mathematics

MDPI

Article

Dynamic Malware Mitigation Strategies for IoT Networks: A Mathematical Epidemiology Approach

Roberto Casado-Vara [1,*], Marcos Severt [2], Antonio Díaz-Longueira [3], Ángel Martín del Rey [4] and Jose Luis Calvo-Rolle [3]

[1] Grupo de Inteligencia Computacional Aplicada (GICAP), Departamento de Matemáticas y Computación, Escuela Politécnica Superior, Universidad de Burgos, Av. Cantabria s/n, 09006 Burgos, Spain
[2] Department of Computer Sciences, Universidad de Salamanca, 37007 Salamanca, Spain; marcos_ss@usal.es
[3] Department of Industrial Engineering, CTC, CITIC, University of A Coruña, Rúa Mendizábal, s/n, 15403 Ferrol, Spain; a.diazl@udc.es (A.D.-L.); jose.rolle@udc.es (J.L.C.-R.)
[4] Department of Applied Mathematics, Universidad de Salamanca, 37007 Salamanca, Spain; delrey@usal.es
* Correspondence: rccasado@ubu.es

Abstract: With the progress and evolution of the IoT, which has resulted in a rise in both the number of devices and their applications, there is a growing number of malware attacks with higher complexity. Countering the spread of malware in IoT networks is a vital aspect of cybersecurity, where mathematical modeling has proven to be a potent tool. In this study, we suggest an approach to enhance IoT security by installing security updates on IoT nodes. The proposed method employs a physically informed neural network to estimate parameters related to malware propagation. A numerical case study is conducted to evaluate the effectiveness of the mitigation strategy, and novel metrics are presented to test its efficacy. The findings suggest that the mitigation tactic involving the selection of nodes based on network characteristics is more effective than random node selection.

Keywords: malware propagation; individual-based SIR model; PINN; Inverse problem; malware mitigation; IoT networks

MSC: 92D30

Citation: Casado-Vara, R.; Severt, M.; Díaz-Longueira, A.; Rey, Á.M.d.; Calvo-Rolle, J.L. Dynamic Malware Mitigation Strategies for IoT Networks: A Mathematical Epidemiology Approach. *Mathematics* **2024**, *12*, 250. https://doi.org/10.3390/math12020250

Academic Editor: Antanas Cenys

Received: 21 November 2023
Revised: 4 January 2024
Accepted: 9 January 2024
Published: 12 January 2024

1. Introduction

The Internet of Things (IoT) represents a paradigm shift that enables the integration of billions of endpoint devices with embedded intelligence and communication capabilities [1,2]. Its applications include healthcare, home automation, transportation, smart grids, and environmental monitoring, among others. Nonetheless, the IoT faces substantial security risks from malware, which refers to malicious code, such as sensor worms, ransomware, and botnets [3]. IoT devices are at high risk of malware attacks because of inadequate regulations and neglected manufacturing security [4]. Additionally, conventional security measures cannot be easily applied to the hardware of lightweight IoT devices, like wireless sensor nodes, due to their limited processing power, memory, and energy capacity [5]. The mass deployment and restricted conditions of IoT devices exacerbate the risk of malware propagation [6]. Malicious actors can capitalize on vulnerabilities in IoT devices to proliferate malware across heterogeneous IoT networks, comprised of diverse devices with varying communication technologies and capabilities. Malware can propagate between devices, forming a botnet that is capable of executing malicious actions, such as DDoS attacks and spam email delivery [7]. A notable illustration of such an attack on Internet of Things networks is the Mirai botnet [8]. In 2016–2017, Mirai exploited vulnerabilities in IoT devices like smart cameras and routers, infecting them and launching several DDoS attacks. The Mirai botnet resulted in considerable harm to internet infrastructure and disrupted businesses and services worldwide [9].

Software that is intentionally designed to harm or damage users or computer systems is known as malware or malicious software [10]. According to Pachhala et al., there are eight main types of malware [11]: (1) A virus is a type of malicious software that attaches itself to legitimate files and replicates when those files are executed. Its purpose is to damage or alter files and programs (e.g., ILOVEYOU [12]). (2) Unlike viruses, worms can spread autonomously across networks and systems. They do not require host files. They replicate and spread without user intervention (e.g., Conficker [13]). (3) A Trojan horse is a seemingly harmless program that, once installed, allows an attacker to gain remote access to the system without the user's knowledge (e.g., Zeus [14]). (4) Spyware is a type of malware that collects information about a user's activities without their knowledge. It can log keystrokes, capture personal data, or monitor online behavior (e.g., SpyEye [15]). (5) Adware is software that displays unwanted advertisements to users, often in the form of pop-ups, banners, or advertisements. Its primary purpose is to generate revenue through unwanted advertising (e.g., Superfish [16]). (6) Ransomware is a type of malware that encrypts user files and demands a ransom for access. If not handled properly, this form of digital extortion can cause significant problems (e.g., WannaCry [17]). (7) A Rootkit is a collection of tools that enables unauthorized access to a system by concealing its presence. It can provide attackers with complete control over the system (e.g., Sony BMG Rootkit [18]). (8) A botnet is a network of compromised systems that can be remotely controlled to carry out malicious activities, such as distributed denial of service (DDoS) attacks (e.g., Mirai [8]). Malware detection and mitigation encounter new challenges due to the heterogeneity of IoT networks. Traditional methods for malware detection and mitigation are frequently developed for homogeneous networks where all devices have identical capabilities and utilize the same communication protocols [19]. Nonetheless, such techniques prove less effective in heterogeneous IoT networks where devices possess distinct capabilities and employ different communication protocols. The rising prevalence and diversity of IoT networks could result in more widespread malware outbreaks, leading to increased complexity and difficulty in managing and defending against security threats. Despite this, there is a lack of systematic research on the process of malware propagation defense in heterogeneous IoT networks, particularly concerning predicting the critical threshold for malware propagation [3]. This critical threshold determines whether malware can propagate through the network. Malware propagation is only possible when the value of the parameter exceeds this threshold. Comprehending this critical threshold is crucial to determining the daily patching rate. Furthermore, in a heterogeneous IoT network infiltrated by malware, it is essential to consider the distribution of patching probabilities among different IoT devices (known as the patching strategy), as it will affect the regular operation of the devices and the overall cost. Therefore, it is crucial to choose an economic strategy that will effectively prevent malware outbreaks.

Researchers have proposed various techniques for identifying and preventing malware in multifarious IoT networks. Certain methodologies concentrate on creating novel algorithms to detect malware in amalgamated networks, whereas others emphasize innovative methods for mitigating malware proliferation in the same. A promising approach for identifying malware in hybrid IoT networks is the application of machine learning (ML). ML algorithms can train on extensive datasets of recognized malware and non-malicious traffic patterns to distinguish between hostile and normal traffic. Malware poses a significant danger to hybrid IoT networks, which merge conventional information technologies services networks with IoT devices. Such devices are often at risk of attack due to their frequent lack of security features and patch updates. A viable means of malware detection in these networks is through the use of ML algorithms. These algorithms can be trained to recognize malevolent patterns in the network traffic. After training, network traffic can be continuously monitored in real-time to identify potential malware infections. An alternative method for detecting malware in hybrid IoT networks is through anomaly detection techniques which observe unusual or unexpected network activity. Monitoring metrics such as traffic volume, packet sizes, and packet types help detect unusual activity that

may indicate a possible malware infection. In addition to developing detection methods, researchers have created a range of strategies to combat malware in hybrid IoT networks. One common approach is network segmentation in which the network is divided into smaller, isolated networks. This method helps to control potential malware infections and prevent them from spreading to other network areas. Intrusion prevention systems (IPSs) are a widely used approach for addressing malware in hybrid IoT networks. IPSs observe network activity and prevent malignant data from reaching their intended destination. These systems can intercept various strains of malware, such as worms and botnets. Furthermore, experts are devising new tactics to combat malware in hybrid IoT networks; automatic patching techniques are one such approach that can be used for IoT devices. Updating software through patching is crucial for addressing potential security issues. Automatic patching methods can facilitate the upkeep of IoT devices with up-to-date security patches. Nevertheless, hybrid IoT networks pose a challenge for automatic patching since certain devices may be positioned in remote or inaccessible locations. Moreover, patching all devices may result in significant software costs. Consequently, it is crucial to develop optimal strategies to maintain IoT device security standards.

Mathematical modeling is an invaluable tool for uncovering the mechanisms behind malware propagation and exploring ways to implement effective patching strategies across multiple IoT networks. The use of classical epidemic models is a common approach to studying malware propagation within networks. These models divide nodes into distinct compartments based on their propagation states, such as susceptible (S), infected (I), and recovered (R). The classical model most commonly used is the susceptible-infected-recovered (SIR) model. The rise in quantity and variety of IoT networks has led to an increased range of malware incidents, rendering security management and defense more complex and demanding. Mathematical modeling provides a means of obtaining valuable insights into the mechanism of malware propagation and facilitates the development of effective mitigation measures. In this study, we analyze the propagation of malware through the implementation of various patching strategies. We have created a mathematical model to capture the dynamics of malware spread in diverse IoT networks while taking into account the influence of various patching methods. Through simulation experiments, our model is validated, and the outcomes indicate that distinct patching strategies can significantly affect the proliferation of malware in IoT networks. For instance, the findings demonstrate that a random patching approach could efficiently reduce malware propagation, although it might demand a high patching frequency to achieve an adequate level of security. Conversely, a targeted patching plan could prove more effective, but it necessitates precise awareness of vulnerability distribution in the network. Our research indicates that mathematical modeling is a viable tool for devising and assessing efficient patching tactics to mitigate malware in diverse IoT networks. The main contributions of our work are:

- We propose a strategy to mitigate the spread of malware through the installation of security updates on IoT nodes. Node selection is based on two criteria: random selection for updates, and selection based on the degree of IoT nodes.
- Since the propagation parameters of malware are unknown in real-world scenarios, we propose to estimate these parameters by solving the inverse problem with a physics-informed neural network.
- We conducted a numerical study to evaluate our proposal. Moreover, we developed new metrics to assess the effectiveness of the mitigation strategy based on node selection criteria. This assessment suggests that the IoT network-based mitigation approach is more effective.

The rest of the paper is organized as follows. Section 2 presents the literature review. Next, we present the methods in Section 3. Section 4 presents the setup of our simulations and the results. Finally, Section 5 concludes the conducted research and proposes future lines of work.

