Future Generation Computer Systems 179 (2026) 108314

Contents lists available at ScienceDirect

Future Generation Computer Systems

journal homepage: www.elsevier.com/locate/fgcs

Dispatching advanced and adaptive intrusion responses for IIoT-based systems

Jacobo Elicha *, Javier Lopez

Computer Science Department, University of Malaga, Campus de Teatinos s/n, 29071, Malaga, Andalusia, Spain

ARTICLE INFO

Keywords:
Advanced recovery
Industrial cyber-security
Intrusion response
Attack mitigation
IIoT
Situational awareness

ABSTRACT

The ever-increasing number of cyber-attacks poses a serious challenge to incident response teams. Recent cyber-attacks, such as the attack against energy distribution companies in Ukraine, highlight the disruption which can be caused and its consequences. More than 53 % of recorded incidents targeted essential entities, which heavily rely in Industrial Internet of Things (IIoT) devices, according to ENISA. Despite the amount of work in Cyber Threat Intelligence (CTI) and Intrusion Detection Systems (IDSs), automated response systems have been avoided in connected industrial environments mainly due to the criticality of the underlying assets, where a misstep has the potential to result in the disruption of critical processes. This paper therefore presents an Early and Adaptive Automated Intrusion Response Service for industrial environments, named EAIRS, which combines several techniques, including expert systems and reinforcement learning, to classify and mitigate anomalies detected by IDSs. The incidents EAIRS is designed to face range from network to host-based attacks. This paper provides the architecture for the described approach and the evaluation of a proof-of-concept implementation on an experimental testbed.

1. Introduction

With the advent of Industry 5.0, many critical applications have adopted the most disruptive information technologies in operational processes in order to modernize infrastructures and improve the value chain and its productivity [1]. However, they have not only increased the attack surface in these scenarios, but also placed critical infrastructures in the sights of cyber-criminals [2]. This trend is evidenced with the increase in cyber-attacks to these assets, which now account for 53 % of incidents reported according to ENISA [3]. The impact of incidents on these scenarios, such as the attack against energy distribution companies in Ukraine [4], remark the need to protect them and prevent disruptions. The literature has focused on Intrusion Detection Systems (IDSs), which aim to detect a wide range of cyber-incidents [5]. On the other hand, the response to incidents in industrial environments, nowadays normally composed of the limited Industrial Internet of Things (IIoT), has been a manual task carried out by teams of cybersecurity experts, who can be overwhelmed by the growing number of incidents and the amount of information provided by the detection systems for each of the alerts. Analysing each cyber-incident is also a time-demanding task which added to the high number of alerts that teams may receive result in delayed responses to cyber-attacks, with no mitigation in the meantime. This delay in response increases the impact that an incident may have.

The approach proposed in this study is designed to enable a rapid response to cyber incidents, effectively mitigating attacks or at least minimizing their impact on the underlying infrastructure. In doing so, it provides critical time for further investigation and facilitates proactive engagement of adversaries employing previously unidentified tactics. Another major concern considered is preventing the response system from causing disruptions, which in other cases could narrow the application scope. The techniques considered to achieve these capabilities include Expert Systems and Reinforcement Learning (hereinafter, ES and RL respectively). The term knowledge-based systems refers to information systems that use symbolic representations of human knowledge. Expert systems are specialized knowledge-based systems designed to provide solutions to specific problems or provide advice within a certain domain [6]. On the other hand, RL is a goal-directed and decision-making learning approach, in which the system learns through direct interaction with the environment [7]. Each of the techniques incorporates features that support the early and adaptive response to cyber-incidents, which the proposed approach aims to provide.

The paper is structured as follows. Section 2 explores the literature on autonomous response systems, their characteristics, capabilities and

* Corresponding author.
E-mail addresses: jacoboeg@uma.es (J. Elicha), jlopez@uma.es (J. Lopez).

https://doi.org/10.1016/j.future.2025.108314
Received 30 September 2025; Received in revised form 14 November 2025; Accepted 11 December 2025
Available online 18 December 2025

limitations. Section 3 introduces the architecture of the proposed Early and Adaptive Intrusion Response System (EAIRS), emphasizing the two main modules that constitute the core of the logic of the system. In addition, this section also presents the operating and training algorithms of the system. The modelling of the threats considered in this paper, the experimental environment implemented to evaluate the proposal, and the results are discussed in Section 4. Finally, Section 5 presents the conclusions obtained in this work.

2. Related work

Industrial systems are integral to modern industrial infrastructures, making their resilience against cyber threats a critical area of study. Research on self-protecting Industrial Cyber-Physical Systems (ICPSs) has largely concentrated on IDSs, leaving a notable gap in the literature regarding intrusion response mechanisms. Although the concept of intrusion response systems is not new, their development and integration within ICPSs have not kept pace with the advancements seen in detection technologies. This shortfall highlights an urgent need for innovative strategies that not only identify intrusions but also provide adaptive and effective responses. By bridging this gap, future work can enhance the overall security posture of ICPSs and ensure a more robust defense against evolving cyber threats.

