

Article

Machine Learning White-Hat Worm Launcher for Tactical Response by Zoning in Botnet Defense System [†]

Xiangnan Pan and Shingo Yamaguchi * 

Graduate School of Sciences and Technology for Innovation, Yamaguchi University, Ube 755-8611, Japan; pan25010@gmail.com

* Correspondence: shingo@yamaguchi-u.ac.jp

[†] This paper is an extended version of our paper published in Pan, X. and Yamaguchi, S. A DBSCAN-based White-Hat Worm Launcher for Botnet Defense System. In Proceedings of the IEEE LifeTech 2022, Osaka, Japan, 7–9 March 2022; pp. 54–57.

Abstract: Malicious botnets such as Mirai are a major threat to IoT networks regarding cyber security. The Botnet Defense System (BDS) is a network security system based on the concept of “fight fire with fire”, and it uses white-hat botnets to fight against malicious botnets. However, the existing white-hat Worm Launcher of the BDS decides the number of white-hat worms, but it does not consider the white-hat worms’ placement. This paper proposes a novel machine learning (ML)-based white-hat Worm Launcher for tactical response by zoning in the BDS. The concept of zoning is introduced to grasp the malicious botnet spread with bias over the IoT network. This enables the Launcher to divide the network into zones and make tactical responses for each zone. Three tactics for tactical responses for each zone are also proposed. Then, the BDS with the Launcher is modeled by using agent-oriented Petri nets, and the effect of the proposed Launcher is evaluated. The result shows that the proposed Launcher can reduce the number of infected IoT devices by about 30%.

Keywords: botnet; BDS; machine learning (ML); zoning; IoT; white-hat; Petri net



Citation: Pan, X.; Yamaguchi, S. Machine Learning White-Hat Worm Launcher for Tactical Response by Zoning in Botnet Defense System. *Sensors* **2022**, *22*, 4666. <https://doi.org/10.3390/s22134666>

Academic Editors: Himanshu Thapliyal and Akhilesh Tyagi

Received: 22 May 2022

Accepted: 19 June 2022

Published: 21 June 2022

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2022 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

The Internet of Things (IoT) has become an exciting technology in the world, and it has grown into the largest platform of millions of IoT devices interacting with each other to improve our daily lives. However, such a popular IoT can also be used as a potential tool for uploading malicious entities. These entities use IoT vulnerabilities caused by various constraints such as limited resources and weak security to access legitimate devices and can further take various forms of attack [1]. For example, distributed denial of service (DDoS) in IoT networks is a very popular malicious attack, which disrupts the normal traffic of the target server by launching multiple consecutive requests from distributed IoT devices, eventually causing the server to crash [2–4].

Mirai, a malware capable of launching DDoS attacks, is also accountable for the largest record of attack traffic to date, targeting a French-managed service provider and up to 15 million IoT devices at a maximum speed of 1 Tbps [5]. The most remarkable thing is that the source code of Mirai was published by its author [6]. This may give chances for unethical behavior on the Internet. So far, the devices infected by Mirai have already reached 164 countries; the speed of spread is amazing. It is estimated that the number of deployed IoT devices will be about 20–30 billion by 2030 [7].

Faced with such a huge number of IoT devices, one method is to use a kind of white-hat worm, such as Hajime, to fight Mirai [8]. The white-hat worm has two attributes, the lifespan and the secondary infection possibility [9]. These two attributes ensure that the white-hat worms can destroy themselves after the Mirai bots are eliminated by them. Yamaguchi [10] proposed a promising network security system called the Botnet Defense

System (BDS). It can use white-hat botnets to eliminate malicious botnets like Mirai. Yamaguchi also proposed three basic strategies for white-hat worms according to the network situation and confirmed the effect of the BDS through an agent-oriented Petri net model (PN² model) [11].

The BDS only considers the number of white-hat worms; there is no discussion on the position of white-hat worms. Pan et al. [12] proposed a machine-learning (ML)-based white-hat Worm Launcher for the BDS. In this ML-based Launcher, the given network is rigidly divided into several subnetworks with the divide-and-conquer algorithm (D&C algorithm) to adapt the machine learning models, and then, the white-hat worms' positions are predicted by the ML models. The ML-based Launcher also has a good adaptability because of the D&C algorithm; even if the given network is very large, we can also successfully input the modified network into the ML models through the preprocessing step of "padding" or "multiple divide-and-conquer". However, the Mirai botnet is not always spread uniformly over the network. Some parts have a high density of Mirai bots, while others have a low one. The D&C algorithm always divides the whole network into the same subnetworks and processes them by ML models, which will increase the time and resources. In general, there is some bias in the distribution of the Mirai bots, so the BDS should deal with the Mirai botnet reasonably according to the distribution, and it is necessary to optimize our ML white-hat worm Launcher.

In this paper, we propose a novel machine learning white-hat Worm Launcher for tactical response by zoning in the BDS. We introduce the concept of zoning to grasp the malicious botnet spread with bias over the IoT network. This enables the BDS to divide the network into areas called zones and make tactical responses for each zone. We also propose tactics for response to each zone. Then, we model the BDS with the Launcher by using agent-oriented Petri nets and evaluate the effect of the proposed Launcher. We use Mirai as a representative of malicious botnets. The proposed method is available to the other malicious botnets.

Our major contributions are as follows:

- (1) The concept of zoning is introduced to grasp malicious botnets' spread with bias over IoT networks.
- (2) Tactical response by zone is proposed to build and operate white-hat botnets efficiently and effectively against the malicious botnets.
- (3) The effect of our white-hat Worm Launcher for BDS is confirmed through simulation evaluation with agent-oriented Petri nets. The result shows that the proposed Launcher can reduce the number of infected IoT devices by about 30%.

