
- $b_octetTotalCount$ - Total backwards packets size;
- $b_flowStart$ - Timestamp of first backwards packet start;
- $label$ - Binary label classification;
- $multi_label$ - Multi classification labeling.

The generated dataset has 33.567 normal instances, 74.013 DoS attack instances, 50 impersonation attack instances, and 7 MITM attack instances. This gives a total of 107.634 instances. Since the dDCA is a binary classification technique, all attacks were grouped into one class (anomaly) and the rest of the instances belong to the normal class. The dataset [39] is available for research purposes and can be accessed from the link [1].

B. Data Pre-Processing

In order to apply the dDCA on the dataset generated, the data IDs of each tuple were used to form an antigen instance, while a pre-processed subset of attributes were used to form the signals input signals. Despite having 32 features, there was the need to perform dimension reduction, in order to reduce the number of features under consideration by obtaining a set of principal features of the dataset, which represent best the CPPS tesbed functionality.

Firstly, the feature values were normalized to fall between the range of 0 - 100%. Then, for dimension reduction, the Pearson correlation coefficient (PCC) was calculated for each feature, by measuring the linear correlation between the feature itself and the $label$. According to the Cauchy–Schwarz

[1]https://digi2-feup.github.io/OPCUADataset/

inequality, the PCC has a value between +1 and -1, where 1 is total positive linear correlation, 0 is no linear correlation, and -1 is total negative linear correlation.

In this case, only features with correlations (absolute values) higher then 0.5 were selected. Of those, the most positive linear correlated features with $label$ are the $count$ and srv_count. On the other side, the most negative linear correlated features are $flags$, $pktTotalCount$, $octetTotalCount$, avg_ps, msg_size, min_msg_size, $f_octetTotalCount$, f_rate, $b_pktTotalCount$, $b_octetTotalCount$ and $b_flowStart$. The rest of the features are very poor correlated, since the absolute correlations are lower then 0.5.

For Ds and Ss signal selection, the guidelines provided by Greensmith were followed [21] in her PhD thesis, regarding the specification of the original DCA. Since the Ss is a 'stronger' signal then the Ds, i.e., it is a more reliable indicator of safe context then Ds is indicative of danger context, Greensmith suggests to use the attribute that has the lowest standard deviation in the attribute subset to derive the safe signal, making it the "most certain" signal. In this case, the feature with lowest standard deviation was the $b_pktTotalCount$. The rest of the features are combined to derive the danger signal.

The Ss signal derivation process is: 1. Select a suitable attribute - in this case, $b_pktTotalCount$ is chosen; 2. Calculate the median of all the selected attribute's values across both classes of data (normal and anomaly); 3. For each attribute value, if value greater then the mean, then the safe value equals to the absolute distance between mean and value. Otherwise, it is 0.

For the Ds signal derivation process: 1. Mean values across all values for each attribute are calculated just for the normal class (not including anomaly class attribute's values); 2. Take each attribute value in turn and calculate the absolute distance between the attribute values and the respective means; 3. The mean of all calculated absolute distance values is used to form a single value for the Ds signal.

V. Results and Discussion

Since this is a classification problem, it was used the area under the receiver operating characteristic Curve (AUC - ROC) curve for dDCA performance measurement (ability for binary classification), by plotting the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. Fig 3 represents the ROC curve for the dDCA setup of $numDCs$ - 100, $numIter$ - 10, and At - 0.688 (derived anomaly threshold by calculating the rate of anomaly tuples within the whole dataset. The calculated AUC is 0.99.

Based on the AUC - ROC curve for performance measurement, the calculated AUC is very near to 1, which means the proposed algorithm has a good measure of separability, i.e., it is very capable of distinguishing between the normal and anomaly classes. The authors believe that the high AUC is due to have so many highly correlated features with the $label$, since from the feature subset, the lowest absolute PCC calculated was of 0.82, which represents a very strong correlation. So, this means that the input signals Ds and Ss define very

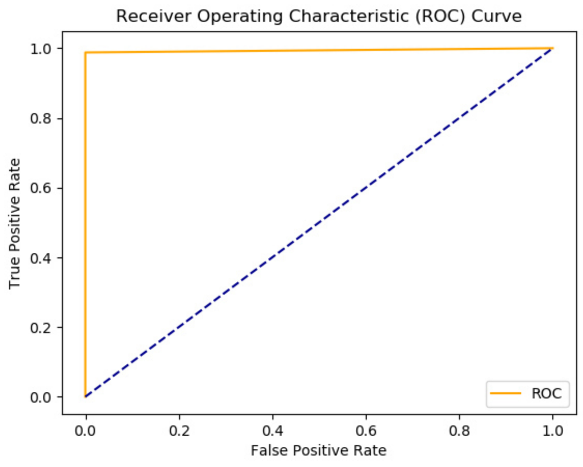

Fig. 3. ROC curve.

well the actual context of each sampled antigen. In order to understand if the PCC influence greatly the performance of the dDCA in the dataset, the same test is performed, but instead of using the highly correlated features, it is used features with PCC lower then 0.5. In this case, poor positive correlated features are $dst_host_same_src_port_rate$ and $same_srv_rate$, while the poor negative correlated features are src_port, $flowEnd$, $flowStart$, $f_flowStart$, $flowInterval$, $avg_flowDuration$, $flowDuration$, dst_port and $f_pktTotalCount$.