2. Mitigation Strategies

Malware propagation is a multifaceted and fluid process, which both control and security experts have examined in depth. Dynamical models based on epidemic models, for example, the Kermack–McKendrick model [20], have been developed for malware propagation. Furthermore, these models have expanded to include diverse containment methods, such as susceptible-infected-susceptible (SIS) and susceptible-infected-recovered models, as well as the proliferation of numerous malware [21,22]. Control-theoretic methods have been utilized to create effective mitigation strategies and to illustrate the propagation structures of destructive malware [23,24]. Optimal control, for instance, has been deployed to decrease patching efforts whilst also eradicating all infections caused by a singular virus. Furthermore, a quarantine-oriented mitigation plan has been researched for time-varying graphs [25], whereby the layout of a graph is employed to represent the necessary measures to overcome a resource allocation dilemma [26]. Control theory and game theory are two main approaches to containing malware in networks [27]. Solutions based on control theory are often based on heuristics and simplifications, which can lead to sub-optimal results [28]. Game theory-based solutions are gaining popularity due to their ability to minimize the damage caused by attackers at minimal cost and their relative simplicity for large networks [29]. However, current game theory solutions are focused on malware propagation minimization and do not take into account network performance constraints [30]. Mitigation strategies for unknown propagation parameters have been developed by assuming that the defender has prior knowledge of the possible range of propagation parameters. To guarantee a minimum level of performance under uncertain propagation conditions, previous research suggests the existence of predetermined patching methods [31]. However, the implementation of these strategies requires an understanding of the propagation model parameters [32]. Recovering from malware infections typically involves patching or removing the malicious code. Although this may seem like the only viable solution, it is limited by the fact that not all nodes can be patched [33–35]. Table 1 presents a review of the benefits and difficulties associated with the malware mitigation methods analyzed in the literature review.

Table 1. Summary of the literature review on the mitigation strategies.

Mitigation Strategy	Reference	Advantages	Challenges
Optimal control	[36]	• Costs can be appropriately selected. • The analyses do not consider how devices react to the policies.	• The devices are homogeneous. • The population consists of a single group of devices.
	[37]	• The devices are heterogeneous. • Global or local coverage.	• Network heterogeneity reduces the efficacy of local awareness mechanisms. • The simulations reveal aspects not captured by the heterogeneous control theory.
	[38]	• Devices decide whether to adopt protective measures (game theory). • Very effective on best response scenarios (Nash equilibrium) for different parameters.	• The devices are homogeneous. • Protection is costly and partially effective.

Table 1. *Cont.*

Mitigation Strategy	Reference	Advantages	Challenges
	[39]	• Decision-making by devices to protect themselves. • Apply evolutionary learning strategies.	• The presence of asymptomatic infectious agents necessitates adding additional compartments to the SIR model. • Parameters of the epidemic dynamics are known to the agents.
Quarantine	[40]	• Minimize the spread while maximizing the network performance. • Maintains the network performance with a minimal cost.	• The devices are homogeneous. • The model does not ensure a minimum level of network performance by restricting malware.
	[41]	• Minimum malware impact in the network. • Effective against several types of malware.	• The model has not proven its efficiency in large IoT networks. • Need simplifications that do not lead to optimal solutions.
	[40]	• Minimize the spread while maximizing the network performance. • Evaluate their model in several malware mathematical models.	• The devices are homogeneous. • High computational complexity if the network is too large.
	[42]	• Guide effectively designing a honeynet. • The honeynet potency can be explicitly enhanced by listed actions.	• The shape of the network is a key factor for determining the model potency. • Works better locally than globally (network topology dependent).
Patching	[43]	• Have a mathematical model focused on wireless IoT networks. • Optimal patching rates are obtained using approximate analysis.	• A network defender is assumed to patch the devices to avert the formation of a botnet. • Optimal policies for both the attacker and defender have been studied based on the knowledge of the attacker's behavior.
	[44]	• Bind epidemiology and topology of the network. • Support for convex network boundaries.	• High complexity. • Dependency in the number of susceptible neighbors.
	[3]	• Heterogeneous IoT networks. • Defense effectiveness provides a direct and efficient method for comparing the costs of various degree-related patching strategies.	• Malware infection is hard to avoid under the way of centralized transmission and network size. • Random immunization has a larger infection size than the case of targeted immunization.
	[45]	• Physical unclonable functions-based virtual patching protocol to contain and limit malware spread. • Faster than the existing techniques in the selected literature.	• Low scalability • High time cost.

In this study, we propose the use of automatic patching as a mitigation strategy for IoT network elements. This mitigation strategy is presented that does not require any prior knowledge of the parameters of the propagation model. Furthermore, depending on the infection observed during the detection process and the chosen strategy, the patch rate is updated. This approach is more flexible and adaptable than fixed patching strategies because it can account for changes in propagation dynamics over time. In addition, our proposal is capable of functioning in heterogeneous networks and is scalable. The selection of nodes to patch only requires knowledge of their degree. To evaluate this strategy, modifications to the parameters of the SIR propagation model are necessary. One of these parameters is the propagation rate, also known as the β parameter. This parameter is the likelihood of an IoT node becoming infected by malware. Reducing the value of β decreases the likelihood of malware infecting a vulnerable device. When a patch is installed, susceptible devices have a low likelihood of risk of infection (since $\beta \neq 0$). To implement an automated patching strategy and mitigate the spreading of malware, two methods will be adopted to select the IoT nodes for updating the software. The initial approach is a random node election strategy, which is based on the comprehensive knowledge of all the nodes in the IoT network. This technique arbitrarily updates the software of a percentage of the nodes within the network. To achieve our aim, we suggest decreasing the propagation rate of each node. This makes it more challenging for malware to spread through updated nodes. The next technique involves selecting nodes with the highest degree for software updates, based on network characteristics. To achieve this, we select the average degree $\langle k \rangle$ as a threshold (the average degree is defined in Section 3.2). This ensures that nodes with a degree $\geq \langle k \rangle$ receive software updates that reduce their beta value, which simulates the malware having a harder time spreading through these IoT network nodes.

3. Methods

In this section, we outline the method used to simulate the spread of malware in an IoT network and assess the efficacy of mitigation techniques. We utilize an individual-based (IB) SIR model to depict malware propagation and a physics-informed neural network (PINN) to approximate the IB SIR model's parameters. We also analyze the impact of security updates with varied strategies as a possible workaround. The PINN is an artificial intelligence model that approximates functions dependent on continuous variables. In this instance, the PINNs are utilized to estimate the parameters of the SIR model, depicting the infection and recovery rates (since we assume that in a real-life scenario, the parameters of the malware are unknown we should estimate them). To assess the efficacy of the mitigation measures, we simulate malware propagation in an IoT network under both circumstances (with and without mitigation measures). The number of devices infected in every simulation is compared to analyze the effectiveness of the mitigation measures.

3.1. Propagation Model for Malware Based on the SIR Classic Model

3.1.1. The Kermack and McKendrick SIR Model

The Kermack and McKendrick model, initially introduced in 1927 [20], elucidates the transmission of communicable illnesses. It is generally known as the SIR model as it comprises three categories: susceptible, infectious, and recovered. The amount of members in each group is represented by a time-based function [46]. $S(t)$ denotes individuals who are susceptible to the disease, $I(t)$ signifies those who are infectious and capable of spreading the disease, whereas $R(t)$ indicates the number of individuals who have either recovered or passed away due to the disease. The removed individuals have either recovered, gained immunity, or have been isolated until recovered. According to the SIR model, removed individuals are no longer susceptible to or infectious with the disease. In the conventional model, it is assumed that the overall population N remains constant, resulting in $N = S(t) + I(t) + R(t)$. Murray et al. offer a comprehensive description of the SIR model, which includes the classical version that assumes constant rates [47]. The differential equation governing the change in $S(t)$ is $-\beta S(t)I(t)$, where $\beta > 0$. Infectious

persons exit $I(t)$ at a rate γ and directly transition into the $R(t)$ category. At the moment, cases of reinfection in the spread of malware within IoT networks are few and far between and their rate cannot be estimated. As a result, the SIR model does not consider this possibility. However, in the case of a botnet, rates can fluctuate depending on several factors, including the use of firewalls, IoT network connectivity restrictions implemented by administrators, and preventative measures such as updates [48]. The SIR model can be represented by the following system of differential equations:

$$\frac{dS}{dt} = -\beta SI \tag{1}$$

$$\frac{dI}{dt} = \beta SI - \gamma I \tag{2}$$

$$\frac{dR}{dt} = \gamma I \tag{3}$$

$$S(0) = S_0, I(0) = I_0, R(0) = 1 - S_0 - I_0 \tag{4}$$

3.1.2. Individual-Based Stochastic SIR Model

Malware propagation through an IoT network can be modeled using a cellular automaton (CA) on a graph $G = (V, E)$, where V is the set of nodes (vertices) and E is the set of edges connecting the nodes. Each node v_i in V represents an entity, such as a computer or an IoT device, and the edges e_{ij} in E represent the relationships between entities. The set S represents the allowed states for a node, such as susceptible (S), infected (I), or recovered (R). The state of each node v_i evolves according to a transition function $f : S^{|N(v_i)|} \longrightarrow S$ that depends on the states of the neighboring nodes $N(v_i)$. This iterative process is applied to all nodes at each time step. Formally, for a given node v_i at time $t + 1$ the transition rule is as follows:

$$s_i(t + 1) = f(s_{j_1}(t), s_{j_2}(t), \ldots, s_{j_k}(t)) \tag{5}$$

where $j_1, j_2, \ldots, j_k$ denotes the neighbor of v_i. This process is repeated across all nodes at each time step.

To simulate the spread of malware, the transition function f can include parameters that define infection probabilities, recovery rates, and other relevant factors. This allows the cellular automaton implemented on the graph to simulate the spread of malware accurately. Martin del Rey et al. presented transition rules in their paper, which we have implemented to simulate the propagation of malware [49]. Several assumptions need to be considered before outlining the transmission functions:

- A susceptible device can become infected depending on the infection rate, β_i, of each IoT node, where $0 \leq \beta_i \leq 1$.
- An infected device can be recovered, depending on the recovery rate, γ_i, of each IoT node, where $0 \leq \gamma_i \leq 1$.
- This model considers permanent immunity, preventing the malware from infecting recovered devices.

Consequently:

If $s_i^t = (S)$:

$$s_i^{t+1} = \begin{cases} I, \text{ with probability } \beta_i \\ S, \text{ with probability } 1 - \beta_i \end{cases} \tag{6}$$

If $s_i^t = (I)$:

$$s_i^{t+1} = \begin{cases} R, \text{ with probability } \gamma_i \\ I, \text{ with probability } 1 - \gamma_i \end{cases} \tag{7}$$

If $s_i^t = R$, then $s_i^{t+1} = R$

Algorithm 1 contains the pseudo-code for the transmission functions.