Several studies have explored the development of Intrusion Response Systems (IRS) rooted in Expert Systems. For example, as detailed in [8], an automated response system is designed with the primary objective of minimizing response times. In this approach, logs from multiple cloud service providers are collected using Logstash as the event log shipper and subsequently stored in a centralized database. These logs are then filtered and normalized into a common format, emphasizing the extraction of the most critical information. Subsequently, a rule-based correlation algorithm is applied to detect recurring attributes across various events, facilitating the rapid identification of potential threats. Once anomalies are identified, the 'Active Responder' component executes predetermined countermeasures to effectively neutralize the threats. The performance of this system is quantified using metrics such as Mean Time to Detect (MTTD) and Mean Time to Respond (MTTR), achieving an MTTD of 30 s and an MTTR ranging from 25 to 90 minutes. These promising results highlight the potential of expert system-based IRS architectures to enhance the responsiveness and overall security of cloud environments.

Other studies have delved deeper into the integration of different analytical approaches. In [9], the authors introduce a novel methodology named Automatic Incident Responder (AIR), which leverages graph analytics to enhance countermeasure selection. AIR initiates its process by generating a hypothesis through a multinomial naive Bayes classifier applied to a comprehensive knowledge graph. This graph comprises nodes representing historical attacks, detailed descriptions of these attacks, candidate countermeasures along with their respective descriptions, and the intricate relationships among these elements. Following hypothesis generation, AIR employs a greedy heuristic to identify an optimal subset of defensive techniques that can effectively mitigate the hypothesized threat while minimizing techniques usage. The methodology shows significant potential, achieving an average precision of 84 %. In [10], authors propose an adaptive system which detects and reacts to cyber-attacks in ICPS environments. This approach leverages quantitative Hierarchical Risk Correlation Tree to model attack scenarios and establish the response by solving a Competitive Markov Decision Process (CMPD) which models reactions. The system is also assessed in a testbed, proving its performance and significant improvement in comparison with existing solutions. Authors in [11] also model the reciprocal interactions between the attacker and system through CMPD. This approach also leverages quantitative Hierarchical Risk Correlation Tree (HRCT) to model the paths an attacker may follow to achieve certain objectives, and measures the financial risk that cyberattacks pose to the CPS assets. The proposed system has been assessed in an ICPS testbed, improving traditional intrusion response systems by 43.61 %.

In [12], the authors introduce autonomous response agents that leverage model-free deep RL. Specifically, the agent is trained using the Double Deep Q Network (DDQN) algorithm, an off-policy, model-free, online learning method that employs a value-based approach with Deep Neural Networks (DNN). The response system operates in conjunction with an IDS, which is responsible for the initial detection phase and subsequently alerts the response agent. This approach defines a state space that integrates both physical parameters (such as temperature) and cyber indicators (including IDS alerts), fostering a comprehensive situational awareness. The action space is structured as vectors that encompass a range of atomic actions, including dropping attack packets, blocking network traffic, modifying firewall rules, and restricting access, among others. The reward function is formulated to consider both production quantity and quality, as well as the proximity of the system to a hazardous state. However, this function remains unnormalized, exhibiting a range from -3000 to 3000 in the test environment. The experimental testbed is implemented on a cyber-physical system modeled as a continuous stirred tank reactor, providing a realistic industrial scenario. Empirical results demonstrate that the proposed approach achieves a 59.5 % improvement over traditional rule-based systems, proving the suitability of RL to respond to cyber-attacks with a very interesting approach.

The work [13] utilises Game Theory to address cyber-attacks in ICPSs. Their work quantifies the probability of a successful attack by leveraging vulnerability metrics from the Common Vulnerability Scoring System (CVSS). The attacker-defender interactions are modelled as a two-player stochastic game. In this framework, it is assumed that the defender has complete visibility into the attacker's actions and system states, a premise supported by the integration of an IDS. Concurrently, the attacker is able to probe the system, gaining insights into the defender's countermeasures. The attacker's action set comprises atomic actions against the system associated with the vulnerabilities, for instance buffer overflow, and the no-operation action. The defender can also take this no-operation action, and security countermeasures, such as installing patches. In formulating the game, each player's objective is to maximize its payoff, which reflects their respective mission goals. For the attacker, the payoff function integrates several components: the duration required to carry out an attack, the delay in the defender's response, and the recovery time of the compromised device, which includes the restoration of any affected physical processes. The defender's payoff is designed to be the negative of the attacker's, defining the adversarial, zero-sum nature of the interaction. To derive optimal strategies, the authors apply a Q-Learning model, a RL technique, and uses the defined payoff as the reward function. The proposed approach is validated on a chemical reactor testbed modelled after a simplified Tennessee-Eastman process control system, demonstrating its applicability and potential effectiveness in securing real-world ICPS environments.

Table 1 summarises the literature reviewed and compares it with the approach suggested in this work. Following the advances in the literature, this paper proposes a system that exploits the benefits of two of the best performing techniques in the field, combining an expert agent with a neural network-based RL agent. In this way, there is the advantage of early response to known and considered threats in the expert agent, whereas the reinforcement learning agent adapts to unprecedented or unknown situations for the system. This aspect is also represented in the table, where the three main states of a response (*proactive*, *reactive* and *adaptive* - as stated in [14]) are specially covered by EAIRS, which makes it a potentially interesting defence service for critical systems. By "*early*" (or proactive) we mean that the system is capable of reacting before an exploit fully completes; by "*reactive*", the system reacts after the exploit; and by "*adaptive*", the system is capable of adapting a response to unknown situations. The latter also means that if the system must be capable of responding with the same level of aggressiveness as the attacker in order to maintain an adequate confrontation.