According to the guidelines provided by Greensmith for Ss and Ds signal calculation, the feature with lowest standard deviation was the $f_pktTotalCount$. The rest of the features of the poorly correlated subset are combined to derive the danger signal. With the same dDCA setup, i.e., $numDCs$ - 100, $numIter$ - 10, and At - 0.688 , the calculated AUC is 0.5. This means that, with the wrong features, the dDCA has the same capability of distinguishing between the normal and anomaly classes as a random selection.

VI. Conclusion and Future Work

This paper analyzed various threats and vulnerabilities in a CPPS using OPC UA as standard communication. An OPC UA dataset was generated, containing normal traffic within the CPPS and anomalies, which resulted from the security penetration tests, injecting in the network multiple attack scenarios including DoS, MITM and Spoofing. In this research, the dDCA was used to detect the cyber-attacks. The generated dataset was used to evaluate the DCA's effectiveness, and it was shown that the dDCA performed very well, not only in detecting the attacks, but also clearly distinct those from the normal system behaviour, as the calculated AUC is 0.99. The dDCA classification performance is greatly dependent on the feature selection process, which will be used to generate the danger and safe signals. From the tests performed, one can conclude that features poorly correlated with the classification label have a very negative impact in the dDCA classification, and highly correlated features result in an almost perfect classification.

For future work, it will be explored and analysed how the dDCA can be used for classification of data streams in a real-time detection scenario, instead of using a dataset. This mainly concerns the dDCA's great dependency of data pre-processing and selection. Because anomaly detection is typically an online and unsupervised nature problem, robustness of dDCA may be achieved by embedding techniques for unsupervised signal generation, which don't rely on application dependent data engineering performed manually by human experts.

ACKNOWLEDGMENT

This research was supported by the project "INDTECH 4.0 - Novas Tecnologias para Fabricação Inteligente", which embodies a comprehensive response towards the development and implementation of new production systems, embedding advanced production technologies, that will equip the manufacturing industry to meet the challenges and opportunities of the 4th industrial revolution. The authors would like to thank all partners for their support and discussions that contributed to these results.

REFERENCES

[1] L. Wigle, "Securing critical infrastructure," http://www.darkreading.com/partner-perspectives/intel/securing-critical-infrastructure-/a/d-id/1321123?, June 2015.

[2] S. Mustard, "Security of distributed control systems: The concern increases," *Computing & Control Engineering Journal*, vol. 16, no. 6, pp. 19–25, 2005.

[3] K. O'Connell, "Cia report: cyber extortionists attacked foreign power grid, disrupting delivery," *Internet Business Law Services*, 2008.

[4] E. Mills, "Report: Hackers broke into FAA air traffic control systems," http://www.cnet.com/news/report-hackers-broke-into-faa-air-traffic-control-systems/, may 2009.

[5] J. Finkle, "Virus targets Siemens industrial control systems," https://www.reuters.com/article/us-siemens-microsoft-virus-idUSTRE66I5VX20100719, july 2010.

[6] J. P. Farwell and R. Rohozinski, "Stuxnet and the future of cyber war," *Survival*, vol. 53, no. 1, pp. 23–40, 2011.

[7] L. Constantin, "Cyberattack grounds planes in Poland," http://www.computerworld.com/article/2938486/security/cyberattack-grounds-planes-in-poland.html, June 2015.

[8] B. News, "Hackers caused power cut in western Ukraine - US," http://www.bbc.com/news/technology-35297464, January 2016.

[9] L. Antao, R. Pinto, J. Reis, and G. Gonçalves, "Requirements for testing and validating the industrial internet of things," in *2018 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 2018, pp. 110–115.

[10] O. Igbe, I. Darwish, and T. Saadawi, "Deterministic dendritic cell algorithm application to smart grid cyber-attack detection," in *2017 IEEE 4th International Conference on Cyber Security and Cloud Computing (CSCloud)*. IEEE, 2017, pp. 199–204.

[11] R. Tsang, "Cyberthreats, vulnerabilities and attacks on scada networks," *University of California, Berkeley, Working Paper, http://gspp. berkeley. edu/iths/Tsang_SCADA% 20Attacks. pdf (as of Dec. 28, 2011)*, 2010.

[12] M. Müller, E. Wings, and L. Bergmann, "Developing open source cyber-physical systems for service-oriented architectures using opc ua," in *2017 IEEE 15th international conference on industrial informatics (INDIN)*. IEEE, 2017, pp. 83–88.

[13] J. Polge, J. Robert, and Y. Le Traon, "Assessing the impact of attacks on opc-ua applications in the industry 4.0 era," in *2019 16th IEEE Annual Consumer Communications & Networking Conference (CCNC)*. IEEE, 2019, pp. 1–6.