Algorithm 1 IB malware spread transition rules pseudocode

```
for each node in infected_nodes do
    neighbors ← list(graph.neighbors(node))
    for each neighbor in neighbors do
        if neighbor not in infected_nodes and neighbor not in recovered_nodes then
            if random.random() < node_transmission_rates[neighbor] then
                if len(new_infected) < max_new_infected then
                    new_infected.append(neighbor)
                end if
            end if
        end if
    end for
    if random.random() < recovery_rates[node] and node not in new_infected then
        new_recovered.append(node)
    end if
end for
```

3.2. A Complex Network Approach to IoT Network

This section outlines how we create complex networks for simulating IoT networks, specifically random networks. We also demonstrate the connectivity of these random networks. In addition, we will force the random networks that are created to be connected. A random network is a network where each node's degree is a random variable, denoted as $k_i \in [0, N]$. The analysis of random networks began systematically in 1959 when mathematicians Erdős and Rényi started studying the properties of graphs as they are affected by the addition of connections at random, using probabilistic methods [50]. A significant breakthrough was achieved in the conventional mathematical theory of graphs that transformed the modeling of problems. The new approach utilized random graphs to depict complex network topology, forming the basis of random network theory. In the second half of the twentieth century, the Erdős and Rényi model prevailed as the primary logical and rigorous approach influencing scientists' thinking about complex networks, even though it is clear that many complex networks in the real world are not entirely regular or random. The term "random graph" refers to the random positioning of edges between different nodes in the ER graph. According to their first paper [50], Erdős and Rényi introduced a mechanism for producing random graphs having N nodes and L edges. We shall henceforth refer to it as the Erdős–Rényi random network (ER). To construct the graph, we begin by creating N unconnected nodes. Afterward, we select pairs of nodes at random and connect them through edges, making sure to avoid any possibility of multiple edges between nodes. We continue connecting nodes in this way until we have L edges in total. By using this method, we can generate one of many possible graphs containing N nodes and L edges. This forms part of the statistical set of all combinations of L edges that can exist. For a comprehensive explanation of the ER model, we require the complete statistical set of all conceivable realizations in the matrix representation of the set of adjacency matrices. To conclude this section, we shall clarify the terms node degree, average degree, and clustering coefficient.

- Degree of a node: In an undirected network, a node's degree is defined as the total number of members connected to it and denoted by k_i. In a simple undirected graph $k_i \in [0, N-1]$.
- The average degree is defined by the following equation $\langle k \rangle = \frac{1}{N} \sum_i k_i$.

NetworkX is a Python package used for exploring and analyzing networks and network algorithms. The core package provides data structures for representing various types of networks, including simple graphs, directed graphs, and graphs with parallel edges

and self-loops. The nodes in NetworkX graphs can be any hashable Python object, and edges can contain arbitrary data. This flexibility makes NetworkX suitable for representing networks found in various scientific fields. NetworkX implements various graph algorithms for calculating network properties and structure measures, including shortest paths, betweenness centrality, clustering, and degree distribution. It can read and write various graph formats for easy data exchange and provides generators for many classic and popular graph models, such as the Erdős–Rényi, Small World, and Barabási–Albert models. The Python programming language's ease of use and flexibility, combined with its connection to the SciPy tools, make NetworkX a powerful tool for scientific computations [51]. NetworkX is a versatile tool for modeling various types of networks, such as social networks, communication networks, transport networks, biological networks, and IoT networks. In this research, NetworkX was used to model the spread of malware in an IoT network by creating graphs using library functions for simulations (see Figure 1).

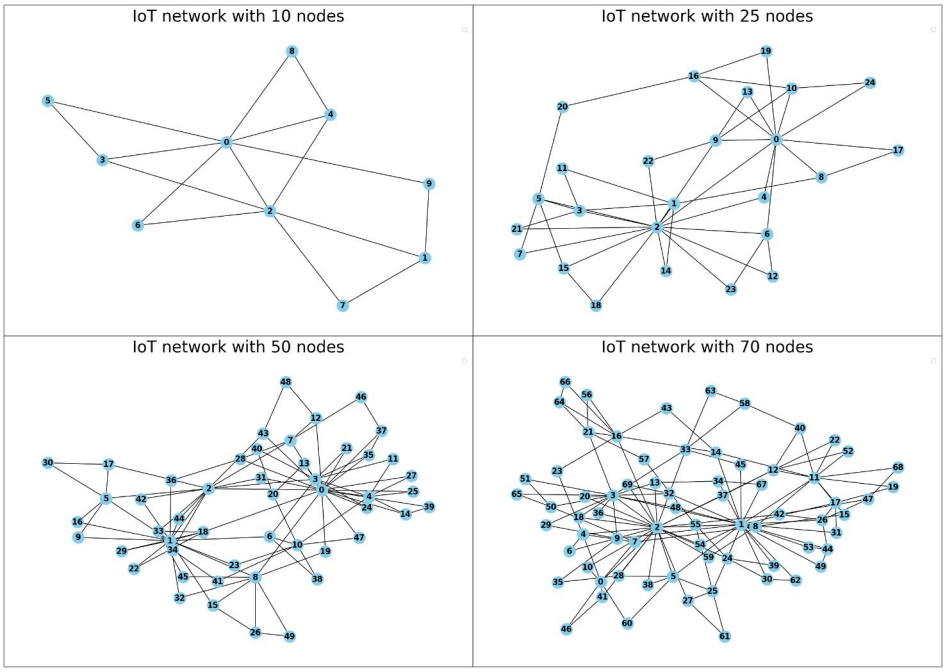

Figure 1. Examples of IoT networks generated with the NetworkX powerlaw_cluster_graph function with a different number of nodes.

3.3. Evaluation Metrics

In this section, we present the criteria for objectively evaluating the effectiveness of the mitigation strategies proposed and determining the optimal approach. Our initial evaluation measure is the global infection rate. As nodes in IoT networks are typically heterogeneous and, thus, show distinct characteristics, we randomize the β parameter of each node in each simulation to simulate the heterogeneity. Considering this, we define the global infection rate as the average of the β parameters of all the nodes in the IoT network. To determine the global infection rate, we solve the inverse problem detailed in the next section, employing PINNs at every time step. This approach will provide us with the time evolution of this metric in all simulations. We also employ a metric that determines the percentage of IoT devices that have remained susceptible to malware, the percentage of final susceptibles (PFS). This metric evaluates the initial state of such devices and determines if they have ever been infected or remained untouched due to the malware's inability to infect them. The significance of this metric lies in identifying the number of devices that have never encountered malware or, if they have, have remained inactive. Finally, we suggest using the average recovery speed and average infected speed as measures. These measures calculate the average speed at which devices recover from infection over a specific period

and the average speed at which devices are infected over a specific period. The proposed evaluation metrics in this work are described below:

- Global infection rate: This rate represents the average infection rate from a susceptible device to an infected device. The global infection rate is calculated as the sum of the betas of all devices divided by the total number of devices.

$$\text{Global infection rate} = \frac{1}{N}\sum_{i=1}^{N}\beta_i \tag{8}$$

  where β_i is the infection rate of the device i and N is the number of devices in the IoT network. In this case, we are solving an inverse problem with one PINN for the whole network as the β_i of each device is not fully known at time t.
- Percentage of final susceptibles: The final susceptibles percentage is a metric used to evaluate the proportion of devices that remain uninfected at the end of a simulation. It is calculated by dividing the number of uninfected devices by the total number of devices. A high PFS indicates that mitigation was effective in preventing device infection, while a low PFS indicates that mitigation was less effective or that the malware was highly contagious.

$$\text{Percentage of final susceptibles} = \frac{\text{Susceptible devices}}{N} \tag{9}$$

- Average infected speed: The 'Average infected Speed' metric quantifies the rate at which malware infects devices within the IoT network. It represents the average number of infected devices per time step. It is calculated as follows:

$$\text{Average infected speed} = \frac{n\text{-th discrete difference (Infected devices array)}}{\text{time step}} \tag{10}$$

  The 'Average infected Speed' metric is a valuable tool for evaluating the performance of a mathematical model that predicts the spread of malware in an IoT network. A high average infection rate indicates that the model predicts rapid malware spread throughout the network.
- Average recovery speed: The metric 'Average recovery speed' measures the rate at which infected devices recover from malware and are no longer infectious. It is calculated as follows:

$$\text{Average recovery speed} = \frac{n\text{-th discrete difference (Recovery devices array)}}{\text{time step}} \tag{11}$$

3.4. The Inverse Problem for the Parameter Estimation

The estimation of SIR model parameters from observation data in IoT networks is a significant parameter estimation problem in mathematical modeling. The inverse problem involves estimating the model parameters from data that result from these parameters in the context of malware propagation. When dealing with the propagation of malware in IoT networks, it is possible to use the infection and recovery rate of IoT network devices as data. This section will introduce PINNs for parameter estimation in the SIR model, along with an explanation of their theoretical basis [52–55].

Solving the Inverse Problem for a SIR Model

Let $t \in \mathbb{R}_+$ be the input of the PINN, and set $f(t;\theta) \in \mathbb{R}_+^{m+1}$ the output of the PINN, where m is the number of hidden layers. Based on the compartmental SIR Kermack and

McKendrick model [20], the PINN model has the basic three-compartment SIR model with their parameters. Consequently, the PINN output is given by:

$$f(t;\theta) = \begin{bmatrix} f_1(t;\theta) \\ f_2(t;\theta) \end{bmatrix} \tag{12}$$

where $f_1(t;\theta)$, and $f_2(t;\theta)$ approximate the number of susceptible devices, $S(t)$, and the number of infectious devices, $I(t)$, respectively. Notice that the total amount of recovered devices, $R(t)$, is completely determined by the others since it is supposed that the population remains constant: $R(t) = N - S(t) - I(t)$. As a consequence, the computational complexity can be reduced by considering only two variables, for example, $S(t)$ and $I(t)$ [55]:

$$\frac{dS}{dt} = -\frac{\beta}{N}SI, \tag{13}$$

$$\frac{dI}{dt} = \frac{\beta}{N}SI - \gamma I, \tag{14}$$

where β is the infection rate and γ is the recovery rate. Assuming that no data are available for compartments of susceptibles S, and recovered R, and that $\{u_k\}_{k=0}^{K}$ is a discrete-time series of observations in compartment I at time t_k, the MSE data loss is defined as follows:

$$MSE_{data} = \frac{1}{K+1}\sum_{k=0}^{K}(u_k - f_2(t_k;\theta))^2 \tag{15}$$

where MSE_{data} is the loss function.

The inverse problem can be described as follows: Given an incomplete dataset, PINN aims to learn a mapping from time (t) to each of the state variables in the existing model. Thus, using the incomplete dataset, we can extrapolate the unknown time series of the S and R compartments, and learn the transmission dynamics represented by the values of the SIR parameters β and γ. The PINN must access information from the pre-existing model during training (i.e., the SIR model). As a consequence, the subsystem can be written as

$$G\left(y, \frac{dy}{dt}; \lambda\right) = \frac{dy}{dt} + N[y] = 0, \tag{16}$$

where $N[\cdot]$ is, in general, a differential operator (however, in the case of ordinary differential equations, it is also possible for $N[\cdot]$ to represent a nonlinear function of the variable y), and

$$y(t) := \begin{bmatrix} S(t) \\ I(t) \end{bmatrix}, \quad \frac{dy}{dt} = \begin{bmatrix} \frac{dS(t)}{dt} \\ \frac{dI(t)}{dt} \end{bmatrix}, \quad N[y] = \begin{bmatrix} \frac{\beta}{N}SI \\ -\frac{\beta}{N}SI + \gamma I \end{bmatrix}. \tag{17}$$

If $N[y;\lambda]$ depends on $\lambda = (\beta, \gamma)^\intercal \in \mathbb{R}^2$ with λ unknown a priori, then it is as follows:

$$G(y, y_t; \lambda) = y_t + N[y;\lambda], \quad t \in [0, T]. \tag{18}$$

Therefore, in order to train the PINN effectively, we need to minimize the targets of the following form:

$$\min_{\theta,\lambda}(MSE_{data}(\theta) + MSE_G(\theta,\lambda)), \tag{19}$$

enabling the PINN to learn model parameters from data [55,56].

