

Received August 11, 2021, accepted August 15, 2021, date of publication August 24, 2021, date of current version September 1, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3107403

A Moving Target Defense Strategy for Internet of Things Cybersecurity

ANDRÉS AHARHEL MERCADO-VELÁZQUEZ¹,
PONCIANO JORGE ESCAMILLA-AMBROSIO¹, (Senior Member, IEEE),
AND FLORIBERTO ORTIZ-RODRÍGUEZ²

¹Instituto Politécnico Nacional, Centro de Investigación en Computación, Ciudad de México 07738, Mexico

²Instituto Politécnico Nacional, Escuela Superior de Ingeniería Mecánica y Eléctrica, Unidad Zacatenco, Ciudad de México 07738, Mexico

Corresponding author: Ponciano Jorge Escamilla-Ambrosio (pescamilla@cic.ipn.mx)

This work was supported in part by Consejo Nacional de Ciencia y Tecnología (CONACYT), and in part by Instituto Politécnico Nacional (IPN) under Grant SIP-1999 and Grant SIP-20210039.

ABSTRACT Internet of Things (IoT) systems are becoming more common and present in our daily lives. The increase of Internet-connected devices has caused attackers to focus their attention more on these devices. Therefore, new and more sophisticated attacks on IoT systems are discovered every day. Currently, to ensure reliability and operability, most IoT systems are designed to operate in a relatively static configuration in a highly heterogeneous environment. However, a system that does not continuously change its configurations, i.e., a static system, gives an advantage to attackers; with enough time and resources, an attacker will eventually find and exploit the vulnerabilities of any static target. This work proposes a Moving Target Defense (MTD) strategy that randomly shuffles the communication protocols through which a node communicates to a gateway in an IoT network. The system's configuration changes have an associated cost. The objective of the proposed MTD strategy in this work is to balance the increase in system performance overhead, the increase in business impact (system unavailability), and, at the same time, the decrease in the probability of success of a given attack. A framework is proposed to design this strategy; this framework can guide any MTD strategy for IoT systems. The framework's objective is to find, after several iterations, the MTD strategy parameters that achieve a balance between five different measurable variables of an IoT system.

INDEX TERMS Moving target defense, Internet of Things, cybersecurity, framework.

I. INTRODUCTION


According to [1], the number of IoT devices has increased since 2019 by 30%, and the number of IoT devices worldwide is expected to almost triplicate from 8.74 billion in 2020 to more than 25.4 billion devices in 2030. This increased surge has made IoT systems an increasingly common target for attackers.

In today's environment, IoT systems and all information technology are built to operate in a relatively static configuration. This static approach is a legacy of information technology systems designed for simplicity when malicious exploitation of system vulnerabilities was not a concern [2]. A static system, i.e., a system that does not continuously change its configuration over time, provides attackers with enough time to study such static system configurations. In other words, with enough time and resources, attackers

will eventually find and exploit vulnerabilities of any static target. In that sense, Moving Target Defense (MTD) emerges as a viable paradigm to defend these systems.

MTD strategies aim to substantially increase the cost of attacks by deploying and operating networks and systems in a manner that makes them less deterministic, less homogeneous, and less static [2]. However, the fundamental characteristics of IoT systems, such as heterogeneity and energy efficiency, require defense strategies that focus their efforts on reducing the probability of success of attacks.

This paper presents the results of designing an MTD strategy for an IoT system. This strategy aims to balance the increase in performance overhead, the increase in business impact (service unavailable), and the decrease of the probability of success of attacks on an IoT system. To reach this balance, it is proposed to randomly shuffle IoT communication protocols as part of "What to move" in the design of an MTD strategy. Regarding the determination of the remaining elements that constitute an MTD strategy, i.e., "When to

The associate editor coordinating the review of this manuscript and approving it for publication was Tyson Brooks .

move” and “How to move,” a framework is proposed for these tasks.

This approach intends that the proposed framework can be used as a guideline to design any MTD strategy for IoT systems. The framework will help balance the three goals mentioned in the previous paragraph when designing an MTD strategy.

This work proposes an MTD strategy that uses random shuffling applied to IoT communication protocols as preventive mechanisms to minimize the probability of a successful attack while achieving a balance between increase business impact and system performance overhead. This work has the following unique contributions related to the current state-of-the-art:

- 1) This is the first work that proposes an MTD strategy that uses the shuffling method in IoT communication protocols. The goal of randomly shuffling the communication protocols through which a node communicates to the gateway in an IoT network is to minimize the probability of having a successful attack that exploits vulnerabilities in IoT systems communication protocols. There is no previous research in the literature that takes advantage of IoT systems’ heterogeneity; particularly, shuffling methods to switch between different IoT communication protocols have not been proposed.
- 2) In order to address the issues of “When to move” and “How to move” to perform the shuffling of IoT network communication protocols, a framework has been proposed. However, note that the proposed framework can be used to address the “What to move,” “When to move,” and “How to move” questions in the design of any MTD strategy for IoT systems. Although, in this work, since the “What to move” question remains a static element in the design of the proposed strategy, the framework is used to address the “When to move” and “How to move” questions.
- 3) This paper proposes to find a balance or trade-off between five measurable variables of the IoT system. To obtain the parameters in the strategy to achieve this balance, the system’s five variables have been considered objective functions. The evaluation of these objective functions has been approached as a multiple-criteria decision problem.

The rest of this paper is organized as follows:

Section II provides an overview of the related work reported in the literature in terms of MTD strategies for IoT systems that aim to balance the costs and benefits represented by the MTD paradigm for IoT systems. Section III describes the elements that compose an MTD strategy, i.e., the three fundamental issues that constitute an MTD strategy: “What to move,” “How to move,” and “When to move.” In addition, section III presents the proposed framework for designing MTD strategies for IoT systems, detailing its components. Section IV describes the metrics used to evaluate the behavior of the proposed strategy according to the framework presented. Section V describes the multi-criteria analysis

method used in this work to compare the results of the evaluation of the proposed strategy. Section VI presents the results of the MTD strategy design for a use case IoT system, i.e., the final parameters obtained by using the proposed framework. Finally, section VII presents the conclusions and future work of this research.

II. RELATED WORK

Although MTD is a cyber-defense paradigm that emerged more than ten years ago, its application in IoT systems has not been widely explored and can still be considered an immature field. The fundamental characteristics of IoT systems prevent classical MTD strategies from being fully feasible for these systems. However, several works have recently emerged in the literature of the area proposing MTD strategies applied to IoT systems.

A. MTD STRATEGIES FOR IOT

An MTD strategy proposed in the literature that uses the shuffling method to modify MAC-IPv6 addresses is proposed in [4]. The authors propose an algorithm referred to as Address Shuffling Algorithm (AShA). According to the authors, this algorithm is energy-efficient, has minimal impact on the network overhead, and is easy to implement. The proposed algorithm uses a cryptographic hash that enables a controlled and collision-free address shuffling. The objective of the proposed strategy is to increase the security of systems that are bandwidth and power-constrained, as an overload can severely affect the performance of the IoT system. The algorithm was evaluated theoretically and through simulation. While the work presents a novel proposal that can positively impact the state-of-the-art in the context of MTD strategies for resource-constrained devices, the authors focused on evaluating only the performance of the system and do not clearly explain how the proposed algorithm improves the cybersecurity of the system or what is the cost of the strategy concerning service availability.

The work of Judmayer *et al.* [13], published in 2018, presents the results obtained by deploying an MTD strategy that constantly changes the IP addresses of network interfaces of connected IoT devices. The results obtained are shown in terms of the performance assessed on the network end devices or nodes. The evaluation is carried out on both IPv4 and IPv6 addressing. The experimental setup from which the results were derived was implemented with hardware widely used in IoT environments, such as the Raspberry Pi [14] and Carambola [15]. While the paper presents results that extend the state-of-the-art concerning the performance of IoT devices when deploying a shuffling-based MTD strategy specifically applied to IP network addresses, the work does not show how changing IP addresses influences the cybersecurity of the system. Although a priori shuffling the IP addresses of an IoT device can increase the system’s randomness and thus decrease the probability of a successful attack, the work presented does not evaluate the system’s cybersecurity or the positive impact of continuously changing

IP addresses in an IoT system. However, the approach contributes to the state-of-the-art results that demonstrate the viability of the shuffling method at the network level applied to IoT systems, evaluating this strategy on real hardware widely used in the IoT ecosystem.