Table 1
Related work comparison.

Work	Approach	Early	Reactive	Adaptive
[8]	Rule-based engine	•	•	
[9]	Graph analytics	•	•	
[10]	Competitive Markov Decision Process		•	•
[11]	CMDP and HRCT		•	•
[12]	Deep RL using DNN		•	•
[13]	Game Theory with RL		•	
EAIRS	Expert agent and NN with RL	•	•	•

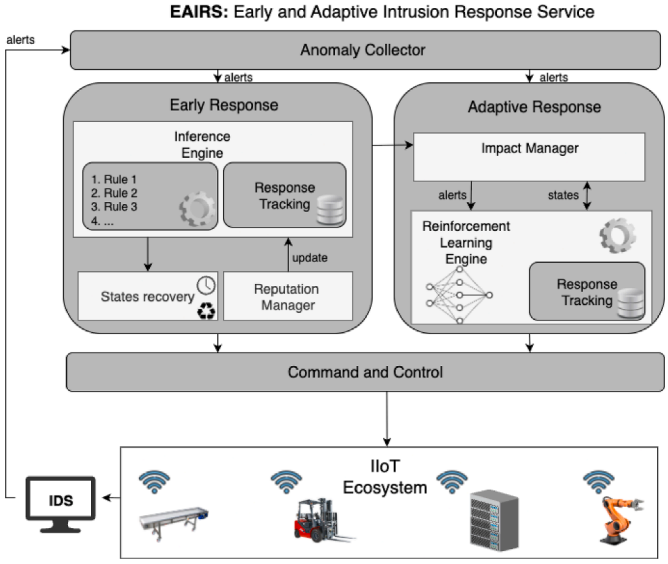

Fig. 1. General EAIRS architecture and its integrated modules.

3. Immune system architecture - EAIRS

These three main aspects ("*early*", "*reactive*" and "*adaptive*") are present in EAIRS architecture, illustrated in Fig. 1. The proposed system is composed of two principal modules: the *Early Response* (ER) and the *Adaptive Response* (AR). These modules work collaboratively to (i) deal with both known and unknown cyber-attacks, and (ii) intensify the system's responsiveness to various types of attack and attacker profiles. Initially, the ER module engages to counteract known threats; however, if it encounters scenarios that exceed its capabilities, it activates the AR module. Together, these modules orchestrate a comprehensive immune response.

As depicted in Fig. 1, the ER and AR modules rely on the *Anomaly Collector* (AC), the function of which is to collect alerts and distribute them. The manner in which these alerts are distributed varies according to the state of the response system. Namely, the response system can handle: (i) alerts that are exclusively delivered to the ER to determine an action; and (ii) alerts that are forward both to the ER, in order to carry out the previous task, and to the AR to compute certain learning parameters required to model the agent's behavior in the response process. These modules must discriminate the following two Response Scenarios (from now on as RSc):

- *RSc-A*: when the anomaly is already known by the ER module and the system has certain prior experience or knowledge, the response is provided by this module. This knowledge is kept up-to-date using the most complete threat intelligence repositories, such as MITRE ATT&CK for industrial control systems repository. This scenario demonstrates the early and reactive capabilities of the proposed system, proving the rapid response that the system implements against known threats.
- *RSc-B*: when an unidentified anomaly reaches the ER module, it is delivered to the *Impact Manager* (IM) submodule, located within the AR module. The IM appends a value that represents the criticality of the observed node, and sends the alert to the *Reinforcement Learning Engine*, which will select the most suitable measure to mitigate the alert. Both modules, IM and RL Engine, will be analyzed in detail below. The adaptive capacities of the system are leveraged to solve this scenario.

The system's set of countermeasures is strategically designed to neutralize attacks and mitigate threats while ensuring minimal disruption to operational processes, safeguarding the integrity of the business model and its value chain. To formalize these EAIRS responses in their respective anomaly scenarios, ASc-A and ASc-B, the following five relevant actions are considered: $A = \{a_1, a_2, a_3, a_4, a_5\}$ such that:

- *Void action* - $[a_1]$: no threat mitigation actions are undertaken. The assumption of inaction is key to avoiding noise and ignoring false positives from the IDS.
- *Network output reconfiguration* - $[a_2]$: limit outbound traffic. This action only allows some known destinations.
- *Network reconfiguration* - $[a_3]$: limit outbound and inbound traffic, allowing just some known destinations in both directions.
- *Restrict credentials* - $[a_4]$: changes the remote access configuration, disables the password authentication allowing just certificate authentication. Also, freezes all current sessions.
- *Process killing* - $[a_5]$: kills every process that is not found on a whitelist kept within the nodes. This list is updated periodically, when the device is in a normal and safe state.

Note that these five types of action have been initially established to demonstrate the usefulness of EAIRS in the experimental studies detailed below, however, the approach is potentially scalable to add new response and neutralization configurations. The dispatch of these actions and their automatic orchestration in the three fundamental response states (proactive, reactive and adaptive) are explicitly outlined in the following subsections.