Note that, if the system has initial conditions, the function to be minimized is

$$\min_{\theta,\lambda}(MSE_{data}(\theta) + MSE_G(\theta,\lambda)) + MSE_{IC}(\theta) \tag{20}$$

Moreover, note that we have constrained the parameters to be time-independent. If they were time-dependent, it would be necessary to create a sliding window with an amplitude of $\alpha\Delta t$ with $\alpha \in \mathbb{R}$ to be passed as input to the PINN small frames as follows $(t, t + \alpha\Delta t)$.

4. Results

This section outlines the simulation setup and tests to assess the effectiveness of the mitigation measure using randomized and network-based node election strategies. Several simulations were conducted to compare three scenarios: (1) the spread of malware through the IoT network, (2) the spread of malware through the IoT network with a random node selection mitigation strategy, and (3) the spread of malware through the IoT network with the mitigation strategy based on the degree of each node.

4.1. Simulation Setup

The simulations were conducted in Python with the deepxde library to code the PINNs [57] and NetworkX to code the complex networks [51] on a desktop (CPU: Intel (R) Core (TM) i7-8700 CPU @ 3.20 GHz, Memory 16 GB, Os: Microsoft Windows 10 with 64 bits). The following parameters were common to all of our simulations. The IoT network was simulated with a network of 70 nodes, and in the creation of the network, 2 new connections were added with a probability of 0.5 of reconnection between the existing nodes in the network. On the other hand, the SIR model to be propagated through the IoT network had a fixed reconnection rate for all nodes in the network of 0.2, while the reconnection rate for each of the IoT nodes in the network is randomly initialized in the range of 0.1–0.8. The IoT network was modeled as a random graph using the powerlaw_cluster_graph function of NetworkX. This model enables the creation of networks with a heavy-tailed degree distribution, which is typical of real networks like IoT networks. This means that some nodes have a large number of connections, while others have very few. This type of degree distribution is common in real networks, such as social and communication networks. The reconnection probability of 0.5 indicates that, on average, 50% of new connections will be established between existing nodes in the network, while the other 50% will be established between new nodes. The fixed reconnection rate of 0.2 indicates that, on average, 20% of the existing nodes in the network will reconnect with a new node at each time step. To ensure consistency in each simulation, the graph.copy() function from NetworkX was used to maintain the same graph. This allows for a fair comparison of malware spread across the network. On the other hand, the proposed neural network architecture, PINN, consists of an input layer with a single neuron, followed by two hidden layers, each containing 40 neurons, and an output layer with three neurons. The internal parameters of each layer are initialized using the Glorot uniform function, the optimizer employed is Adam with a learning rate (LR) of 0.001 and the activation function in all the neurons is Tanh. The PINN receives a tuple (time, S(t), I(t), R(t)) at the input layer, which contains actual or simulated data on the spread of malware in the IoT network. When solving an inverse problem in PINN, the parameters to be estimated (γ-recovery rate and β-infection rate, in this case) are the target of the PINN (see Equation (18)). The output layer of the PINN will return these two estimated parameters. Finally, the PINN was trained for 20,000 epochs. An overview of the PINN architecture is shown in Table 2. The decision has been made to use the same PINN configuration for all simulations without optimizing the PINN hyperparameters for each simulation. This is because it is a theoretical approach to proposing mitigation measures to stop the spread of malware, and we did not want parameter estimation to have a significant impact on the results.

To test which of the node selection strategies for automatic network patching has the best performance, the following simulations were defined:

- In Simulation 1, we aim to examine how the number of nodes selected using the random election strategy for patching impacts the spread of malware. The potential choices constitute 25%, 50%, and 75% of the entire nodes. Additionally, we assume early detection of malware and implementation of mitigation measures after a brief

delay, specifically in time step 4. Finally, we assume the installed security patch to be highly effective, with beta_patched_node $= \frac{\beta_i}{3}$.

- In Simulation 2, we aim to observe the impact of early detection and subsequent security patching on malware propagation dynamics. By evaluating various early time steps = $\{1, 3, 5\}$, we can assess the mitigation measures' performance with the two-node election strategies. Additionally, for this simulation, we will choose 25% of the nodes using the random strategy. Finally, it is assumed that the installed security patch is highly effective, with beta_patched_node $= \frac{\beta_i}{3}$.

Table 2. Hyperparameter of the PINN in all the simulations.

Hyperparameter	Value
Input neuron number	1
PINN input	(time, S(t), I(t), R(t)))
Output neuron number	2
PINN output	(global-β,global-γ)
Hidden layers	3
Optimizer & LR	Adam with lr = 0.001
Activation function	Tanh
Epochs	20,000

4.2. Simulation Results

This section details the outcomes of the initial two simulations. Furthermore, three supplementary simulations were conducted for each of the primary simulations, yielding diverse options for validating the effectiveness of the proposed mitigation measures. The six simulations utilized to demonstrate the findings were carefully monitored for statistical anomalies, ensuring that none of the mitigation strategies were favored and that the comparison was valid. Each simulation was run at least thirty times to guarantee accuracy.

4.2.1. Simulation 1.1 with 25% Nodes, Patching $= \frac{\beta}{3}$, Early Patching Time Step $= 4$

In this simulation, it can be observed that the number of infected IoT nodes throughout the malware propagation was less in the scenario where mitigation with node selection strategy based on the nodes' degree was applied. In Figure 2, it can be observed that the curve associated with malware propagation with mitigation and the node selection strategy based on degree is lower than in the other two cases starting from time step 4. This difference is between 10 and 15 nodes compared to the other curves of infected individuals. However, at time step 20, the three curves of infected individuals have similar values. This is due to nodes in IoT transitioning to a recovered state in the three evaluated options in this simulation, causing a sharp increase in the maximum number of nodes. This indicates that the spread of the pandemic has been halted. This means that the pandemic has been halted. As shown in Figure 3, the propagation of malware, to which the automatic patching technique has been applied but with a node selection strategy based on degree, has a lower number of infected IoT nodes from time steps 4 to 18. At this point, the nodes start changing from the infected to the recovered state.

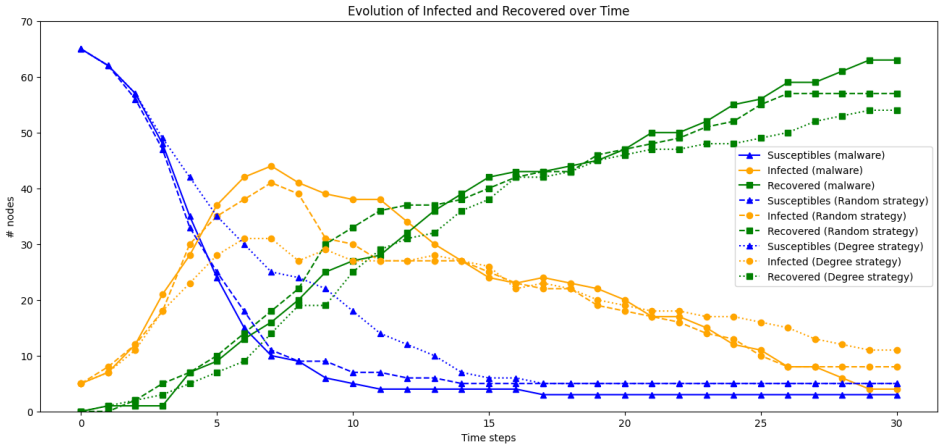

Figure 2. The figure illustrates the spread of malware in an IoT network using yellow, green, and blue curves to represent infected, recovered, and susceptible devices. The simulation includes 25% of the nodes with an infection rate of $\frac{\beta}{3}$, and it was patched in the fourth time step.

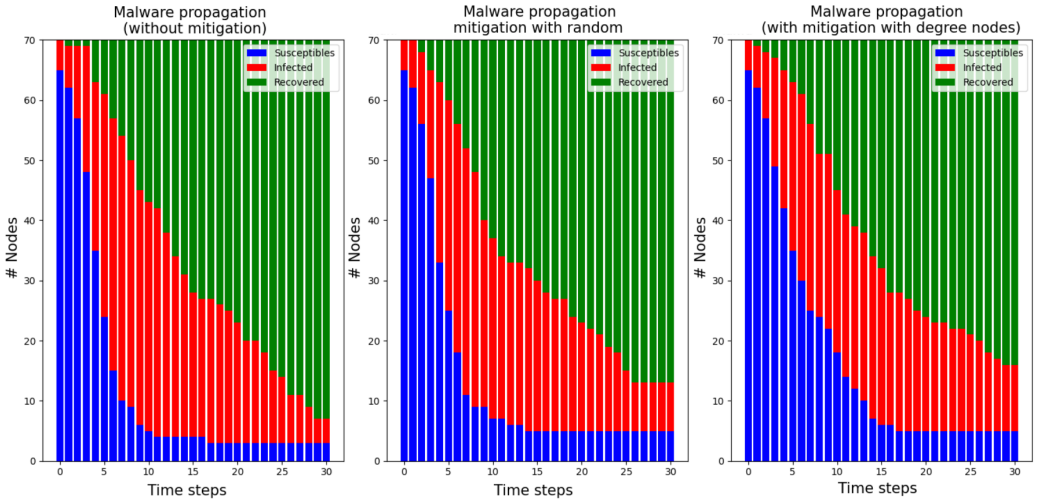

Figure 3. The evolution of malware in an IoT network is demonstrated over time. Stacked bars in red, blue, and green show the numbers of infected, susceptible, and recovered, respectively. This simulation includes 25% of the nodes with a new infection rate of $\frac{\beta}{3}$, which are patched at the fourth time step.