Ge *et al.* [6] proposed an MTD strategy that randomizes the topology of an IoT network by shuffling the configuration of a network composed of decoy and real nodes. This proposal is unique in the state-of-the-art. There is no MTD strategy for IoT in the literature that proposes shuffling the topology of an IoT network composed of decoy and real nodes to maximize the attacker's complexity. This strategy is called by the authors NTS-MTD. To address the question "When to move," the authors deployed and evaluated four different methods (fixed/random/adaptive/hybrid). Regarding the question "How to move," the authors proposed three different methods (genetic algorithm/optimization based on the lure attack path/random). Similarly, the evaluation was performed to determine the best solution to the question "When to move," the results obtained from the evaluation of each proposed method were compared to address the solution to the question "How to move." While Ge *et al.* evaluated system performance, downtime, and network cybersecurity, the work lacks a multi-criteria analysis to determine which one is the best solution from all the proposed.

In their paper, Navas *et al.* [5] presented an MTD strategy that aims to increase the resilience of IoT systems against jamming attacks. The authors proposed Direct-Sequence Spread-Spectrum (DSSS) as the parameter to move in the wireless communication of a node in an IoT network, i.e., the "What to move" in the proposed MTD strategy. Unlike most works in the literature, in the context of MTD for IoT, this work focuses on the security of the physical layer. The objective of the strategy proposed by the authors is to increase the resilience of the system to a jamming attack. To achieve this goal, the author proposed to randomize the spreading sequences in a DSSS system. The spreading sequences are generated using Cryptographically Secure Pseudo-Random (CSPR) number generators. Sharing the cryptographic key between the two communicating network elements is necessary to synchronize the spreading sequences. To address the question "When to move?" in the strategy proposed by the authors, the devices to communicate decide on a variable that changes according to the shared sequence; this variable determines the hopping frequency. The tool used to evaluate both the strategy's performance and the system's resilience to interference attacks was MATLAB, so the results obtained were derived from simulations of the system. For the design of the strategy, the authors present a model that explains the procedure used to agree on the spreading sequences between two devices. Although frequency hopping is not a new method to avoid interference between wireless communications, this is the first work that proposes a strategy inspired by the MTD paradigm to add randomness to the frequency hopping agreed in wireless communications. Although Navas *et al.* evaluated the system's resilience

against jamming attacks and evaluated the system's performance when deploying their strategy, the work does not specify or evaluate the business impact of implementing their strategy in a real IoT system. The presented work also does not compare the results obtained with other MTD strategies. While the authors contribute to the state-of-the-art by evaluating the proposed MTD strategy using simulations, the strategy's viability was not evaluated in real environments, such as a testbed or using real IoT systems hardware. However, their work demonstrated that an MTD strategy could be deployed in any element of an IoT system, either at the physical layer level or at any level.

Another proposed MTD strategy focused on randomizing IP addresses in IoT devices was presented by Zeitz *et al.* in [16]. This strategy explores the use of Micro Moving Target IPv6 Defense (μ MT6D). The presented strategy is designed as a defense mechanism for low power consumption and resource constrained devices. The strategy is based on IPv6 address rotation, whereby each network node or end device rotates its IP address based on a lightweight hashing algorithm for address calculation. The strategy aims to protect IoT devices from targeted attacks, taking advantage of a dynamic MTD-based configuration's benefits against an adversary. According to the authors, the presented strategy successfully prevents denial-of-service attacks or eavesdropping passive attacks. The authors in [16] evaluated the system's performance when the strategy is running and compared the results obtained with the evaluation of the system when the MTD strategy is not running. The evaluation of the MTD strategy concerning system performance was carried out through simulation using virtualization of operating systems widely used in IoT applications such as Contiki. The results obtained in the simulations correspond to the power consumption of the nodes when the MTD strategy is being executed and when the strategy changes the IP address of each device every five minutes. The work presented demonstrates (at least in simulation) the viability of deploying MTD strategies in resource-constrained IoT devices at the network level. However, the work does not present results on the viability of the benefits of implementing MTD strategies with respect to cybersecurity; the measurements conducted focus only on system performance and do not consider the impact on service availability between each IP address change. Finally, the proposed work presents results obtained through simulation; however, this does not necessarily represent that the strategy is viable in a real environment.

B. MTD-RELATED IOT FRAMEWORKS

Few works have been published in the literature about frameworks that serve as guidelines for designing MTD strategies for IoT. In this section, a couple of frameworks proposed in the state-of-the-art are reviewed.

Navas *et al.* in [3] proposed an MTD framework suitable for IoT systems. The proposed framework aims to assist in the design and implementation of MTD strategies for IoT systems. According to the authors, the framework abstracts,

generalizes, and links common components of MTD strategies. Furthermore, MTD strategies based on this framework can be evaluated against each other. The proposed framework is conceived to design strategies where there is an interaction between two elements of the IoT network, e.g., the gateway and a node. The authors designed two MTD strategies based on the framework: one that targets UDP port numbers (port-hopping) and the other, the Constrained Application Protocol (CoAP); both strategies were implemented using a real IoT hardware platform. Furthermore, based on the experiments performed, the authors evaluated the system's security and calculated the probability of a successful Reconnaissance-Phase attack. Despite that the framework presented in [3] has the necessary elements to design an MTD strategy for IoT, the framework lacks components to evaluate the designed MTD strategies. Furthermore, the framework also lacks components that aim to apply optimization methods and thus generate strategies that decrease the costs associated with the MTD paradigm. Finally, the framework proposed by the authors does not allow the conception of strategies that are not 100% based on randomness, i.e., the framework does not consider the design of reactive strategies or strategies that use hybrid methods that are not 100% based on randomness.

Kyi and Koide in [17] proposed a framework for securing IoT systems with an MTD approach to be a starting point for combining various defense strategies at different levels of the IoT. The proposed framework consists of two main components and a third complementary component. The first component corresponds to a real IoT system. The second component of the framework corresponds to a virtual IoT system. The third component corresponds to an attack detection system. The authors propose to apply MTD methods such as IP address translation and code diversification to the two main components of the framework. The virtual system receives information from the real system when an attack happens; then the virtual system sends feedback to the real system. According to the authors, through this communication is possible to estimate the future status of the virtual system by sending feedback information to the real system. The framework presented in [17] is quite simple, compelling the idea of implementing any MTD strategy in a cyber-physical system composed of a "virtual system" and a "real system." However, while this framework can be used to understand basic MTD concepts, the framework does not have sufficient elements to design MTD strategies for IoT systems.

Both the work in [3], as well as the work presented in [17], have served as a starting point and inspiration for the proposed "MTD strategy design framework" presented in section III of this paper.

III. MTD STRATEGY DESIGN

A. ELEMENTS OF AN MTD STRATEGY

According to Cai *et al.* [7], there are three fundamental design questions that any MTD strategy needs to define: "What to move," "How to move," and "When to move."

1) "WHAT TO MOVE"

Refers to the elements or components of the system that change over time, i.e., the moving parameters.

Let S be the set of all distinct states to which an element of the system can change to, then,

$$S = \{S_1, \dots, S_n\} \quad (1)$$

where n is the total number of states. The difference in system configuration between any of the elements of S is the "What to move" in the design of the MTD strategy.

For the strategy proposed in this work, the "What to move" corresponds to the IoT communication protocols; in other words, the parameter that changes in the system configuration is the communication protocol through which a node communicates with the gateway in the IoT network. In total the number of protocols shuffled in the proposed strategy is equal to 4, i.e., for this strategy, in equation (1), $n = 4$, where $S_1 = \text{Wifi}$, $S_2 = \text{BLE}$, $S_3 = \text{Zigbee}$ and $S_4 = \text{LoRa}$.