3.1. RSc-A: ER module for reaction and adaptivity

To streamline the reactive decision-making, an ES based on defined rules, established as the set $R = \{r_1, r_2, \ldots, r_n\}$, and actions such as those detailed in A, is considered for the ER module. As depicted in Fig. 1, this ES comprises an *Inference Engine* (IE), which counts with the set R to classify the anomaly received according to the deviation of the hardware resource usage values from their usual value. Information assessed by this module per device, which includes hardware resource usage and network traffic, as been selected so that gathering it in constraint devices, such as IIoT, does not impact the host performance. This type of classification is key to discriminate actions, enabling the IE to either (i) determine the most suitable action or (ii) send the alert to the AR module.

In case the threat is identified, the associated action is sent to the *Command and Control* (CC) module which will eventually perform the determined action. In order to monitor the level of success of all the actions taken, it is necessary to consider the role of another submodule within the ER called *Response Tracking* (RT). RT stores in a database every measure the ER takes, and also, in a buffer, the last measure applied in each node. This latter mechanism keeps track of the actions executed in order to escalate the threat mitigation toughness if an anomaly persists, aiming to prevent persistent threats, which is a trait present in the rules. This consideration implies the adaptive nature of the system. On the other hand, the database is used to hold the system responsible for the actions it has taken.

The *Reputation Manager* (RM) submodule, also included in the ER, maintains a reputation register for the actions performed. The RM assesses the trust of the system in each action taken through a numeric

value, $\tau_i \in [0, 100]$ with $i \in [1, 5]$. When the system is started, all mitigation measures have maximum reputation, $\tau_i = 100\ \forall i \in [1, 5]$, if an alert is received from a node in which the ER actuated recently, the threat mitigation measure carried out will have its reputation diminished. In addition, the RM module will periodically increase the reputation of those measures that have not been taken recently. This register, along with a reputation threshold, δ, prevents the ER from carrying out measures that are failing repeatedly; meaning this that if it infers an action but its reputation is below the reputational threshold, $\tau_i < \delta$, the alert will be sent to the AR module instead. Complete threat mitigation also entails state recovery; therefore, ER also incorporates a *State Recovery* (SR) submodule. The SR service, following the action executed by the ER, and after a carefully determined waiting period, restores the device to its prior configuration.

The operations of the ER, as described in this subsection, are presented in the Algorithm 1.

Algorithm 1 ER algorithm.

```
t_{k-1}, a_i ← actions_buffer(s_k)
if t_{k-1} − t_k ≤ κ then
    τ_i ← τ_i − γ
end if
a_j ← es_expert_engine(s_k)
if τ_j < γ then
    send_alert_to_ar(s_k)
else
    send_action_to_command_&_control(a_j)
    actions_buffer.save(s_k, a_j, timestamp)
end if
```

3.2. RSc-B: AR module for proactivity and adaptivity

The AR module utilizes a RL-based agent, which aims to classify the unknown anomalies received from the previously presented ER module. This classification process is based on the same set of factors considered in the ER module. When applying RL to solve a problem, the main challenges are modeling states, actions, and the reward function, being the latter a critical part of the RL problem formulation, since it will determine the agent's behavior and performance. More specifically, the computation of the "states" within the proposed system depends on how the agent is fed with the anomalies detected by the *Anomaly Detector*, which includes information related to hardware usage, such as the actual usage and the mean usage. This data provides significant insight into the status of the node and gathering and sending it does not pose an overload for the end device, which is a critical aspect in constraint devices, such as IIoT. In this approach, a one-dimensional tensor, with each column representing each of the alert features, was the solution that best suited this approach. The features that characterize the alert are scaled in order to prevent them from gaining excessive relevance and to ease its convergence. Thus, the input alert received is transformed into a tensor of ratios for each feature, such ratios can be described as $\frac{|x-\bar{x}|}{\bar{x}}$.

Defining the set of "actions" is also an important task, but it can also be one of the drawbacks when considering RL for threat mitigation, because the space of action is discrete and finite. Indeed, defining a finite set of actions to effectively mitigate threats, while acknowledging that not all threats in general can be addressed, poses a significant challenge. This limitation hinders the practical implementation of this approach by others. Since RL is based on trial-error, a large action set would cause the model to take an excessive amount of time to converge, thus considering too many actions may not be the solution. For this reason, only a finite set of actions, A, has been predefined above, not only to carry out experiments and subsequent discussions, but also to deal with anomalies once they have been detected. Moreover, each action has an associated cost, $c_i \in [0, 1)\ \forall i \in [1, 5]$, which represents the impact that measure has on the regular operation of the device. This will be taken into account when computing the reward for the self-learning and adaptation. This cost prevents the agent from choosing the most aggressive response regardless of the situation, as it has a detrimental effect on some functionalities of the device. The *Response Tracking* module stores in a database every action the agent takes, along with a timestamp, to provide traceability of the actions carried out by the agent on each device.