Based on the established metrics for assessing the efficiency of the proposed mitigation measures utilizing both strategies, Table 3 shows that the global infection rate was 0.46 at the initial time across the three simulations. At time step 4, upon patching nodes from both strategies, the rates of infection decreased to 0.37 and 0.31 for the random and degree-based node selection strategies, respectively. In both instances, the mitigation measure ensures that 7.14% of network nodes are protected from malware infection. Furthermore, the implementation of a node election strategy which is based on degree, ultimately reduces the average infection rate to 1.8 nodes per time step. This rate is lower than the infection rate of 2.3 nodes per time step in the random node election scenario and 2.7 nodes per time step when there is no mitigation measure in place. Note that the variance for node selection based on degree is 3.03, while in other situations, it is considerably higher, even doubling for random node selection. With regard to the average recovery speed, it is worth noting that it equals 1.8 nodes per time step, which is equal to the average infection speed. Based on this simulation, it appears that once the widespread propagation throughout the IoT network is halted, the same nodes are reinfected on average each time step, considering the constraint that a recovered node cannot be infected again. This indicates that the most efficient strategy for node selection, given the recovery rates and patching application, is

based on degree. Note that these findings rely on a simulation conducted with a graph generated at random according to the parameters described in this section. The results found in both malware propagation and simulation with mitigation measures and node selection strategy based on degree should be similar. Only minimal variations can be expected in the mitigation strategy with random node selection due to the increase of patched nodes.

Table 3. Summary of the metrics of simulations 1.1, 1.2, and 1.3.

Simulation #	Metric	Malware without Mitigation	Malware with Mitigation and Random Node Selection	Malware with Mitigation and Degree Node Selection
Simulation 1.1	Global infection rate patching strategy	0.46	0.37	0.31
	Percentage of final susceptibles	4.29	7.14	7.14
	Average infection speed	2.7	2.3	1.8
	Std infection speed	2.33	2.62	1.74
	Var infection speed susceptibles	5.41	6.88	3.03
	Average recovery speed	2.1	1.9	1.8
	Std recovery speed	1.6	1.72	1.6
	Var recovery speed	2.56	2.96	2.56
Simulation 1.2	Global infection rate patching strategy	0.44	0.28	0.23
	Percentage of final susceptibles	0.0	8.57	10.0
	Average infection speed	2.67	2.27	2.17
	Std infection speed	3.06	2.88	2.74
	Var infection speed susceptibles	9.36	8.27	7.52
	Average recovery speed	1.9	1.87	1.87
	Std recovery speed	2.13	1.78	2.22
	Var recovery speed	4.56	3.18	4.92
Simulation 1.3	Global infection rate patching strategy	0.46	0.23	0.2
	Percentage of final susceptibles	0.0	11.43	15.43
	Average infection speed	2.77	2.17	2.12
	Std infection speed	3.4	2.46	1.79
	Var infection speed susceptibles	11.58	6.07	5.21
	Average recovery speed	2.0	1.73	1.67
	Std recovery speed	1.95	1.53	1.48
	Var recovery speed	3.8	2.33	2.18

4.2.2. Simulation 1.2 with 50% Nodes, Patching $= \frac{\beta}{3}$, Early Patching Time Step $= 4$

During the second simulation, 50% of the nodes in the IoT network underwent a security update to tackle the detected malware. The results show that the spread of malware decreases when the mitigation strategy involves randomly selecting nodes for the update, especially from time step 4, where the patch is installed (see Figures 4 and 5). However, the curve of infected devices from time step 7 onward remains comparable to the spread of malware when no mitigation measures are taken. The mitigation strategy with degree-based node selection exhibits the same pattern, but only after time step 10. On the other hand, the growth of infected nodes during the early stages of the malware outbreak is significantly mitigated with the latter strategy.

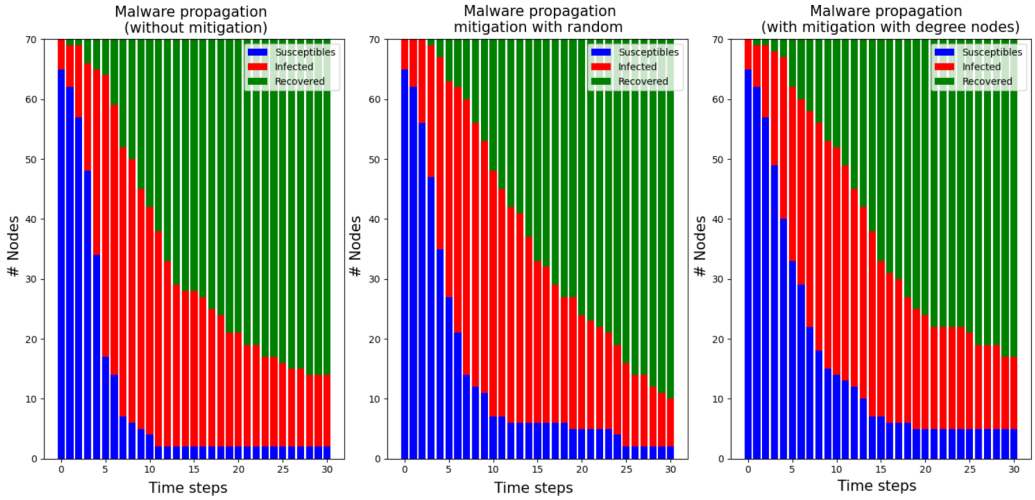

Figure 4. Malware evolution over time in an IoT network. Stacked bars in red, blue, and green represent infected, susceptible, and recovered nodes respectively. The simulation includes 50% of the nodes with a new infection rate of $\frac{\beta}{3}$, which are patched at the fourth time step.

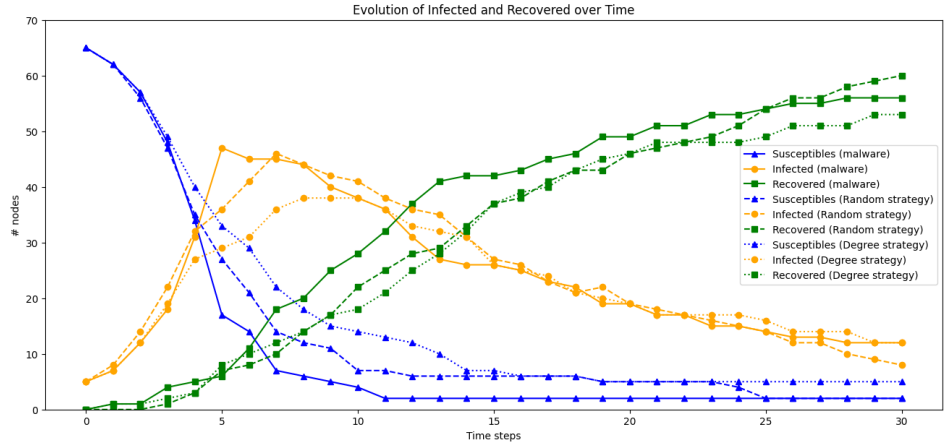

Figure 5. The evolution of malware propagation within an IoT network is depicted through yellow, green, and blue curves, representing infected, recovered, and susceptible devices. The simulation includes 50% of the nodes with an infection rate of $\frac{\beta}{3}$, and it was patched in the fourth time step.

The analysis of chosen metrics for assessing the efficacy of mitigation measures in Table 3 reveals negligible variance between the two node selection techniques and the extent of malware propagation. Therefore, it is apparent that different node selection strategies yield comparable results in mitigating the malware.

4.2.3. Simulation 1.3 with 75% Nodes, Patching $= \frac{\beta}{3}$, Early Patching Time Step $= 4$

For this simulation, the mitigation strategy using random node selection patches 75% of the nodes, whereas the mitigating strategy using a degree-based node selection strategy patches the same number. Figures 6 and 7 illustrate that both strategies perform similarly.

Regarding the statistical indicators presented in Table 3, it is evident that simulations applying various node selection strategies with mitigation techniques achieved similar performances.

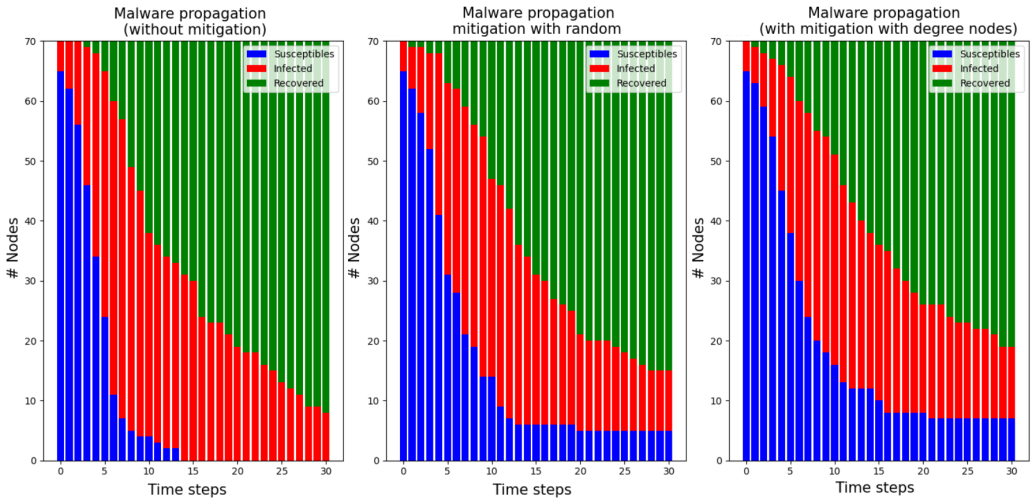

Figure 6. This figure shows how malware is evolving in an IoT network over time. The number of infected, susceptible and recovered nodes are shown as stacked bars in red, blue and green respectively. This simulation includes 75% of the nodes with a new infection rate of $\frac{\beta}{3}$, which are patched at the fourth time step.

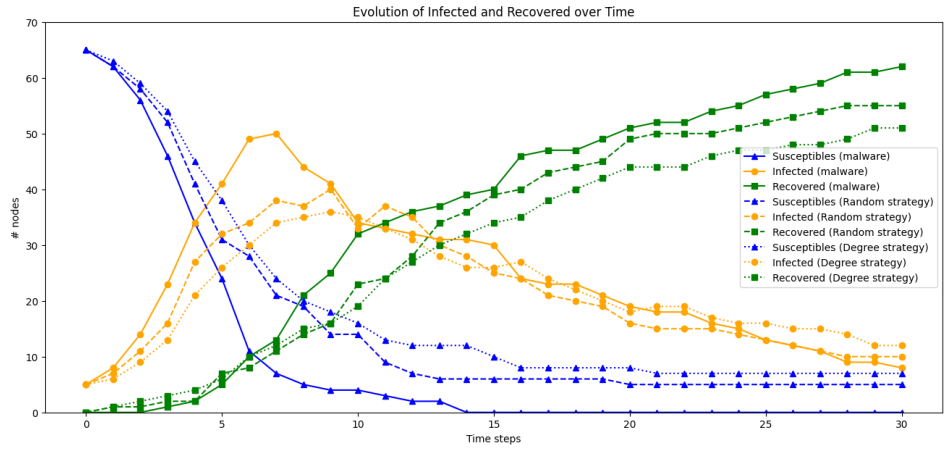

Figure 7. The figure shows the propagation of malware in an IoT network, with the yellow, green, and blue curves representing infected, recovered, and susceptible devices, respectively. The simulation includes 75% of the nodes with an infection rate of $\frac{\beta}{3}$, and it was patched in the fourth time step.