2) "WHEN TO MOVE"

It refers to the time between one state change to another. Formally, the "When to move" can be defined as the time elapsed between the state S_i and the state S_{i+1} ; this time can be fixed, random, or based on a specific event.

Due that a random time can be within the range $(0, +\infty)$, for this work, we consider "random time" to be any time given between the interval $[\alpha, \beta]$, where α and β correspond to a fixed time that must satisfy the inequality $\alpha < \beta$.

For the proposed strategy, a random time with a discrete uniform probability distribution has been considered in all experiments. However, the limits of the range defined by $[\alpha, \beta]$ are determined by using the proposed framework.

3) "HOW TO MOVE"

It refers to how the elements or parameters of the system change from one state to another, i.e., the way to move of the "What to move" set.

In the literature, many methods try to guide the best possible movement, from game theory, probability distributions, or a fixed movement sequence.

For this work, a discrete uniform probability distribution is used to determine the next move or state of the system. The evaluation of these moves is carried out using the proposed framework.

B. MTD STRATEGY DESIGN FRAMEWORK

The framework shown in Fig. 1 is proposed to obtain the "When to Move" and "How to Move" parameters of the IoT MTD strategy developed in this approach. This framework can be used as a guideline to design any MTD strategy for IoT systems to find a balance between the increase in performance overhead, the increase in business impact, and the decrease in the probability of a successful attack. A specific MTD design strategy for a given IoT system can use this framework as an archetype to build upon it.

The proposed framework is an iterative cycle that is repeated until the parameters of the strategy achieve a

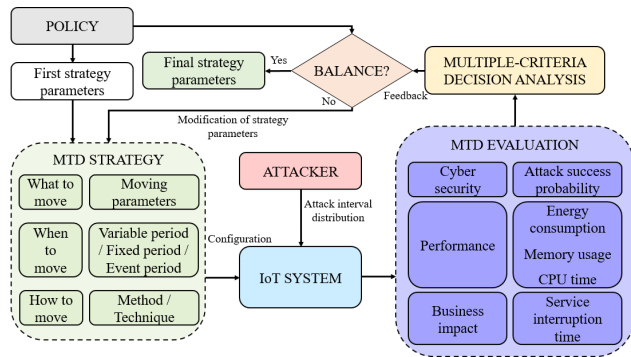


FIGURE 1. MTD strategy design framework components.

balance between the increase in system performance overhead, the increase in business impact, and the decrease in the probability of success of a given attack (or set of attacks). A description of each component (see Fig. 1) of the proposed framework is provided as follows.

1) POLICY

The POLICY component establishes a set of statements that should be stipulated by the governance or organizational entity responsible for establishing the system’s cybersecurity countermeasures.

This set of statements should establish general objectives or desirable goals in the context of the system’s cybersecurity (attack success probability). Furthermore, it is a fact that the use of MTD strategies in any system impacts its performance (overhead on energy consumption, memory usage, CPU time) and generates an effect on the business availability (service unavailable).

Hence, the implementation of any MTD strategy in any system must establish desirable limits on the impact in the system’s performance and business availability and define the hierarchy of three objectives: performance overhead, business availability, and cybersecurity.

The POLICY component is used in the framework as the a priori preferences set by the “Decision Maker” in a Multiple-criteria decision analysis.

2) MTD STRATEGY

The MTD STRATEGY component contains the parameters that answer the questions “What,” “When,” and “How” to move. These parameters are used as inputs to the IoT SYSTEM component.

3) IOT SYSTEM

The IoT SYSTEM component corresponds to the sum of all the elements that make up the set of configurable and non-configurable elements, whether physical or logical. For the use case presented in this work, the IoT SYSTEM component corresponds to an IoT testbed.

4) ATTACKER

In order to measure the probability of a successful attack, the ATTACKER component generates constant attacks to

the IoT SYSTEM. These attacks can be real or simulated. This component of the framework establishes the limits and capabilities of an attacker, defines the interval or frequency of attacks, and the types of attacks performed on the system, i.e., it characterizes the attacker.

5) MTD EVALUATION

The MTD EVALUATION component requires as inputs measurements of the performance of the system and service unavailability, as well as an estimation of the probability of a successful attack.

The MTD EVALUATION takes the information from the measurements made on the system during the execution of the MTD strategy. It produces as output five variables or metrics that will be analyzed in the next component of the framework.

6) MULTIPLE-CRITERIA DECISION ANALYSIS

In the MULTIPLE-CRITERIA DECISION ANALYSIS (MCDA) component, the variables from the MTD EVALUATION component are analyzed. These variables are treated as objective variables. Using multi-criteria decision-making methods, those parameters of the strategy that perform best according to the policies stipulated by governance are selected.

7) BALANCE

The BALANCE component receives as inputs the results of the MCDA component and compares them with the targets established in the policies. If a balance on the established objectives is achieved, then the final parameters of the strategy are extracted; otherwise, new parameters are generated for the strategy, and the framework cycle is repeated.

C. IOT TESTBED

In order to evaluate the probability of a successful attack, the system performance overhead, and the business impact (unavailability of the service), an IoT testbed has been implemented; this testbed belongs to the IoT SYSTEM component of the proposed framework. In other words, the design of the proposed strategy has been tested and evaluated on an IoT testbed.

The IoT testbed consists of three nodes that communicate to a gateway through different IoT communication protocols. Each node in the network can communicate to the gateway using four different communication protocols, Bluetooth, WiFi, LoRa, and Zigbee. The nodes in the network send and receive information from the gateway. The gateway communicates with a private web server in the cloud, where the collected data can be stored, processed, analyzed, and viewed via a smartphone or web browser. Fig. 2 shows the general configuration of the network and the devices.

The nodes and the gateway of the testbed’s IoT network are implemented with Raspberry Pi development boards and communications HATs.

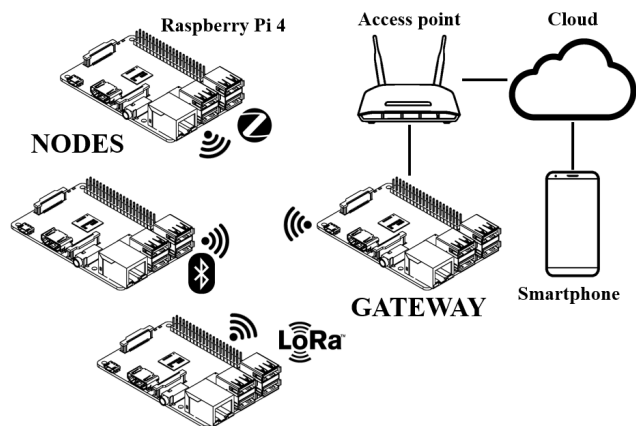


FIGURE 2. IoT testbed setup.

IV. MTD STRATEGY EVALUATION

According to the proposed framework, the MTD EVALUATION component takes the information from the measurements made on the system during the execution of the MTD strategy and produces as outputs five variables or metrics that evaluate the system performance overhead, the impact on the business, and cybersecurity.

The results obtained of the variables are a consequence of measurements taken during the execution of each MTD strategy. The strategies were executed in the testbed for a period ranging from 12 to 24 hours. The time of execution of each strategy depended on the statistics of the data collected. If the arithmetic mean stabilized at a point, then the execution of the strategy was stopped and, therefore, the sampling of measurements. In other words, if the average of the results of the measurements obtained stabilized at a point, then the execution of the strategy in the testbed was stopped.

It is relevant to clarify that because IoT devices are the most important elements in an IoT system, the measurements were performed on the IoT network nodes, so the results obtained correspond to the behavior of the network nodes. This section describes the procedure followed to measure five variables of the system, which describe its performance.

A. PERFORMANCE

For the system performance evaluation, three variables were considered: energy consumption, memory usage and CPU time.