Despite the importance of this task, defining the "reward" function has been over-simplified or even avoided in previous works in this area. As an example, in [15] the reward function implemented in this RL used for DDoS mitigation is, in case there is a congested network link, R = 1, otherwise, R = 0. Such an oversimplification may limit the agent's insight into the state of the network and the result of the actions executed. As the reward function determines the agent's behavior, it seemed reasonable to develop two different rewards and validate them through the experiment described in the following section, selecting the most suitable one. When designing the reward functions, several variables were considered to be involved in the response to a cyber-incident, such variables include not only the response's cost and the node's criticality, but also the efficacy of such action. Thus, the reward functions proposed depend on whether an alert is received short time after a mitigation measure is applied or not, in an attempt to measure the mitigation's effectiveness. As mentioned above, the damage an attack can cause to an industrial infrastructure also depends on its particular target, therefore the reward should also vary depending on the node. To achieve such discrimination, the reward functions should take into account the node's criticality. As a result of this consideration, the reward functions described below were modeled to:

$$r_{i,k,j} = \theta * r^0_{i,k,j} \qquad i = 1, 2$$

where θ is the aforementioned criticality of the node and $r^0_{i,k,j}$ are defined as follows,

$$r^0_{1,k,j} = \begin{cases} 1 - c_j & \text{no alert is received} \\ (\varrho_{k+1} - 1) - c_j & \text{an alert is received} \end{cases}$$

$$r^0_{2,k,j} = \begin{cases} 1 - c_j & \text{no alert is received} \\ (\varrho_{k+1} - \varrho_k) - c_j & \text{an alert is received} \end{cases}$$

being c_j the cost of action a_j, and ϱ_k the anomaly value of the alert set by the *Anomaly Detector* at step k.

The difference between them lies in the scenario in which an alert is received. As can be observed, in r^0_1, the reward when an alert is received is calculated as $(\varrho_{k+1} - 1) - c_j$, the unitary subtraction is due to the fact that the anomalies are modeled as numeric, with $\varrho \in [0, 1]$, where a healthy node is represented with $\varrho = 1$, and on the contrary, $\varrho = 0$ represents the most severe anomaly. This arithmetic operation quantifies the deviation of the node from its optimal state. In contrast, r^0_2 accounts for the improvements in the condition of the node, since it is derived from the difference between the previous and updated anomaly values. Therefore, even if the action does not fully restore the status of the node but leads to partial improvement, the reward remains positive.

This RL agent implements the Neural Fitted Q iteration (NFQ) algorithm [16] with,

$$target = r_{k,j} + \gamma \max_{a_j \in A} Q_k(s, a_j)$$

where s denotes the state where the transition starts, A the action space, $r_{k,j}$ the reward at step k for action a_j, and γ a discounting factor. NFQ model is combined with Adam optimizer [17] and Mean Squared Error (MSE) as the stochastic objective function to optimize agent behavior. The training algorithm described is summarised in Algorithm 2.

These two scenarios are also depicted in Algorithm 3, where a set of actions (a_i) is established according to the state (s_i).

Algorithm 2 RL training.

```
if s_k is known then
    a_j ← es_expert_engine(s_k)
else
    S.reset()
    repeat
        a_j ← ar_agent().actor(s_k)
        s_{k+1}, r_k ← S.step(a_j)
        if k mod D = 0 then
            agent.fit(buffer.get_batch(D))
            buffer.clean()
        else
            buffer.save(s_k, a_j, r_k, s_{k+1})
        end if
    until k ≤ max_steps
end if
```

Algorithm 3 Modified algorithm conversion.

```
1:  if s_k is known then                                   ▷ RSc-A
2:      t_{k−1}, a_i ← actions_buffer(s_k)
3:      if t_{k−1} − t_k ≤ κ then
4:          τ_i ← τ_i − γ
5:      end if
6:      a_j ← es_expert_engine(s_k)
7:      send_action_to_command_&_control(a_j)
8:      actions_buffer.save(s_k, a_j, timestamp)
9:  else                                                   ▷ RSc-B
10:     repeat
11:         a_j ← ar_agent().actor(s_k)
12:         s_{k+1}, r_k ← S.step(a_j)
13:     until r_k ≥ min_reward ∨ k ≥ max_iter
14: end if
```

4. Threat model, testbed and experiments

To assess the effectiveness of the approach formally described in the previous section, the experimental design based on a model of threats and initial assumptions, in addition to the discussions of the results obtained, are presented below.

4.1. Threat modelling and assumptions

The threat model is based on: (i) the susceptibility of ICPS to targeted attacks at essential industrial operations, and (ii) the nature of EAIRS to derive deviations related to the computational (the percentage of CPU used), storage (the percentage of memory used) and communication (the number of packets and bytes sent and received) overheads. In turn, it is widely assumed that the attacker has the ability to gain (probably with limitations) access to the system and with the capacity to execute a set of possible attacks within the ICPS. These attacks correspond to:

- **Reconnaissance** - [t_1]: This threat considers an attacker with access to the network, comprising to major steps. As an outsider, the network topology and node's information are unknown to the adversary. Thus, it begins with (a) the enumeration of assets within the network. This provides the malicious actor with a list of devices from which the target will be selected. Afterward, (b) a deeper scan on the target device is performed. The latter action enriches the attacker's knowledge on the victim's attack surface, crucial information for the success of the next steps. Both scans are performed leveraging specific tools for enumeration.
- **Brute Force** - [t_2]: The adversary aims to gain access to the target device, thus it forces one of the services operating on the victim. This phase has been tailored to the targeted organization with keywords that increase the success of the process. The core of this attack consists of the attacker using tools designed to perform brute force attacks with industrial-specific wordlists against the SSH service.
- **Researching the victim** - [t_3]: Once inside the target device, the attacker delves into the filesystem and running processes. With this action, the adversary gathers information which will be leveraged in the following steps. Authors prepared a tailored bash script to perform this step, which included (but not limited to): (a) directory enumeration, (b) lateral movement to other users, (c) privilege escalation attempts.
- **External interactions** - [t_4]: Another action considered by the attacker is the connection with external servers both to upload stolen information and to download malicious software to install on the compromised device. Again, a tailored script was implemented to execute this attack. Furthermore, two medium-sized files were prepared (~800 Mb) and placed in the victim and a remote server. This attack comprises (a) sending the file in the host to a remote server, and (b) in case the previous step was successful, downloading the second file from a different remote server.

These attacks, and their phases, are designed to be part of a set of tests performed in an industrial testbed, composed of assets in their default configuration (including IIoT and other devices commonly present in critical infrastructures), emulating a regular industrial environment. Furthermore, these devices perform their operations communicating through several protocols that can be found in most industrial infrastructures, such as Modbus TCP. To perform these attacks, the malicious actor is placed within the testbed's network, with a Kali Linux machine.

4.2. IIoT testbed and experimental design

The environment designed to assess the proposed system is an industrial testbed that emulates an energy generation scenario. This setting comprises multiple devices, including Programmable Logic Controllers (PLCs), such as SIMATIC-S7-1200 (which uses Profinet) and an Open Source PLC based on Raspberry Pi 3 (leveraging Modbus TCP); Supervisory Control and Data Acquisition systems (SCADAs); Field Devices (FDs) based on Intel Galileo Gen1, RevPi Core 3 and Raspberry Pi; and 'low-level' sensors and actuators, including industrial sensors (communicating through IO-Link, WirelessHART and ISA100.11a) and TelosB sensors (using 6LoWPAN over IEEE 802.15.4); among others. To accomplish the operations involved in this scenario, the devices also communicate through industrial protocols including OPC UA and Modbus TCP.

All devices within the network will be monitored with an IDS responsible for providing alerts to the response system. The IDS chosen for these experiments is the one proposed by the authors in [18]. This selection is motivated by its lightweight nature and high precision, with the former being particularly crucial for devices operating under resource constraints. The alerts generated by the IDS will not only be sent directly to EAIRS, but will also be stored in a database. Together with the alerts, in this database, the states of the nodes will be stored so that they can be consulted by the response system. The AR queries this database during training. In addition to the devices that build the energy generation scenario, to carry out the experiments presented, a malicious machine is deployed on the same network to perform the attacks on the devices.

4.3. Results and discussions

Threats previously presented were executed against the IIoT testbed with EAIRS deployed, to finally evaluate the behaviour of the system. Fig. 2 illustrates EAIRS when confronting t_1 and t_2. As can be observed, initially the device is carrying out its operations and presents some network traffic. During this phase, the adversary deploys a reconnaissance scanning against the network, and then another scan targets the device monitored in this figure. Both scans can be seen as peaks in the received

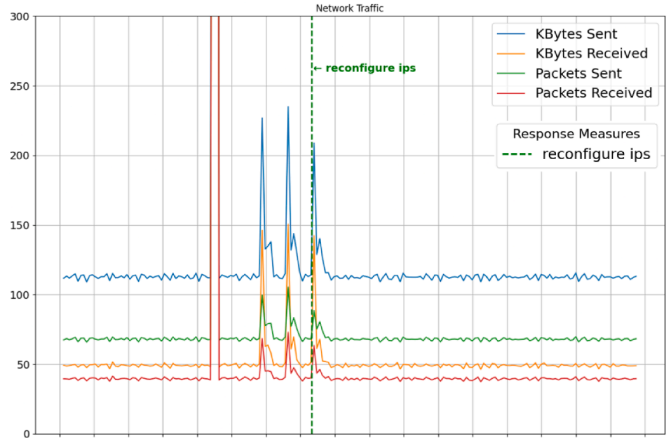

Fig. 2. t_1 and t_2 results.

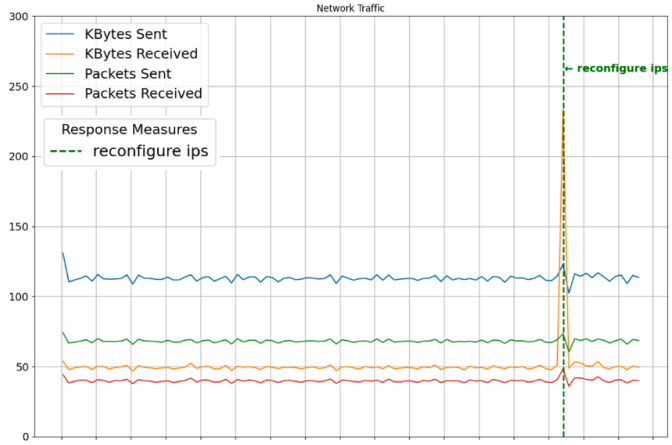

Fig. 3. Initial actions of t_4 results.