4.2.4. Simulation 2.1 with 25% Nodes, Patching $= \frac{\beta}{3}$, Early Patching Time Step $= 1$

In this simulation, the malware is detected at an early stage, and the automatic patching starts from time step 1. The mitigation strategies, including random node election and degree-based node election, exhibit similar performance and notably reduce malware propagation from time step 2. Although the degree-based node election criterion reduces the performance of the mitigation strategy in the first stage of the simulation until time step 10, it leads to a 20% reduction in the number of infected nodes on average compared to the mitigation strategy with random node election. Additionally, this strategy results in a 50% reduction of infected nodes on average compared to the spread of malware without any mitigation measures (see Figures 8 and 9).

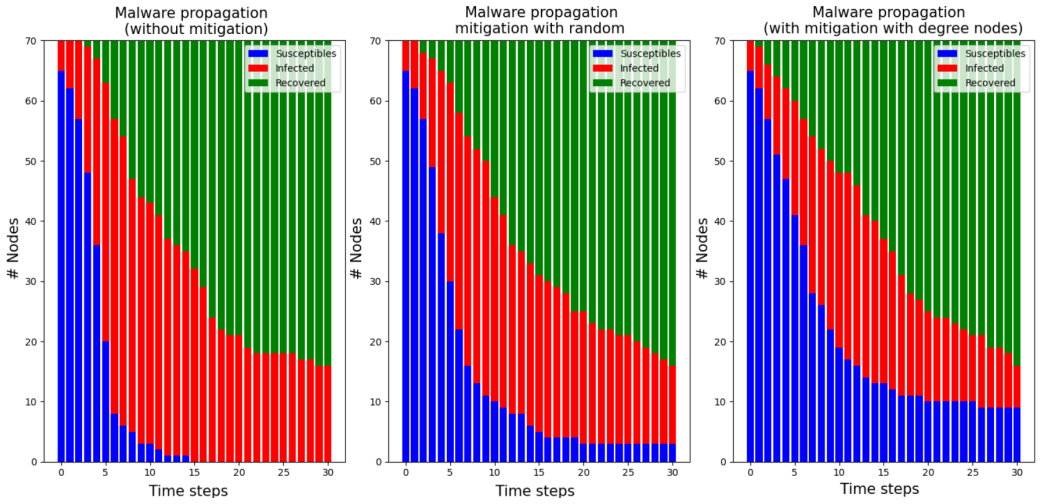

Figure 8. This picture describes the evolution of malware in an IoT network over time. Infected, susceptible, and recovered nodes are represented by the stacked bars in red, blue, and green respectively. The simulation involves 25% of the nodes with a new infection rate of $\frac{\beta}{3}$, which are patched at the first time step.

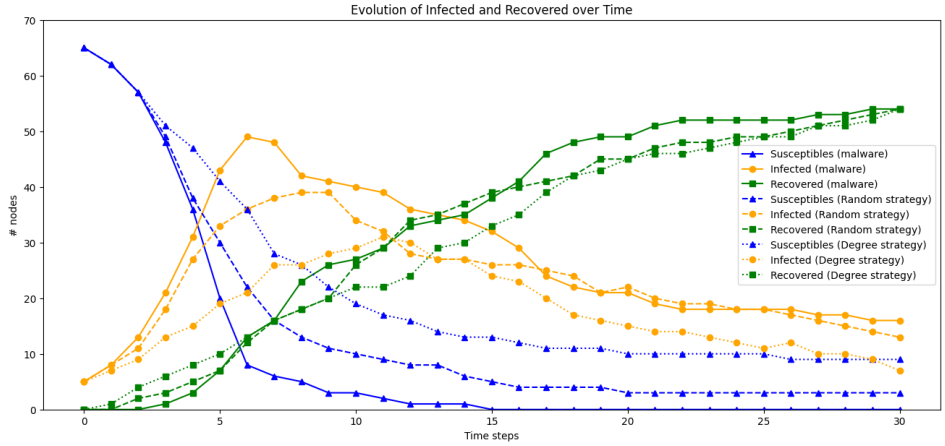

Figure 9. The evolution of malware propagation within an IoT network is depicted through yellow, green, and blue curves, representing infected, recovered, and susceptible devices. The simulation includes 25% of the nodes with an infection rate of $\frac{\beta}{3}$, and it was patched in the first time step.

The statistical analysis of this simulation has proven that—using the defined metrics—the mitigation strategy with degree-based node election is the most efficient. It should be noted that the strategy based on degree-based node election ensures that 13% of nodes remain in a susceptible state at the end of the simulation, while other simulations demonstrate at best 4.29% (see Table 4). The average infection speed for the mitigation strategy with degree-based node election is 1.73 nodes per time step, while for the random node election strategy, it is 2.07 nodes per time step. In the absence of the mitigation strategy, the malware propagation speed is 2.57 nodes per time step. These results show that the mitigation strategy successfully slows down malware propagation.

Table 4. Summary of the metrics of simulations 2.1, 2.2, and 2.3.

Simulation #	Metric	Malware without Mitigation	Malware with Mitigation and Random Node Selection	Malware with Mitigation and Degree Node Selection
Simulation 2.1	Global infection rate patching strategy	0.43	0.35	0.3
	Percentage of final susceptibles	0.0	4.29	12.86
	Average infection speed	2.57	2.07	1.73
	Std infection speed	3.06	2.16	1.24
	Var infection speed susceptibles	9.38	4.66	1.53
	Average recovery speed	1.8	1.8	1.8
	Std recovery speed	1.87	1.49	1.17
	Var recovery speed	3.49	2.23	1.36
Simulation 2.2	Global infection rate patching strategy	0.48	0.4	0.34
	Percentage of final susceptibles	4.29	4.29	4.29
	Average infection speed	2.07	2.03	1.83
	Std infection speed	2.28	2.69	2.28
	Var infection speed susceptibles	5.2	7.23	5.21
	Average recovery speed	1.87	1.83	1.77
	Std recovery speed	1.61	1.53	1.61
	Var recovery speed	2.58	2.34	2.58
Simulation 2.3	Global infection rate patching strategy	0.45	0.37	0.32
	Percentage of final susceptibles	5.71	2.86	2.79
	Average infection speed	2.5	2.43	2.53
	Std infection speed	3.11	2.67	2.81
	Var infection speed susceptibles	9.65	7.11	7.92
	Average recovery speed	1.87	2.07	1.93
	Std recovery speed	1.91	1.71	1.59
	Var recovery speed	3.65	2.93	2.53

4.2.5. Simulation 2.2 with 25% Nodes, Patching $= \frac{\beta}{3}$, Early Patching Time Step $= 3$

In this simulation, the malware was detected early. However, the automatic patching system experiences a two-time step delay, resulting in the security update being implemented at time step 3. In this scenario, it is apparent from Figure 10 that the mitigation strategy involving random node selection and the spread of malware without any implementation of mitigation measures displays a similar behavior. However, the strategy based on node selection according to the degree outperforms it, ensuring that the number of infected nodes remains below 40% of the total network. Figure 11 reveals that in none of the three cases could a high number of IoT nodes maintain the susceptible state, indicating that the malware did not infect them. Furthermore, node recovery behavior was similar across all three cases.

The statistical analysis of this simulation reveals in Table 4 that the mitigation strategy based on node selection by their degree has performed the best in the chosen metrics. It can be observed in Table 4 that the average infection speed is 1.83 nodes per time step, whereas the average recovery speed is 1.77 nodes per time step. On average, two nodes are infected and two nodes are recovered at each time step, indicating a balanced system. Due to this fact, it can be explained that the spread did not infect more than 40% of the network nodes

at any point during the simulation. Although in terms of the number of infected nodes, the mitigation strategy based on random node selection performs worse than the strategy based on node selection based on degree, from a statistical point of view, it can be seen that the average infection speed is 2.03 nodes/time step while the average recovery speed is 1.83 nodes/time step, having a behavior, on average, similar to the other strategy.

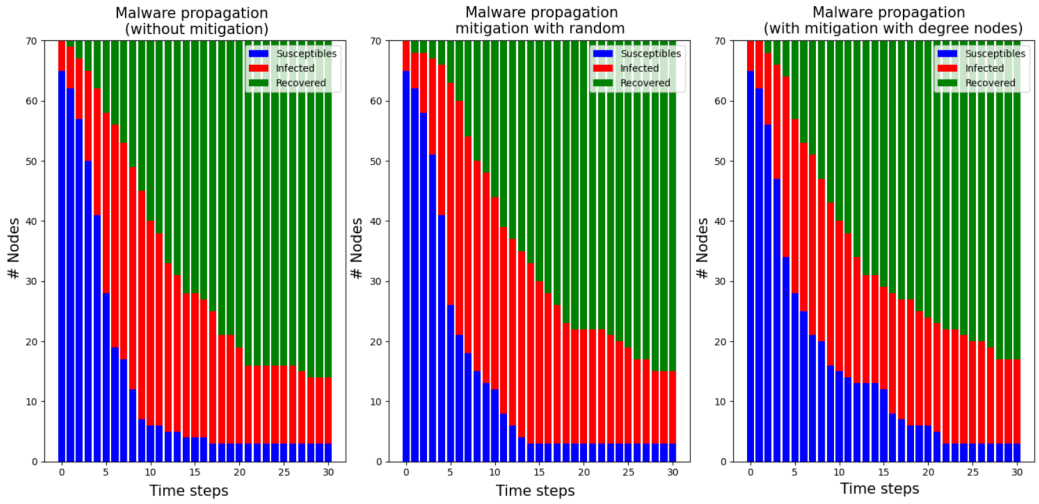

Figure 10. The evolution of malware in an IoT network over time is illustrated in this figure. The number of infected, susceptible and recovered nodes is represented by the stacked bars in red, blue, and green, respectively. The simulation includes 25% of the nodes with a new infection rate of $\frac{\beta}{3}$, which are patched at the third time step.

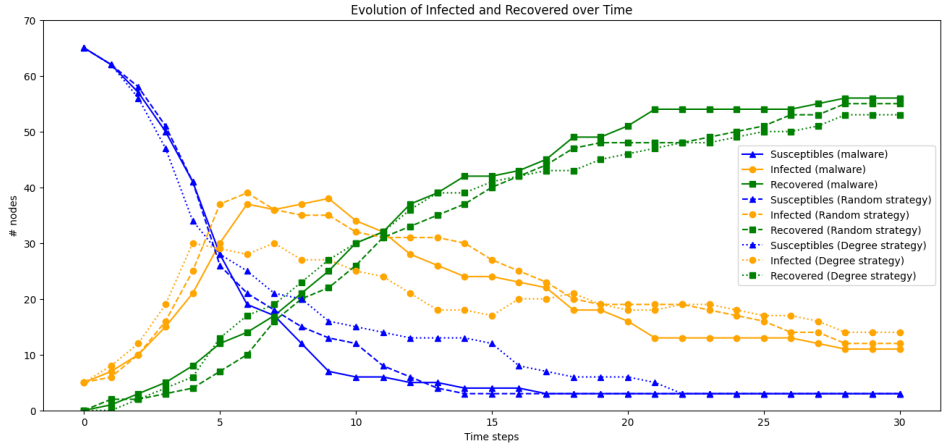

Figure 11. The figure shows the propagation of malware in an IoT network, with the yellow, green, and blue curves representing infected, recovered, and susceptible devices, respectively. The simulation includes 25% of the nodes with an infection rate of $\frac{\beta}{3}$, and it was patched in the third time step.