1) ENERGY CONSUMPTION

In order to obtain the energy consumption levels achieved by the network nodes, electrical current measurements were performed during the execution of the MTD strategy. Fig. 3 shows the electrical schematic diagram used to measure the electrical current in the testbed nodes.

Direct measurements were performed with the help of an ammeter to measure the electrical current of the network nodes. The measuring device is essentially a voltmeter with a precision resistor. The ammeter uses Ohm's law to obtain the value of the current flowing through the circuit.

As shown in Fig. 3, the procedure consists of interrupting the flow of electrons in the circuit and connecting the ammeter as a bridge in the circuit through which the current is flowing.

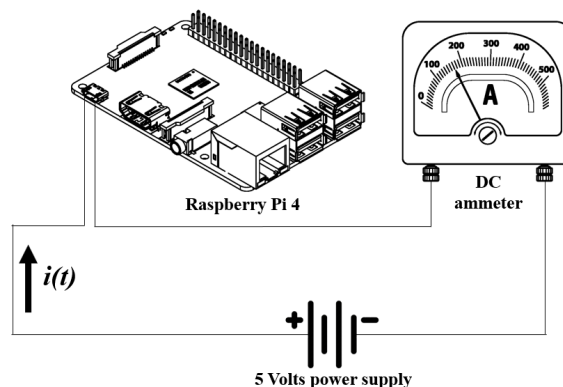


FIGURE 3. Electrical schematic diagram used to measure the electrical current in the testbed nodes.

2) MEMORY USAGE

The Node.js OS Library was used to measure the node's memory usage. The OS module provides operating system-related utility methods and properties [8].

Memory usage was sampled every 100 milliseconds (at a sampling frequency of 10 Hz) to obtain the average memory used by the node during the execution of the strategy.

The data collected from each sample was dumped into a CSV file. Once this process was completed, the arithmetic mean of all the samples collected was obtained. An average of half a million samples was collected for each experiment.

3) CPU TIME

As well as the memory usage, the Node.js OS Library [8] was used to measure the CPU time of the node.

The "idle task" was used to measure the processor time. The idle task is executed every time the processor is not running any other task. In other words, the CPU idle time is the inactivity time of the processor. The total run time minus the total idle time is subtracted to calculate the CPU working time.

The total working time is divided by the number of hours the testbed executed the MTD strategy to obtain the average CPU working time per hour.

B. BUSINESS IMPACT

Making continuous changes to the system configuration when implementing MTD strategies increases power consumption, memory usage, or processor time. In addition, switching from one state to another causes an interruption in the service; this service interruption time generates a business impact. In the case of the strategy proposed for this work, randomly shuffling the communication protocol used by the network nodes to communicate with the gateway generates a service interruption, affecting the service's availability.

Hence, to evaluate the business impact, the service downtime was measured. Fig. 4 shows the events that occur when switching from one IoT communication protocol to another in the order of their occurrence.

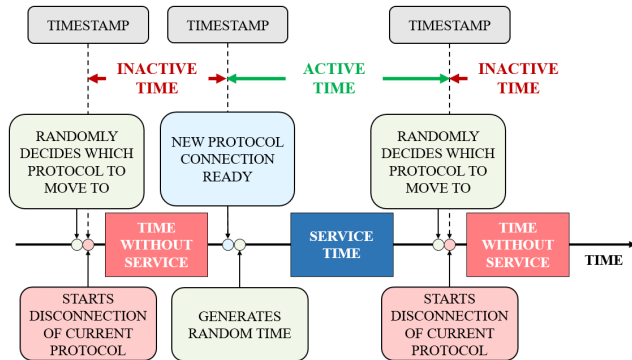


FIGURE 4. Active and inactive time measurement.

According to the three-layer IoT reference architecture [19], the shuffling of protocols is done at layer two, i.e., at the network layer. Although the purpose of the four technologies used is to communicate the IoT network nodes with the gateway, the establishment and negotiation of the communication are different for each protocol. For example, the address negotiation is different for WiFi and LoRa communications protocols. Therefore, if a protocol is switched to any other, then new negotiations are generated according to each communication protocol. Under this context, the time of service unavailable includes the time spent for configurations of the protocols in higher layers.

Timestamps were used to measure downtime and active service time. The IoT SYSTEM downtime was measured by calculating the difference between the timestamp when the connection of the new protocol was ready and the timestamp when the disconnection of the previous communication protocol was initiated.

The total service downtime was divided by the total time the testbed was executing the MTD strategy to obtain the percentage of time the service was down.

C. CYBERSECURITY

In the context of system cybersecurity, the probability of a successful attack was evaluated. Several assumptions on attack behaviors and goals were made to characterize attackers to estimate the probability of success of an attack.

1) ATTACKER CHARACTERIZATION

The following assumptions were made about the attacker’s goals and behavior:

- It is assumed that the attacker knows that the target is constantly changing communication protocols. Hence, white box attacks are assumed.
- The attacker has the capabilities to attack all the communication protocols considered in this work.

- The attacker’s goal is to deny service to the target by attacking vulnerabilities in the communication protocols.
- An attack is assumed to be successful when an unauthorized result is produced.

2) SUCCESSFUL ATTACK

In this work, an attack is assumed to be successful when an unauthorized result is produced. Also, this research assumes that an unauthorized result is produced when the attack crafting and execution period is over and the system has not changed to the next state (change of communication protocol).

Fig. 5 shows a graphical representation of the conditions for a successful and an unsuccessful attack. This representation is based on the model presented by Zheng and Namin in [9].

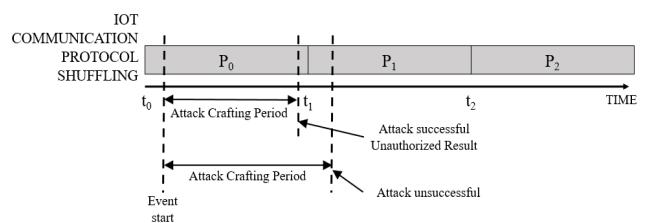


FIGURE 5. Successful and unsuccessful attack representation.

3) PROBABILITY OF SUCCESS OF AN ATTACK

In order to estimate the probability of success of an attack, in this case, a DoS attack, to the wireless communication protocols of the IoT network node, four different real attacks were conducted, one for each communication protocol. The objective of the execution of the attacks was to deny the service of the IoT network nodes. For the Bluetooth protocol, hijacking attacks were executed using the BtleJack tool [11]. The Aircrack-ng [12] tool was used to execute deauthentication attacks over WiFi. Attacks targeting LoRa and Zigbee communication protocols were performed through a script written in JavaScript, executing replay attacks.

A total of approximately 90 real attacks were executed on the testbed. These attacks were executed while the IoT system was running the MTD strategy of randomly changing states (shuffling communication protocols) between a time range of 5.5 min to 6.5 min. Of the total number of attacks carried out, 61 were successful. Fig. 6 shows the distribution of successful attacks versus the “attack crafting period” by the IoT communication protocol being attacked.

However, carrying out real attacks to the testbed caused two problems. The first problem encountered was that the attacks to be performed were limited by the hardware available. The second problem faced was that performing continuous attacks to the testbed was impractical because once the system had been compromised, the system and attack must be manually restarted.

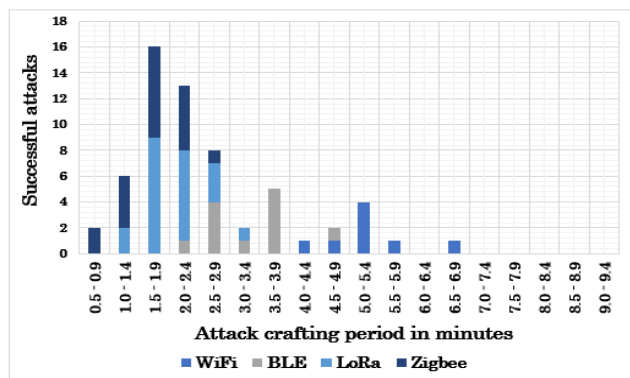


FIGURE 6. Histogram of the distribution of 61 real successful attacks.