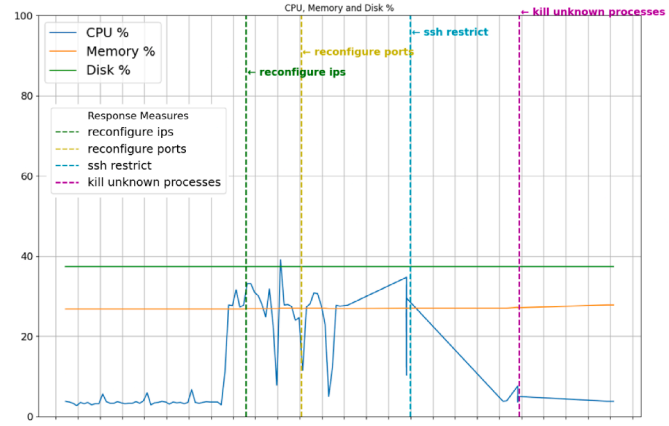

Fig. 4. Final actions of t_4 results.

network traffic. Nevertheless, the IDS deployed does not notice these actions since they are accomplished in short periods, thus EAIRS does not consider responses. Once the adversary has selected the target and port, it begins with the Brute Force attack against the SSH service. The increase in the received network traffic, which can be observed in the same figure, is noticed by the IDS, which sends several alerts to EAIRS. The vertical green-dashed lines represent the measures taken by EAIRS to perform the action a_3. Following this action, the device starts recovering from the attack and returns to its healthy state. Despite the IDS missing the initial scans, the system provided an *early* response, stopping the exploit shortly after it started its activity and preventing it from succeeding or fully completing its activity. EAIRS achieved ∼3 minutes time-to-response in this scenario, and a ∼4 minutes mean-time-to-response after several runs of this scenario. Furthermore, in this case time-to-response also means time-to-recover, since the device returns to its normal state after the initial response action. On the other hand (despite being unsuccessful, the previous step is considered accomplished and the attacker gains credentials), through t_3 the malicious actor gains further knowledge regarding the device. Since these actions require minimal computation power, the IDS deployed does not detect the activity and consequently EAIRS does not provide response.

The system's response to t_4, depicted in Figs. 3 and 4, starts with the device in its normal state. The adversary logs into the device is unperceived, since the credentials used are valid the whole system considers it a legitimate access. The adversary aims to download malicious software to be executed in the victim asset. Promptly, EAIRS receives the alerts sent by the IDS, which presents an increase in incoming network traffic and CPU usage. EAIRS blocks communications with the external servers from which the attacker is downloading the software using action a_3. After EAIRS intervention, the device returns to its usual state. EAIRS best time-to-response in t_4 was ∼0.5 minutes, as in the previous scenario, it matches time-to-recover since with the initial action the device is restored. On the other hand, multiple executions revealed mean-time-to-response rose to ∼0.8 minutes.

Following, EAIRS response to information exfiltration as part of t_4 has been studied. Again, the attacker is supposed to have gain credentials in the previous steps and is logged in. Following, the attacker starts sending this information to an external address. This behavior increases device network traffic and CPU usage, as can also be seen in Fig. 4, which triggers the IDS. The response system considers these alerts and, consequently, deploys a_2. However, due to the severity of the incident and since it persists, it also executes a_3. However, the attacker notices how the firewall is reconfigured and the operations are halted. Thus, the adversary starts to counter the actions of EAIRS and changes the firewall configurations as the system responds. EAIRS notices how the attack persists and the attacker continues to extract information so it considers a_4 and a_5, terminating all the running processes which are not strictly necessary for the device's operations and preventing the attacker from logging in again with credentials. As a result, the device starts to return to its previous state, depicted in Fig. 4. In this part of t_4 it is found that EAIRS presents also ∼0.5 minutes time-to-response when providing its initial response. However, this action is not sufficient and the attack persists, thus time-to-recover rises to ∼7 minutes in this experiment.

5. Conclusion

In this study, an early and adaptive automated response service, combining expert systems and neural networks leveraging reinforcement learning, is proposed for connected industrial cyber-physical control systems and networks based on industrial Internet of Things. The suggested architecture has been implemented and deployed on an experimental testbed and validated through several experiments. These experiments proved EAIRS capabilities, outlining its rapid response, which could be critical in an industrial cyber-physical system and its autonomy to neutralize the threat over time. Although this system is autonomous and capable of responding to cyberincidents, it is designed to provide the initial response, which can, and in many cases must, be complemented with the actions of a human expert, which would benefit from EAIRS by gaining time to analyze the incident and respond. This collaborative approach positions this system within the latest industrial paradigms.

However, there are still challenges and limitations to be faced, future work includes considering higher-dimension states, providing a more holistic view of the environment, and studying mitigation actions that specifically target the adversary.

CRediT authorship contribution statement

Jacobo Elicha: Writing – original draft, Software, Investigation, Conceptualization; **Javier Lopez:** Writing – review & editing, Investigation.

Data availability

The authors do not have permission to share data.