4.2.6. Simulation 2.3 with 25% Nodes, Patching $= \frac{\beta}{3}$, Early Patching Time Step $= 5$

In this simulation, the malware was detected early. However, due to the delay in security updates, it begins to install on selected nodes in time step 5. This results in suboptimal mitigation techniques, as both perform similarly to the malware spreading without mitigation techniques (see Figure 12). With regard to the number of infected and recovered individuals throughout the simulation, it is discernible from Figure 13 that it is comparable in all three instances.

Regarding the statistical analysis of the chosen metrics, no significant differences have been found (refer to Table 4), confirming the observations made in the propagation graphs.

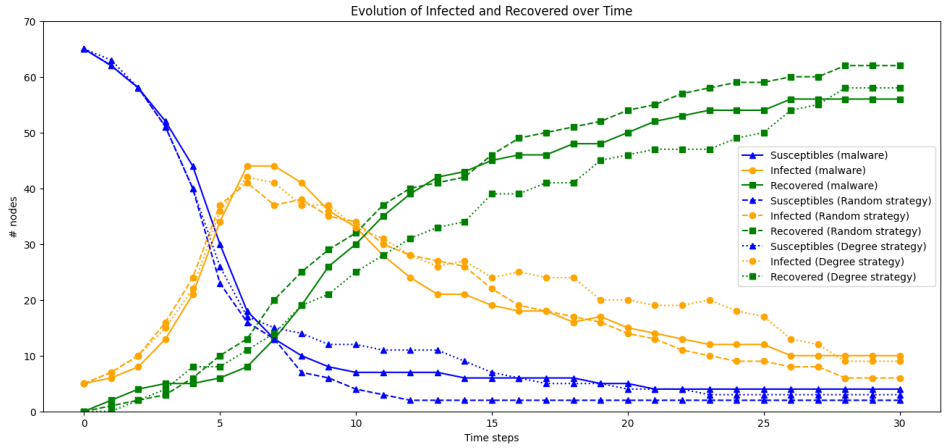

Figure 12. The figure illustrates the spread of malware in an IoT network, with the yellow, green, and blue lines representing the infected devices, the recovered devices, and the susceptible devices, respectively. The simulation included 25% of the nodes with an infection rate of $\frac{\beta}{3}$, and it was patched in the third time step.

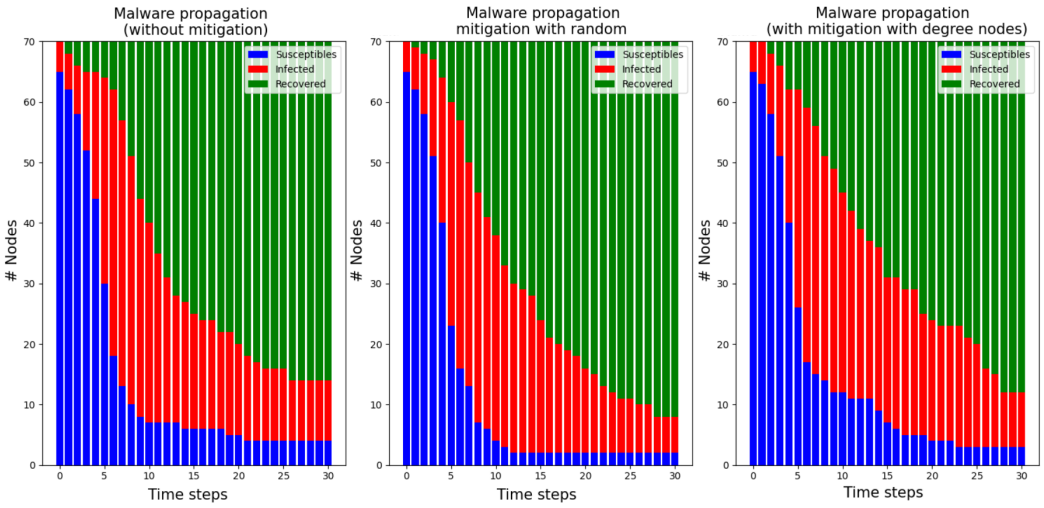

Figure 13. This diagram illustrates the progression of malware within an IoT network over time. The stacked bars in red, blue, and green represent the infected nodes, the susceptible nodes, and the recovered nodes, respectively. The simulation involves 25% of the nodes with a new infection rate of $\frac{\beta}{3}$, which are patched at the fifth time step.

5. Conclusions

In this paper, we examined the effects of the proposed mitigation measure on malware propagation in an IoT network. We propose updating the infection rate of individual nodes by simulating a security update with customizable effectiveness against malware. This mitigation measure imposes two criteria for node selection: one randomly and the other based on node degree in the IoT network. To examine the effectiveness of the proposed mitigation measures, we conducted two numerical simulations. The first simulation indicated that the mitigation strategy based on randomly selecting nodes necessitates updating at least 75% of nodes to attain a performance level comparable to that of the strategy based on node degree. The second simulation demonstrated that implementing a mitigation strategy that considers both criteria for node selection produces superior results in the event of early malware detection and minimal delay in the installation on IoT nodes. The metrics created to assess the effectiveness of the mitigation strategy using both node selection criteria demonstrate that opting for nodes based on node degree is efficient. Furthermore, it diminishes the economic cost since the mitigation strategy involving random node selection

demands the installation of updates on a large number of nodes (>75% of total nodes). For feature work, we will examine the same mitigation strategy, yet with complex criteria for selecting nodes that are founded on the features of the IoT network. Moreover, let us investigate other strategies for reducing harm, like segregating the malware, and hybrid methods that amalgamate several approaches for mitigating security threats. Finally, we will broaden the investigation to various specific models of malware dissemination.

Author Contributions: Conceptualization, R.C.-V., J.L.C.-R. and Á.M.d.R.; methodology, M.S. and R.C.-V.; software, M.S. and A.D.-L.; validation, R.C.-V., Á.M.d.R. and J.L.C.-R.; investigation, R.C.-V., M.S. and A.D.-L.; data curation, M.S. and A.D.-L.; writing—original draft preparation, R.C.-V., M.S., A.D.-L., Á.M.d.R. and J.L.C.-R.; writing—review and editing, R.C.-V., Á.M.d.R. and J.L.C.-R.; visualization, M.S. and A.D.-L.; supervision, R.C.-V., Á.M.d.R. and J.L.C.-R. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available on request from the corresponding author.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Stoyanova, M.; Nikoloudakis, Y.; Panagiotakis, S.; Pallis, E.; Markakis, E.K. A survey on the internet of things (IoT) forensics: Challenges, approaches, and open issues. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1191–1221. [CrossRef]
2. Xie, H.; Qin, Z. A lite distributed semantic communication system for Internet of Things. *IEEE J. Sel. Areas Commun.* **2020**, *39*, 142–153. [CrossRef]
3. Wang, X.; Zhang, X.; Wang, S.; Xiao, J.; Tao, X. Modeling, Critical Threshold, and Lowest-Cost Patching Strategy of Malware Propagation in Heterogeneous IoT Networks. *IEEE Trans. Inf. Forensics Secur.* **2023**, *18* , 3531–3545. [CrossRef]
4. Swessi, D.; Idoudi, H. A survey on internet-of-things security: Threats and emerging countermeasures. *Wirel. Pers. Commun.* **2022**, *124*, 1557–1592. [CrossRef]
5. Xu, D.; Wang, X.; Hao, Y.; Zhang, Z.; Hao, Q.; Zhou, Z. A More Accurate and Robust Binary Ring-LWE Decryption Scheme and Its Hardware Implementation for IoT Devices. *IEEE Trans. Very Large Scale Integr. VLSI Syst.* **2022**, *30*, 1007–1019. [CrossRef]
6. Zografopoulos, I.; Hatziargyriou, N.D.; Konstantinou, C. Distributed energy resources cybersecurity outlook: Vulnerabilities, attacks, impacts, and mitigations. *IEEE Syst. J.* **2023**, *17* , 6695–6709. [CrossRef]
7. Ahmad, S.; Jha, S.; Alam, A.; Alharbi, M.; Nazeer, J. Analysis of intrusion detection approaches for network traffic anomalies with comparative analysis on botnets (2008–2020). *Secur. Commun. Netw.* **2022**, *2022*, 9199703. [CrossRef]
8. Antonakakis, M.; April, T.; Bailey, M.; Bernhard, M.; Bursztein, E.; Cochran, J.; Durumeric, Z.; Halderman, J.A.; Invernizzi, L.; Kallitsis, M.; et al. Understanding the mirai botnet. In Proceedings of the 26th USENIX Security Symposium (USENIX Security 17), Vancouver, BC, Canada, 16–18 August 2017; pp. 1093–1110.
9. Güven, E.Y. Mirai Botnet Attack Detection in Low-Scale Network Traffic. *Intell. Autom. Soft Comput.* **2023**, *37*, 419–437. [CrossRef]
10. James, A.V.; Sabitha, S. Malware attacks: A survey on mitigation measures. In *Proceedings of the Second International Conference on Networks and Advances in Computational Technologies: NetACT 19*; Springer: Cham, Switzerland, 2021; pp. 1–11.
11. Pachhala, N.; Jothilakshmi, S.; Battula, B.P. A comprehensive survey on identification of malware types and malware classification using machine learning techniques. In Proceedings of the 2021 2nd International Conference on Smart Electronics and Communication (ICOSEC), Trichy, India, 7–9 October 2021; pp. 1207–1214.
12. Sprinkel, S.C. Global Internet Regulation: The Residual Effects of the ILoveYou Computer Virus and the Draft Convention on Cyber-Crime. *Suffolk Transnat'L Rev.* **2001**, *25*, 491.
13. Zhang, C.; Zhou, S.; Chain, B.M. Hybrid epidemics—A case study on computer worm conficker. *PloS ONE* **2015**, *10*, e0127478. [CrossRef]
14. Mohaisen, A.; Alrawi, O. Unveiling zeus: Automated classification of malware samples. In Proceedings of the 22nd International Conference on World Wide Web, Rio de Janeiro, Brazil, 13–17 May 2013; pp. 829–832.
15. Sood, A.K.; Enbody, R.J.; Bansal, R. Dissecting SpyEye–Understanding the design of third generation botnets. *Comput. Netw.* **2013**, *57*, 436–450. [CrossRef]
16. Thomas, K.; Bursztein, E.; Grier, C.; Ho, G.; Jagpal, N.; Kapravelos, A.; McCoy, D.; Nappa, A.; Paxson, V.; Pearce, P.; et al. Ad injection at scale: Assessing deceptive advertisement modifications. In Proceedings of the 2015 IEEE Symposium on Security and Privacy, San Jose, CA, USA, 17–21 May 2015; pp. 151–167.
17. Mohurle, S.; Patil, M. A brief study of wannacry threat: Ransomware attack 2017. *Int. J. Adv. Res. Comput. Sci.* **2017**, *8*, 1938–1940.
18. Mulligan, D.K.; Perzanowski, A.K. The magnificence of the disaster: Reconstructing the Sony BMG rootkit incident. *Berkeley Technol. Law J.* **2007**, *22*, 1157.