To solve the two problems mentioned above and estimate the attack success probability more accurately, the Monte Carlo method [18] was used to simulate attacks to the testbed.

Simulating the execution of attacks through the Monte Carlo method has several advantages over performing real attacks; firstly, it is possible to simulate any capability of the attacker, i.e., the attacker’s capabilities are not limited by the available hardware. Another advantage is that no user interaction is required; therefore, the simulation can be left running without the need to intervene each time a successful attack is carried out. Finally, through Monte Carlo simulations, more attacks can be performed in less time.

An agent was run on each node of the IoT network to estimate the probability of success of an attack. The agent performs the same steps that a real attacker would perform, with the difference that the agent does not execute the actual attack; however, the agent can scan the environment and obtain the communication protocol through which the network node is communicating with the gateway at that moment. Based on this information, the agent simulates the execution of the attack.

The following paragraphs describe the steps performed by the agent to determine whether an attack was successful or unsuccessful.

The first step of the attack simulation is to scan the environment. Through a scan, the agent obtains the current state of the system, i.e., the current communication protocol through which the node and gateway are communicating.

The second step is to generate a random number. This random number represents the attack crafting period, i.e., the estimated time from the attack launch until an unauthorized result is reached. This time is obtained randomly within a given time range; the time range and the probability distribution obey a baseline.

In this work, the baseline was derived from the data collected by performing real attacks, as previously described, i.e., from the information in Fig. 6.

The aim of using the information obtained from the attack crafting period (see Fig. 6) as a baseline was to achieve a behavior in the simulation, i.e., the behavior of the agent, to be as close as possible to the behavior that would have been

obtained by executing real attacks. In consequence, the results obtained are as close as possible to those that would have been obtained if real attacks had been performed.

Fig. 7 shows a histogram constructed from a sample of 75,000 pseudo-random numbers. The probability distribution of the histogram in Fig. 7 was derived from the average of the attack crafting period obtained from the execution of the real attacks, i.e., from the baseline.

In the third step of the simulation, the agent waits the time obtained randomly (in the second step), i.e., the agent spends a waiting time equal to the time obtained randomly. This time is the one referred to as the “attack crafting period,” previously mentioned.

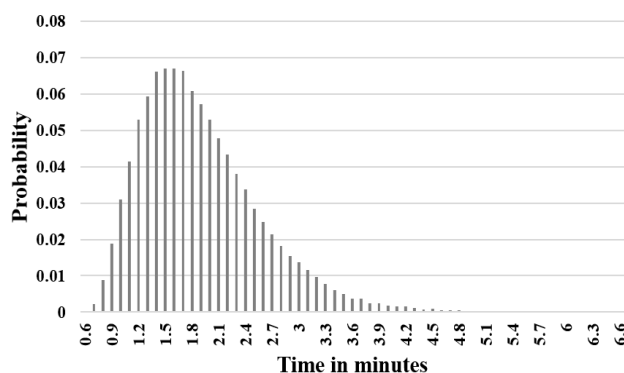


FIGURE 7. Histogram generated from a sample of 75,000 pseudorandom numbers.

Finally, after the attack crafting period, the agent scans the environment again and obtains the current state of the system a second time. If the current system’s state is the same as the system’s state registered in step 1, then the agent considers that a successful attack has taken place; if the system’s state obtained at the beginning is different from the system’s state after the attack crafting period, then the agent registers an unsuccessful attack.

The process described in the previous paragraph was carried out repeatedly until the attack success probability converged to one point. For the experiments conducted, the probability converged after an average of 200 executions.

A useful analogy to understand the process described above is the game of darts. The player or dartist would be the attacker or agent, while the dartboard is the IoT system. In this analogy, the dartboard is constantly changing color, between a total of four colors. The time that the dartboard remains in one color before changing to another is randomly chosen, and the choice of the next color is also randomized.

In the case of the dartist, she/he has four different colors of darts (the same four colors that the dartboard is constantly changing to). The player’s objective is to hit the target with a dart of the same color as the one on the board before the dartboard changes color again.

In the above analogy, the player is an agent, i.e., a robot. This robot shoots the darts at different speeds; some darts travel at higher speeds, while some other darts travel at lower

speeds, the purpose of the game is that the dartboard changes color before the dart hits the target; thus, if the dart travels at a slow speed, then it is likely to fail to reach the target in time before it changes color.

The speed of each dart, i.e., the time it takes for the dart to reach the target, is determined randomly based on the probability distribution established from the baseline.

V. MULTIPLE-CRITERIA DECISION ANALYSIS

As previously mentioned, the objective of the designed strategy is to achieve a balance between different goals. The balance is given by the multi-criteria analysis applied to the different solutions obtained in each iteration of the framework.

Multiple-criteria decision analysis helps to find one suitable solution that satisfies in a balanced way the goals in the policy set stipulated by the governance or organizational entity responsible for establishing the system’s cybersecurity countermeasures.

The multi-criteria analysis method used in this work is the Multi-Objective Optimization on the basis of Ratio Analysis (MOORA) method. MOORA was first introduced in 2006 by Brauers and Zavadskas [10]. The objective of this method is to find the best possible solution given a set of solutions and certain constraints. It has been successfully applied to solve various types of complex decision-making problems in many fields in economics and engineering.

The procedure for the MOORA implementation method is described as follows:

Step 1: Create a Decision Matrix.

The decision matrix is represented as the x matrix,

$$x = \begin{bmatrix} x_{11} & x_{12} & \cdot & x_{1n} \\ x_{21} & x_{22} & \cdot & x_{2n} \\ \cdot & \cdot & \cdot & \cdot \\ x_{m1} & x_{m2} & \cdot & x_{mn} \end{bmatrix} \quad (2)$$

where x_{ij} is the response of alternative j to objective i , $i = 1, 2, \dots, n$ are the objectives, $j = 1, 2, \dots, m$ are the alternatives [10].

Step 2: Normalize the Decision Matrix.

A second matrix ${}_N x_{ij}$, normalized, is obtained from equation (3),

$${}_N x_{ij} = \frac{x_{ij}}{\sqrt{\sum_{j=1}^m x_{ij}^2}} \quad (3)$$

Step 3: Assign the objectives weights.

The way to assign the weights of the objectives is a procedure that is not established in the MOORA method; therefore, the assignment of weights is at the discretion of the decision-maker, as long as equation (4) is satisfied,

$$\sum_{j=1}^m W_j = 1 \quad (4)$$

Because the objective for this work is to find a balanced solution, the weights assigned to each objective are the

same, i.e., each objective has the same weight and, therefore, the same priority.

Step 4: Weighting the normalized matrix.

Weight the normalized matrix by multiplying each element of the normalized matrix by the weight of each objective or criteria,

$$w_N x_{ij} = {}_N x_{ij} W_{ij} \quad (5)$$

Step 5: Obtain the evaluation of each alternative.

In (6), the value of y_j is the evaluation of each alternative and represents the normalized response of alternative j to objective i ; these normalized responses of the alternatives to the objectives are in the interval $[0, 1]$. For optimization, these responses are added in case of maximization and subtracted in case of minimization.

$$y_j = \sum_{i=1}^{i=g} w_N x_{ij} - \sum_{i=g+1}^{i=n} w_N x_{ij} \quad (6)$$

Step 6: Calculate the Tchebycheff distance to an ideal point.

In order to measure the distance between the alternatives and the reference point, the Tchebycheff Min-Max metric is chosen,

$$\min_j \{ \max_i |r_i - y_j| \} \quad (7)$$

where r_i is the i^{th} coordinate of the maximal objective reference point, each coordinate of the reference point is selected as the highest corresponding coordinate of the alternatives [10].

VI. RESULTS AND DISCUSSION

This section presents the results of the design of the MTD strategy for the chosen IoT use case obtained by using the proposed framework.