[14] C. V. Neu, I. Schiering, and A. Zorzo, "Simulating and detecting attacks of untrusted clients in opc ua networks," in *Proceedings of the Third Central European Cybersecurity Conference*, 2019, pp. 1–6.

[15] D. Dasgupta, S. Yu, and F. Nino, "Recent advances in artificial immune systems: models and applications," *Applied Soft Computing*, vol. 11, no. 2, pp. 1574–1587, 2011.

[16] S. Forrest, A. S. Perelson, L. Allen, R. Cherukuri *et al.*, "Self-nonself discrimination in a computer," in *IEEE Symposium on Security and Privacy*. Oakland, 1994, pp. 202–212.

[17] J. Timmis, M. Neal, and J. Hunt, "An artificial immune system for data analysis," *Biosystems*, vol. 55, no. 1, pp. 143–150, 2000.

[18] T. Knight and J. Timmis, "Aine: An immunological approach to data mining," in *Proceedings of the 2001 IEEE international conference on data mining*. IEEE Computer Society, 2001, pp. 297–304.

[19] L. N. De Castro and F. J. Von Zuben, "The clonal selection algorithm with engineering applications," in *Proceedings of GECCO*, vol. 2000, 2000, pp. 36–39.

[20] U. Aickelin and S. Cayzer, "The danger theory and its application to artificial immune systems," *arXiv preprint arXiv:0801.3549*, 2008.

[21] J. Greensmith, "The dendritic cell algorithm," Ph.D. dissertation, Citeseer, 2007.

[22] J. P. Twycross, "Integrated innate and adaptive artificial immune systems applied to process anomaly detection," Ph.D. dissertation, University of Nottingham Nottingham, UK, 2007.

[23] D. Dasgupta and F. Nino, *Immunological computation: theory and applications*. CRC press, 2008.

[24] J. E. Rubio, C. Alcaraz, R. Roman, and J. Lopez, "Analysis of intrusion detection systems in industrial ecosystems." in *SECRYPT*, 2017, pp. 116–128.

[25] T. Bortoli, "A framework for network intrusion detection on open platform communications unified architecture," Master's thesis, Technische Universität, 2017.

[26] S. D. Antón, M. Gundall, D. Fraunholz, and H. D. Schotten, "Implementing scada scenarios and introducing attacks to obtain training data for intrusion detection methods," in *ICCWS 2019 14th International Conference on Cyber Warfare and Security: ICCWS 2019*. Academic Conferences and publishing limited, 2019, p. 56.

[27] S. D. Bay, D. Kibler, M. J. Pazzani, and P. Smyth, "The uci kdd archive of large data sets for data mining research and experimentation," *ACM SIGKDD explorations newsletter*, vol. 2, no. 2, pp. 81–85, 2000.

[28] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the kdd cup 99 data set," in *2009 IEEE symposium on computational intelligence for security and defense applications*. IEEE, 2009, pp. 1–6.

[29] N. Moustafa and J. Slay, "Unsw-nb15: a comprehensive data set for network intrusion detection systems (unsw-nb15 network data set)," in *2015 military communications and information systems conference (MilCIS)*. IEEE, 2015, pp. 1–6.

[30] S. D. D. Anton, S. Sinha, and H. D. Schotten, "Anomaly-based intrusion detection in industrial data with svm and random forests," in *2019 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. IEEE, 2019, pp. 1–6.

[31] P. Matzinger, "The danger model: a renewed sense of self," *Science*, vol. 296, no. 5566, pp. 301–305, 2002.

[32] J. Greensmith, U. Aickelin, and J. Twycross, "Articulation and clarification of the dendritic cell algorithm," in *International Conference on Artificial Immune Systems*. Springer, 2006, pp. 404–417.

[33] R. M. Steinman and Z. A. Cohn, "Identification of a novel cell type in peripheral lymphoid organs of mice i. morphology, quantitation, tissue distribution," *The Journal of experimental medicine*, vol. 137, no. 5, pp. 1142–1162, 1973.

[34] J. Greensmith and U. Aickelin, "The deterministic dendritic cell algorithm," in *International Conference on Artificial Immune Systems*. Springer, 2008, pp. 291–302.

[35] DIGI2-FEUP, "Dynamic INtelligent Architecture for Software and MOdular REconfiguration," https://digi2-feup.github.io/dinasore-ua/, 2020, [Online; accessed 17-March-2020].

[36] FreeOpcUa, "FreeOpcUa: Open Source C++ and Python OPC-UA Server and Client Libraries and Tools," https://freeopcua.github.io/, 2020, [Online; accessed 17-March-2020].

[37] P. BIONDI, "Network packet manipulation with scapy," 2007.

[38] KimiNewt, "Python packet parser using wireshark's tshark," https://kiminewt.github.io/pyshark/, 2020, [Online; accessed 17-March-2020].

[39] R. Pinto, "M2m using opc ua," 2020. [Online]. Available: http://dx.doi.org/10.21227/ychv-6c68