19. Mannix, K.; Gorey, A.; O'Shea, D.; Newe, T. Sensor Network Environments: A Review of the Attacks and Trust Management Models for Securing Them. *J. Sens. Actuator Netw.* **2022**, *11*, 43. [CrossRef]
20. Kermack, W.O.; McKendrick, A.G. A contribution to the mathematical theory of epidemics. *Proc. R. Soc. London. Ser. Contain. Pap. Math. Phys. Character* **1927**, *115*, 700–721.
21. Shi, X.; Zhang, T.; Zhou, D.; Zhou, X. Dynamical analysis and optimal control of a stochastic SIAR model for computer viruses. *Eur. Phys. J. Plus* **2023**, *138*, 1–27. [CrossRef]
22. She, B.; Gracy, S.; Sundaram, S.; Sandberg, H.; Johansson, K.H.; Paré, P.E. Epidemics spread over networks: Influence of infrastructure and opinions. In *Cyber–Physical–Human Systems: Fundamentals and Applications*; Wiley: Hoboken, NJ, USA, 2023; pp. 429–456.
23. Morris, D.H.; Rossine, F.W.; Plotkin, J.B.; Levin, S.A. Optimal, near-optimal, and robust epidemic control. *Commun. Phys.* **2021**, *4*, 78. [CrossRef]
24. Ojha, R.P.; Srivastava, P.K.; Sanyal, G.; Gupta, N. Improved model for the stability analysis of wireless sensor network against malware attacks. *Wirel. Pers. Commun.* **2021**, *116*, 2525–2548. [CrossRef]
25. Gracy, S.; Wang, Y.; Pare, P.E.; Uribe, C.A. Multi-Competitive Virus Spread over a Time-Varying Networked SIS Model with an Infrastructure Network. *arXiv* **2023**, arXiv:2303.08859.
26. Chen, J.; Huang, Y.; Zhang, R.; Zhu, Q. Optimal curing strategy for competing epidemics spreading over complex networks. *IEEE Trans. Signal Inf. Process. Over Netw.* **2021**, *7*, 294–308. [CrossRef]
27. Dinakarrao, S.M.P.; Guo, X.; Sayadi, H.; Nowzari, C.; Sasan, A.; Rafatirad, S.; Zhao, L.; Homayoun, H. Cognitive and scalable technique for securing IoT networks against malware epidemics. *IEEE Access* **2020**, *8*, 138508–138528. [CrossRef]
28. Khouzani, M.; Altman, E.; Sarkar, S. Optimal quarantining of wireless malware through power control. In Proceedings of the 2009 Information Theory and Applications Workshop, La Jolla, CA, USA, 8–13 February 2009; pp. 301–310.
29. Shen, S.; Li, H.; Han, R.; Vasilakos, A.V.; Wang, Y.; Cao, Q. Differential game-based strategies for preventing malware propagation in wireless sensor networks. *IEEE Trans. Inf. Forensics Secur.* **2014**, *9*, 1962–1973. [CrossRef]
30. Shen, S.; Ma, H.; Fan, E.; Hu, K.; Yu, S.; Liu, J.; Cao, Q. A non-cooperative non-zero-sum game-based dependability assessment of heterogeneous WSNs with malware diffusion. *J. Netw. Comput. Appl.* **2017**, *91*, 26–35. [CrossRef]
31. Alamo, T.; Reina, D.G.; Gata, P.M.; Preciado, V.M.; Giordano, G. Data-driven methods for present and future pandemics: Monitoring, modelling and managing. *Annu. Rev. Control.* **2021**, *52*, 448–464. [CrossRef] [PubMed]
32. Hong, Z.; Li, Y.; Gong, Y.; Chen, W. A data-driven spatially-specific vaccine allocation framework for COVID-19. *Ann. Oper. Res.* **2022**, 1–24. [CrossRef]
33. Castaneda, F.; Sezer, E.C.; Xu, J. Worm vs. worm: Preliminary study of an active counter-attack mechanism. In Proceedings of the 2004 ACM Workshop on Rapid Malcode, Washington, DC, USA, 20 October 2004; pp. 83–93.
34. Musaddiq, A.; Zikria, Y.B.; Zulqarnain; Kim, S.W. Routing protocol for Low-Power and Lossy Networks for heterogeneous traffic network. *EURASIP J. Wirel. Commun. Netw.* **2020**, *2020*, 1–23. [CrossRef]
35. Kuehne, H.; Jhuang, H.; Garrote, E.; Poggio, T.; Serre, T. HMDB: A large video database for human motion recognition. In Proceedings of the 2011 International Conference on Computer Vision, Barcelona, Spain, 6–13 November 2011; pp. 2556–2563.
36. Roy, A.; Singh, C.; Narahari, Y. Recent advances in modeling and control of epidemics using a mean field approach. *Sādhanā* **2023**, *48*, 207. [CrossRef]
37. Silva, D.H.; Anteneodo, C.; Ferreira, S.C. Epidemic outbreaks with adaptive prevention on complex networks. *Commun. Nonlinear Sci. Numer. Simul.* **2023**, *116*, 106877. [CrossRef]
38. Maitra, U.; Hota, A.R.; Srivastava, V. SIS Epidemic Propagation under Strategic Non-myopic Protection: A Dynamic Population Game Approach. *IEEE Control Syst. Lett.* **2023**, *7*, 1578–1583. [CrossRef]
39. Hota, A.R.; Maitra, U.; Elokda, E.; Bolognani, S. Learning to Mitigate Epidemic Risks: A Dynamic Population Game Approach. *Dyn. Games Appl.* **2023**, *13*, 1106–1129. [CrossRef]
40. Hassan, R.; Rafatirad, S.; Homayoun, H.; Dinakarrao, S.M.P. Performance-aware Malware Epidemic Confinement in Large-Scale IoT Networks. In Proceedings of the ICC 2021-IEEE International Conference on Communications, Montreal, QC, Canada, 14–23 June 2021; pp. 1–6.
41. Yang, L.X.; Li, P.; Yang, X.; Xiang, Y.; Jiang, F.; Zhou, W. Effective Quarantine and Recovery Scheme Against Advanced Persistent Threat. *IEEE Trans. Syst. Man Cybern. Syst.* **2021**, *51*, 5977–5991. [CrossRef]
42. Ren, J.; Zhang, C.; Hao, Q. A theoretical method to evaluate honeynet potency. *Future Gener. Comput. Syst.* **2021**, *116*, 76–85. [CrossRef]
43. Farooq, M.J.; Zhu, Q. Modeling, analysis, and mitigation of dynamic botnet formation in wireless IoT networks. *IEEE Trans. Inf. Forensics Secur.* **2019**, *14*, 2412–2426. [CrossRef]
44. Haghighi, M.S.; Wen, S.; Xiang, Y.; Quinn, B.; Zhou, W. On the race of worms and patches: Modeling the spread of information in wireless sensor networks. *IEEE Trans. Inf. Forensics Secur.* **2016**, *11*, 2854–2865. [CrossRef]
45. Aman, M.N.; Javaid, U.; Sikdar, B. IoT-Proctor: A Secure and Lightweight Device Patching Framework for Mitigating Malware Spread in IoT Networks. *IEEE Syst. J.* **2022**, *16*, 3468–3479. [CrossRef]
46. Marinov, T.T.; Marinova, R.S. Inverse problem for adaptive SIR model: Application to COVID-19 in Latin America. *Infect. Dis. Model.* **2022**, *7*, 134–148. [CrossRef] [PubMed]

47. Murray, J.D. Mathematical biology: I. an introduction 2002. In *Mathematical Biology: II. Spatial Models and Biomedical Applications*; Springer: New York, NY, USA, 2003.
48. Shafiq, M.; Gu, Z.; Cheikhrouhou, O.; Alhakami, W.; Hamam, H. The rise of "Internet of Things": Review and open research issues related to detection and prevention of IoT-based security attacks. *Wirel. Commun. Mob. Comput.* **2022**, *2022*, 1–12. [CrossRef]
49. del Rey, A.M.; Vara, R.C.; González, S.R. A computational propagation model for malware based on the SIR classic model. *Neurocomputing* **2022**, *484*, 161–171. [CrossRef]
50. Erdős, P.; Rényi, A. On the evolution of random graphs. *Publ. Math. Inst. Hung. Acad.* **1960**, *5*, 17–60.
51. Hagberg, A.; Swart, P.; Chult, D.S. *Exploring Network Structure, Dynamics, and Function Using NetworkX*; Technical Report; Los Alamos National Lab.(LANL): Los Alamos, NM, USA, 2008.
52. Schiassi, E.; De Florio, M.; D'Ambrosio, A.; Mortari, D.; Furfaro, R. Physics-informed neural networks and functional interpolation for data-driven parameters discovery of epidemiological compartmental models. *Mathematics* **2021**, *9*, 2069. [CrossRef]
53. Yuan, L.; Ni, Y.Q.; Deng, X.Y.; Hao, S. A-PINN: Auxiliary physics informed neural networks for forward and inverse problems of nonlinear integro-differential equations. *J. Comput. Phys.* **2022**, *462*, 111260. [CrossRef]
54. Gao, H.; Zahr, M.J.; Wang, J.X. Physics-informed graph neural Galerkin networks: A unified framework for solving PDE-governed forward and inverse problems. *Comput. Methods Appl. Mech. Eng.* **2022**, *390*, 114502. [CrossRef]
55. Grimm, V.; Heinlein, A.; Klawonn, A.; Lanser, M.; Weber, J. Estimating the time-dependent contact rate of SIR and SEIR models in mathematical epidemiology using physics-informed neural networks. *Electron. Trans. Numer. Anal.* **2022**, *56*, 1–27. [CrossRef]
56. Raissi, M.; Perdikaris, P.; Karniadakis, G.E. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *J. Comput. Phys.* **2019**, *378*, 686–707. [CrossRef]
57. Lu, L.; Meng, X.; Mao, Z.; Karniadakis, G.E. DeepXDE: A deep learning library for solving differential equations. *SIAM Rev.* **2021**, *63*, 208–228. [CrossRef]