The use case, the established policy, and the first parameters of the strategy are described below.

A. USE CASE

An MTD strategy that randomizes IoT communication protocols cannot be used for all applications. Switching between different IoT communication protocols such as Bluetooth, WiFi, Zigbee, or LoRa limits the application scenarios, and the service interruptions caused between each state change in the system configuration also restricts the application scenarios.

The strategy proposed in this work can only be used in application scenarios that satisfy the following two criteria:

- The maximum distance between the nodes and the gateway in the proposed scenario must be less than the maximum distance reached by the wireless communication protocols (LoRa, WiFi, Zigbee, and Bluetooth). This distance is a maximum range of 30 meters.
- Because of interruptions caused by MTD, IoT time-critical application scenarios cannot be accepted, such as autonomous cars or medical monitoring.

The application scenario chosen as use case example in this work is a smart home. This application scenario satisfies the two criteria mentioned in the previous paragraph since the distances between a node and the gateway in an IoT network in a smart home typically do not exceed 30 meters. In that context, the four proposed communication protocols satisfy the distance requirements in a smart home. Furthermore, concerning the tasks performed by an IoT device in a smart home, such as measuring environmental variables, like temperature and humidity, or turning on/off lights, these are not time-critical tasks; hence, they are not highly affected by the downtime caused by protocol shuffling.

B. ESTABLISHED POLICY

The set of policies established for the chosen use case is listed below:

- Because the objective of the strategy is to achieve a balance between the desirable goals, the hierarchical weighting of the increase in system performance overhead, the increase in business impact, and the decrease of the probability of a successful attack are the same, i.e., no one goal is more important than any other.
- According to the application scenario, the strategy should maintain a low increase (30% desirable reference value) in the system performance overhead of the IoT network.
- According to the application scenario, the strategy should achieve a high decrease (70% desirable reference value) in the probability of a successful attack.
- According to the application scenario, the strategy should achieve a low increase (5% desirable reference value) in the business impact (unavailability of the service).

C. FIRST STRATEGY PARAMETERS

In order to carry out the first iteration of the framework, initial strategy parameters have to be proposed. These initial parameters will not necessarily be the final strategy parameters; they will depend on the evaluation and analysis of the system’s behavior and whether any solution achieves the balance proposed by the policy.

The initial strategy parameters proposed were as follows:
What to move.

- IoT communication protocols

When to move.

- At a uniformly random time within a range from 0.5 minutes to 1.5 minutes.
- At a uniformly random time within a range from 1.5 minutes to 2.5 minutes.
- At a uniformly random time within a range from 2.5 minutes to 3.5 minutes.
- At a uniformly random time within a range from 3.5 minutes to 4.5 minutes.
- At a uniformly random time within a range from 4.5 minutes to 5.5 minutes.

- At a uniformly random time within a range from 5.5 minutes to 6.5 minutes.

How to move.

- Randomly shuffling communication protocols in each node. Each communication protocol has the same probability of being the next one, i.e., 1/4 (uniform probability distribution).

The strategy parameters were implemented in each network node, i.e., each node independently executed the strategy according to the proposed parameters. Therefore, each node acts independently with respect to the other nodes in the network.

Fig. 8, Fig. 9, and Fig. 10 show the results obtained at the EVALUATION component output of the first framework iteration. In each of these figures, the horizontal axis corresponds to the strategies evaluated in the first iteration of the framework, as well as the behavior of the system when each strategy is executed (without MTD). In other words, the “x” axis in Fig. 8, Fig. 9 and Fig. 10 corresponds to the six strategies implemented on the testbed in the first cycle of the framework, as well as the behavior of the system without executing MTD.

In Fig. 8, the vertical axis corresponds to the results obtained in terms of the estimation of the probability of a successful attack. It is important to note that, according to

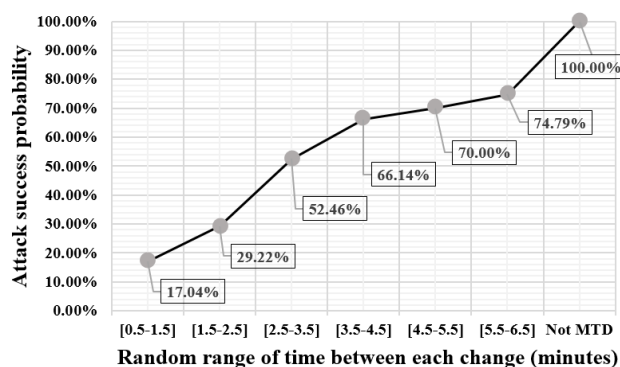


FIGURE 8. Attack success probability estimation.

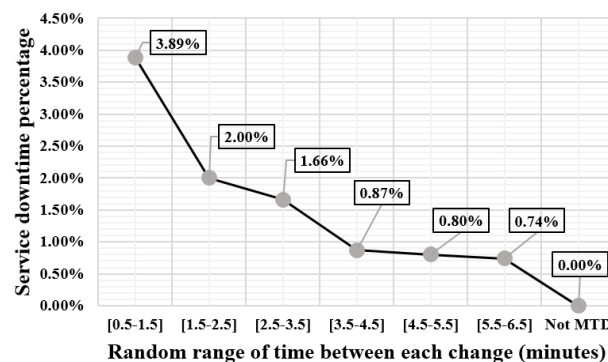


FIGURE 9. Percentage of service downtime.

the experiments conducted in this work, and following the process described in section V.C of this paper, the probability of a successful attack when the system does not execute any MTD strategy was estimated to be 100%. This value can be taken as a reference point to compare the results obtained in each of the six strategies executed in the IoT system.

The vertical axis in Fig. 9 corresponds to the results obtained in terms of service downtime produced in each MTD strategy in terms of percentage. That is, the percentage of time that the service remained interrupted compared to the total time that the MTD strategy was executed in the IoT system.

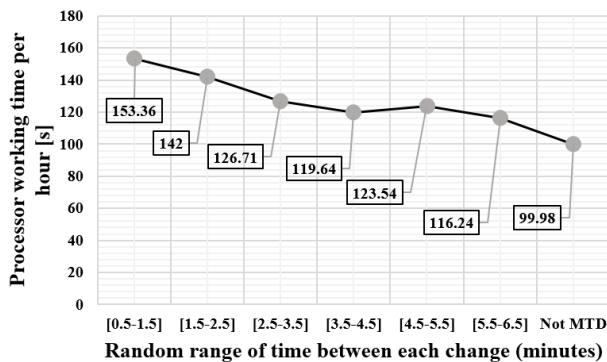
Fig. 10 shows the results concerning the performance obtained in each MTD strategy. The vertical axis in Fig. 10.a represents the average CPU working time per hour during the execution of each of the MTD strategies. In Fig. 10.b,

the vertical axis represents the energy consumption average in mWh per hour. Finally, the vertical axis in Fig. 10.c represents the memory usage average in MB, during the execution of each MTD strategy, during the first iteration of the framework.

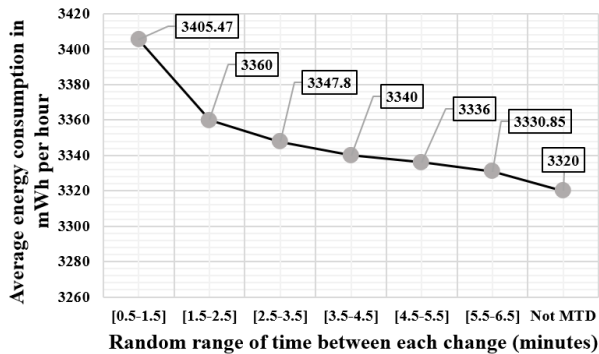
D. OBTAINING A BALANCED SOLUTION USING MCDA

Table 1 summarizes the results obtained from the first iteration of the framework. The weights assigned to each criterion correspond to the first sentence established in the policies of the use case.