Declaration of interests

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests:

Jacobo Elicha reports financial support was provided by Ministerio de Ciencia, Innovación y Universidades, Spain. Authors would like to thank the company S2Grupo for pro- viding useful comments and feedback for improvements of the paper. The work has been supported by the national project SEGRES (EXP-00131359/MIG-20201041) funded by the CDTI as part of the Ministerio de Ciencia, Innovación y Universidades If there are other authors, they declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

Authors would like to thank the company S2Grupo for providing useful comments and feedback for improvements of the paper. The work has been partially supported by the national project SEGRES (EXP-00131359/MIG-20201041) funded by the CDTI as part of the Ministerio de Ciencia, Innovación y Universidades. Also, the work was partially supported by project SYNAPSE funded by the European Union under Grant 101,120,853 (HORIZON-CL3-2022-CS-01). Funding for open access charge: Universidad de Málaga / CBUA.

References

[1] S. Vaidya, P. Ambad, S. Bhosle, Industry 4.0 - a glimpse, Procedia Manuf. 20 (2018) 233–238. https://doi.org/10.1016/j.promfg.2018.02.034

[2] M. Lehto, Cyber-Attacks Against Critical Infrastructure, Springer International Publishing, Cham, 2022, pp. 3–42. https://doi.org/10.1007/978-3-030-91293-2_1.

[3] ENISA European Union Agency for Cybersecurity, ENISA Threat Landscape Report, Technical Report, 2025. Publication date: October 1, 2025, https://www.enisa.europa.eu/publications/enisa-threat-landscape-2025.

[4] CISA America's Cyber Defense Agency, Cyber-Attack Against Ukrainian Critical Infrastructure, Technical Report, 2021. Publication date: July 20, 2021, https://www.cisa.gov/news-events/ics-alerts/ir-alert-h-16-056-01.

[5] J.E. Rubio, C. Alcaraz, R. Roman, J. Lopez, Current cyber-defense trends in industrial control systems 87 (2019) 101561. https://doi.org/10.1016/j.cose.2019.06.015

[6] P.J.F. Lucas, L.C. Van Der Gaag, Principles of expert systems, Addison Wesley Longman, 1991.

[7] R.S. Sutton, A.G. Barto, et al., Reinforcement learning, J. Cogn. Neurosci. 11 (1) (1999) 126–134.

[8] K.A. Torkura, M.I.H. Sukmana, F. Cheng, C. Meinel, Slingshot - Automated threat detection and incident response in multi cloud storage systems, in: 2019 IEEE 18Th International Symposium on Network Computing and Applications (NCA), 2019, pp. 1–5. https://doi.org/10.1109/NCA.2019.8935040

[9] F.K. Kaiser, L.J. Andris, T.F. Tennig, J.M. Iser, M. Wiens, F. Schultmann, Cyber threat intelligence enabled automated attack incident response, in: 2022 3Rd International Conference on Next Generation Computing Applications (NextComp), 2022, pp. 1–6. https://doi.org/10.1109/NextComp55567.2022.9932254

[10] H.A. Kholidy, Autonomous mitigation of cyber risks in the cyber-physical systems, Future Generation Computer Systems 115 (2021) 171–187. https://doi.org/10.1016/j.future.2020.09.002

[11] H.A. Kholidy, Autonomous mitigation of cyber risks in the cyber-physical systems, Future Gener. Comput. Syst. 115 (2021) 171–187. https://doi.org/10.1016/j.future.2020.09.002

[12] M.S. Bashendy, A. Tantawy, A. Erradi, Autonomous response agent for cyber physical system attacks: a model-Free deep reinforcement learning approach (drl-Irs), Available at SSRN 4716080 (2024).

[13] K. Huang, C. Zhou, Y. Qin, W. Tu, A game-Theoretic approach to cross-Layer security decision-Making in industrial cyber-Physical systems, IEEE Trans. Ind. Electron. 67 (3) (2020) 2371–2379. https://doi.org/10.1109/TIE.2019.2907451

[14] C. Alcaraz, J. Lopez, Wide-Area situational awareness for critical infrastructure protection, IEEE Comput. 46 (4) (2013) 30–37. https://doi.org/10.1109/MC.2013.72

[15] L.S.R. Sampaio, P.H.A. Faustini, A.S. Silva, L.Z. Granville, A. Schaeffer-Filho, Using NFV and reinforcement learning for anomalies detection and mitigation in SDN, in: 2018 IEEE Symposium on Computers and Communications (ISCC), 2018, pp. 00432–00437. https://doi.org/10.1109/ISCC.2018.8538614

[16] M. Riedmiller, Neural fitted q iteration – First experiences with a data efficient neural reinforcement learning method, in: J. Gama, R. Camacho, P.B. Brazdil, A.M. Jorge, L. Torgo (Eds.), Machine Learning: ECML 2005, Springer Berlin Heidelberg, Berlin, Heidelberg, 2005, pp. 317–328.

[17] D.P. Kingma, J. Ba, Adam: A Method for Stochastic Optimization, 2017, 1412.6980

[18] A. Garcia, C. Alcaraz, J. Lopez, MAS Para la convergencia de opiniones y detección de anomalías en sistemas ciberfísicos distribuidos, in: VIII Jornadas Nacionales De InvestigaciÓn En Ciberseguridad (JNIC), Vigo, 2023.