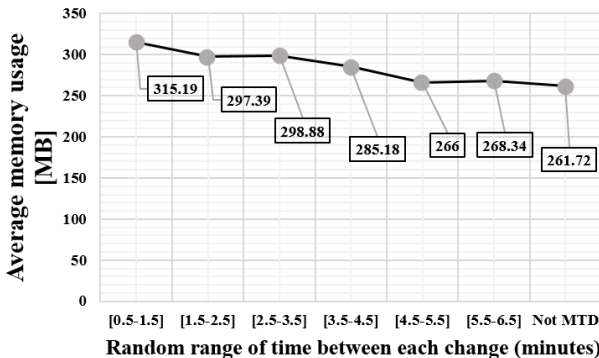
It is necessary to clarify that the MOORA method allows assigning different weights to different criteria. That is, if the policy had prioritized cybersecurity (attack success probability) over all the other criteria, despite the increase in system performance overhead or the increase in the percentage of system downtime, a higher weight could have been given to this aspect and, consequently, lower weights to the other criteria. Although the above criterion was not included in the experiments performed, there is no impediment to carry it out; however, for reasons of space, it is left as future work and as an option to be experimented by the possible adopters of the proposed framework.



(a) CPU time



(b) Energy Consumption



(c) Memory usage

FIGURE 10. Framework’s first iteration results in performance.

TABLE 1. Framework’s first iteration results.

Criteria	Attack success probability (%)	System downtime (%)	CPU time (S)	Energy Consumption (mWh)	Memory usage (MB)
Alternative					
[0.5-1.5] min	17.04	3.89	153.36	3405.47	315.19
[1.5-2.5] min	29.22	2.0	142.0	3360	297.39
[2.5-3.5] min	52.46	1.66	126.71	3347.8	298.88
[3.5-4.5] min	66.14	0.87	119.64	3340	285.18
[4.5-5.5] min	70	0.8	123.54	3336	266
[5.5-6.5] min	74.79	0.74	116.24	3330.85	268.34
Weighted	0.2	0.2	0.2	0.2	0.2
Optimum	Min	Min	Min	Min	Min

Implementing the MOORA method, from Table 1, the decision matrix x is obtained as follows:

$$x = \begin{bmatrix} 17.04 & 3.89 & 135.36 & 3405.4 & 315.19 \\ 29.22 & 2.00 & 142.00 & 3360.0 & 297.39 \\ 52.46 & 1.66 & 1267.71 & 3347.8 & 298.88 \\ 66.14 & 0.87 & 119.64 & 3340.0 & 285.18 \\ 70.00 & 0.80 & 123.54 & 3336.0 & 266.00 \\ 74.79 & 0.74 & 116.24 & 3330.8 & 268.34 \end{bmatrix} \quad (8)$$

After normalizing and weighting the decision matrix using equations (3) and (5), respectively, the weighted normalized

matrix $w_N x_{ij}$ is obtained, as follows:

$$w_N x_{ij} = \begin{bmatrix} 0.0248 & 0.1593 & 0.0956 & 0.0826 & 0.0890 \\ 0.0426 & 0.0819 & 0.0885 & 0.0818 & 0.0840 \\ 0.0765 & 0.0680 & 0.0790 & 0.0815 & 0.0844 \\ 0.0965 & 0.0356 & 0.0746 & 0.0813 & 0.0805 \\ 0.1022 & 0.0327 & 0.0770 & 0.0812 & 0.0751 \\ 0.1091 & 0.0303 & 0.0724 & 0.0810 & 0.0758 \end{bmatrix} \quad (9)$$

Then, after obtaining the evaluation of each alternative with equation (6) and having calculated the Tchebycheff distance to an ideal point with equation (7), the results indicating the best alternative are obtained.

After implementing the MOORA method on the results of the MTD strategies (or alternatives) evaluated in the first cycle of the proposed framework, the method provides a ranking where the alternatives are ordered from the best to the worst.

If no decision criteria are prioritized or assigned a higher weight, then the MOORA method gives the best alternative according to the behavior of each one, i.e., it chooses the most balanced alternative.

Table 2 shows the final ranking of the alternatives in order from the best to the worst. From the multiple-criteria decision analysis, it is concluded that the alternative [1.5-2.5] min is the alternative with the best overall behavior considering the three goals; performance overhead, business impact, and system cybersecurity, while the alternative [0.5-1.5] min cannot be considered.

TABLE 2. Results rank.

Alternative	Results	Rank
[1.5-2.5] min	0.05516200	1
[2.5-3.5] min	0.05171455	2
[3.5-4.5] min	0.07168787	3
[4.5-5.5] min	0.07732362	4
[5.5-6.5] min	0.08431721	5
[0.5-1.5] min	0.12905014	6

Fig. 11 shows a graph with the results of the MOORA method for each alternative and the ranking position of each one.

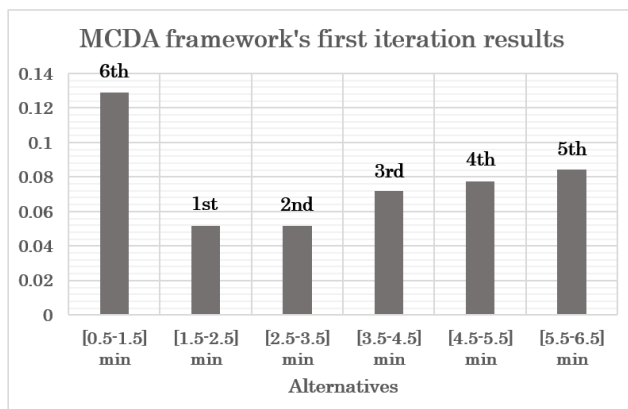


FIGURE 11. Ranking of the alternatives of the framework's first iteration.

It is important to note that although the alternative [0.5-1.5] achieved the lowest probability of a successful attack, it obtained the worst evaluation regarding the system performance overhead and business impact; therefore, according to the MOORA method, it is the alternative worst evaluated.

E. FINAL STRATEGY PARAMETERS

Once the best solution of the framework's first iteration was obtained, it should be verified that this solution satisfies the whole set of proposed policies. In this case, the best solution from the first iteration achieves the objectives established in the use case policy; therefore, it is unnecessary to carry out a second iteration of the framework to find new solutions.

Due to the above, it is established that the final parameters, i.e., the proposed strategy, is as follows:

What to move?

- IoT communication protocols

When to move?

- At a uniformly random time within a range from 1.5 minutes to 2.5 minutes.

How to move?

- Uniform random shuffling from one communication protocol to another. Each communication protocol has the same probability of being the next one, i.e., 1/4.

VII. CONCLUSION & FUTURE WORK

In this work, an MTD strategy that uses shuffling (randomization) applied to IoT communication protocols as preventive mechanisms to minimize the probability of a successful attack while achieving a balance between increasing business impact and system performance overhead was proposed.

In order to design this strategy, a framework has been proposed; the proposed framework can be used to address the "What to move," "When to move," and "How to move" issues in the design of any MTD strategy for IoT systems.

The framework uses multiple-criteria decision analysis to determine the best solution (balance) given a set of parameters (elements of an MTD strategy) and goals (policies).

In this work, we designed an MTD strategy for the chosen IoT use case that reduces by 70% the probability of success of a denial-of-service attack that exploits vulnerabilities in IoT wireless communication protocols, assuming that the probability of success of such attack is equal to 1, i.e., 100% if the MTD strategy is not implemented.

In addition, on average, considering CPU time, power consumption, and memory usage, the designed strategy increases system overhead by only 19% compared to the system performance when it is not running any MTD strategy.

Finally, the designed strategy affects the service availability by only 2%, compared to 0% of service interruption time when the system does not execute any MTD strategy.

The results obtained are favorable according to the policies established for the proposed use case. The strategy was evaluated on a real testbed composed of devices widely used in applications of the IoT.

As future work, the following research areas are planned to be explored:

- Explore game theory applied to the attacker/defender relationship to create more intelligent moving defense strategies, where changes from one state to another do not obey a uniform probability distribution, instead they can be a function of other variables, such as the robustness of the next state, or the capabilities of the attacker.
- Replace the use of multiple-criteria decision analysis and instead explore the behavior of genetic algorithms to find the best possible MTD strategy.
- Explore the use of software-defined networks in IoT systems in the context of MTD strategies.

REFERENCES

- [1] *Number of Connected Devices Worldwide 2030*. Accessed: Jul. 13, 2021. [Online]. Available: <https://www.statista.com/statistics/802690/worldwide-connected-devices-by-access-technology>
- [2] White House. (2011). *Trustworthy Cyberspace: Strategic Plan for the Federal Cyber Security Research and Development Program, Report of the National Science and Technology Council, Executive Office of the President*. Accessed: Jul. 13, 2021. [Online]. Available: https://www.nitrd.gov/pubs/Fed_Cybersecurity_RD_Strategic_Plan_2011.pdf
- [3] R. E. Navas, H. Sandaker, F. Cuppens, N. Cuppens, L. Toutain, and G. Z. Papadopoulos, "IANVS: A moving target defense framework for a resilient Internet of Things," presented at the IEEE Symp. Comput. Commun. (ISCC), Jul. 2020, doi: [10.1109/iscc50000.2020.9219728](https://doi.org/10.1109/iscc50000.2020.9219728).
- [4] F. Nizzi, T. Pecorella, F. Esposito, L. Pierucci, and R. Fantacci, "IoT security via address shuffling: The easy way," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 3764–3774, Apr. 2019, doi: [10.1109/jiot.2019.2892003](https://doi.org/10.1109/jiot.2019.2892003).
- [5] R. E. Navas, F. Cuppens, N. B. Cuppens, L. Toutain, and G. Z. Papadopoulos, "Physical resilience to insider attacks in IoT networks: Independent cryptographically secure sequences for DSSS anti-jamming," *Comput. Netw.*, vol. 187, Mar. 2021, Art. no. 107751, doi: [10.1016/j.comnet.2020.107751](https://doi.org/10.1016/j.comnet.2020.107751).
- [6] M. Ge, J.-H. Cho, D. S. Kim, G. Dixit, and I.-R. Chen, "Proactive defense for Internet-of-Things: Integrating moving target defense with cyberdeception," 2020, *arXiv:2005.04220*. Accessed: Jul. 13, 2021. [Online]. Available: <http://arxiv.org/abs/2005.04220>
- [7] G. Cai, B. Wang, W. Hu, and T. Wang, "Moving target defense: State of the art and characteristics," *Frontiers Inf. Technol. Electron. Eng.*, vol. 17, no. 11, pp. 1122–1153, Nov. 2016, doi: [10.1631/fitee.1601321](https://doi.org/10.1631/fitee.1601321).
- [8] *Node.js V16.4.2 Documentation*. Accessed: Jul. 13, 2021. [Online]. Available: <https://nodejs.org/api/os.html>
- [9] J. Zheng and A. S. Namin, "A survey on the moving target defense strategies: An architectural perspective," *J. Comput. Sci. Technol.*, vol. 34, no. 1, pp. 207–233, Jan. 2019, doi: [10.1007/s11390-019-1906-z](https://doi.org/10.1007/s11390-019-1906-z).
- [10] W. K. M. Brauers and E. K. Zavadskas. (2006). *The MOORA Method and Its Application to Privatization in a Transition Economy*. Accessed: Jul. 13, 2021. [Online]. Available: https://www.researchgate.net/publication/228345226_The_MOORA_method_and_its_application_to_privatization_in_a_transition_economy
- [11] D. Cauquil. *BtleJack*. Accessed: Jul. 13, 2021. [Online]. Available: <https://github.com/virtualabs/btlejack>
- [12] *Aircrack-ng*. Accessed: Jul. 13, 2021. [Online]. Available: <https://www.aircrack-ng.org/>
- [13] A. Judmayer, G. Merzdovnik, J. Ullrich, A. G. Voyiatzis, and E. Weippl, "A performance assessment of network address shuffling in IoT systems," in *Computer Aided Systems Theory—EUROCAST 2017*. Cham, Switzerland: Springer, 2018, pp. 197–204.
- [14] R. Pi. *Teach, Learn, and Make With Raspberry Pi*. Accessed: Jul. 13, 2021. [Online]. Available: <https://www.raspberrypi.org/>
- [15] 8devices. *Carambola—8devices*. Accessed: Jul. 13, 2021. [Online]. Available: <https://www.8devices.com/products/carambola>
- [16] K. Zeitz, M. Cantrell, R. Marchany, and J. Tront, "Changing the game: A micro moving target IPv6 defense for the Internet of Things," *IEEE Wireless Commun. Lett.*, vol. 7, no. 4, pp. 578–581, Aug. 2018, doi: [10.1109/LWC.2018.2797916](https://doi.org/10.1109/LWC.2018.2797916).
- [17] W. Kyi and H. Koide, "A framework of moving target defenses for the Internet of Things," *Bull. Netw., Comput., Syst., Softw.*, vol. 8, no. 2, pp. 104–107, 2019. Accessed: Jul. 13, 2021. [Online]. Available: https://www.researchgate.net/publication/337473697_A_Framework_of_Moving_Target_Defenses_for_the_Internet_of_Things
- [18] N. Metropolis and S. Ulam, "The Monte Carlo method," *J. Amer. Stat. Assoc.*, vol. 44, no. 247, pp. 335–341, Sep. 1949, doi: [10.1080/01621459.1949.10483310](https://doi.org/10.1080/01621459.1949.10483310).
- [19] M. A. J. Jamali, B. Bahrami, A. Heidari, P. Allahverdizadeh, and F. Norouzi, "IoT architecture," in *Towards the Internet of Things*. Cham, Switzerland: Springer, 2019, pp. 9–31, doi: [10.1007/978-3-030-18468-1_2](https://doi.org/10.1007/978-3-030-18468-1_2).



ANDRÉS AHARHEL MERCADO-VELÁZQUEZ

was born in Tepic, Nayarit, Mexico. He received the B.Sc. degree in communications and electronics engineering from the National Polytechnic Institute, Mexico, in 2018, where he is currently pursuing the M.Sc. degree in computer engineering sciences with the Computing Research Centre. His research interests include the Internet of Things, cybersecurity, electronic circuit design, multi-objective optimization, and secure web application development. In 2014, he won the State Youth Award (Nayarit, México) and the "Presea Ing. Bernardo Quintana Arrijoja," both in academic excellence.



PONCIANO JORGE ESCAMILLA-AMBROSIO

(Senior Member, IEEE) received the B.Sc. degree in mechanical and electrical engineering and the M.Sc. degree (Hons.) in electrical engineering from the National Autonomous University of Mexico (UNAM), in 1995 and 2000, respectively, and the Ph.D. degree from The University of Sheffield, U.K., in January 2004. From 2003 to 2010, he was a Research Associate with the Departments of Aerospace Engineering and Computer Science, University of Bristol, U.K. From 2010 to 2011, he was a Research Associate with the Department of Electronics, National Institute of Astrophysics Optics and Electronics, Mexico. From 2011 to 2013, he was the General Director of innovation and development with the Scientific Division of the Secretariat of the Interior, Mexico. He is currently a Researcher with the Computing Research Centre, National Polytechnic Institute, Mexico. He has more than 80 publications among journals, conference proceedings, and book chapters. His research interests include cybersecurity, security in the IoT and wireless sensor networks, applications of the IoT, wireless sensor networks, and multi-sensor data fusion.



FLORIBERTO ORTIZ-RODRÍGUEZ

received the B.S. degree in electronic engineering, in 2001, and the M.S. and Ph.D. degrees in automatic control from CINVESTAV-IPN, Mexico, in 2004 and 2008, respectively. Since 2006, he has been with the Department of Communication and Electronic Engineering, Higher School of Mechanic and Electrical Engineering, IPN. His current research interests include nonlinear systems, neural networks, and fuzzy control.

...