The rest of this paper is organized as follows. Section 2 introduces the previous work and the BDS. Section 3 describes the zoning and the tactics to respond by zoning. Section 4 describes the simulation evaluation. Section 5 summarizes the results obtained in this paper and gives future work.

2. Related Work

2.1. Previous Work

Recent related studies on botnets can be roughly divided into two categories according to their purposes: detection and mitigation.

With the popularity of artificial intelligence (AI), the detection mechanisms of malicious network attacks are keen to use AI technology [13]. Nagisetty and Gupta [14] used four different deep learning models to construct a framework in the IoT backbone network to identify malicious network traffic. Soe et al. [15] proposed a high-performance attack detection system with three different machine learning (ML) algorithm sequential classifiers. Gupta et al. [16] proposed a deep-reinforcement-learning (DRL)-based detection framework with a dataset that contains IoT traces. Qu et al. [17] proposed an evaluation method based on percolation theory to quantify the security risk in a power CPS network. Qu et al. [18] also proposed a hybrid detection method with improved AKF and GRU-CNN to detect false data injection attacks. Wang et al. [19] proposed a new method to automatically mine

attack patterns from measurement data and information alarm logs. Vishwakarma et al. [20] proposed a honeypot-based framework with machine learning techniques, which shows a high performance on zero-day DDoS attacks. In addition, the detection model usually needs to rely on reliable datasets, and some researchers also pay attention to the generation of network datasets or simulators. Guerra-Manzanares et al. [21] proposed an IoT dataset for ML-based detection frameworks, which includes actual malicious network traffic in a medium-sized IoT. Adan et al. [22] designed an Internet of Things attack simulator, which shows how Mirai works in DDoS attacks. Other botnet detection frameworks also include a blockchain-based mechanism [23–25] and an SDN-based mechanism [26].

Botnet mitigation measures are also very important. This aims to mitigate the threats from the DDoS. Gopal et al. [27] thought of a white-list-based solution to prevent the spread of Mirai and confirmed the blocking function against botnets through experiments. Manso et al. [28] proposed an intrusion detection system through a software layer, and it can launch a software-defined network controller to control the traffic when the system detects an attack. This can reduce the traffic and makes it possible to maintain normal traffic when an attack happens. Ceron et al. [29] proposed a method to handle the network traffic caused by IoT malware. The method can characterize the malware behaviors and modify the traffic through the network layer to limit malicious activities.

We briefly summarize the attributes and limitations of the above detection and mitigation methods in Tables 1 and 2. The previous studies only focused on the detection methods or mitigation measures; they are not able to exterminate existing botnets.

Table 1. Overview of the detection methods.

Authors	Year	Attributes	Limitations	ML Algorithm ¹	Eval. Measure
Qu et al. [17]	2018	Quantify the security risk in CPS network	No corresponding strategy for risk outbreak	N/A	Failure ratio
Nagisetty et al. [14]	2019	Monitor the network traffic by DL	Over-reliance on IoT traffic dataset	MLP, CNN	RMSE, F1-score
Vishwakarma et al. [20]	2019	Design a honeypot-based ML framework	Adaptability limitation to real-time scenarios	SVM, NB	N/A
Adan et al. [22]	2019	Design an IoT attack simulator	Limitation to public IoT networks	N/A	N/A
Soe et al. [15]	2020	Made sequential classifiers based on ML	Specific botnet can be classified	ANN, DT, NB	ACC
Guerra-Manzanares et al. [21]	2020	Generate traffic dataset of botnet	Low adaptability to real-time network	N/A	N/A
Wang et al. [19]	2020	Mine attack patterns from alarm logs	Lack of adaptability to local attack sequences	NB	ACC
Qu et al. [18]	2022	Detect attacks with AKF and GRU-CNN	Limitations for data loss and external interference	AFK, GRU-CNN	FPR, ACC
May raju et al. [16]	2022	Detect intrusion based on DRL	Compare with only two common ML models	DRL	ACC

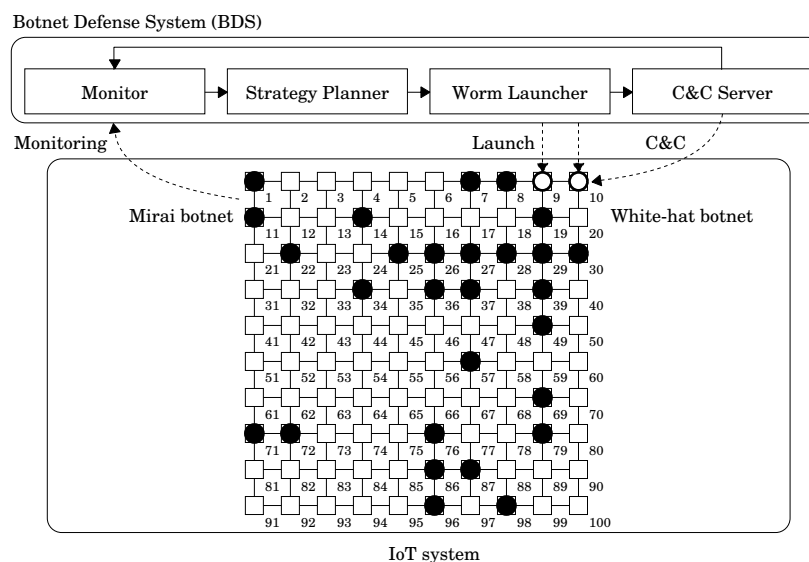
¹ MLP: multilayer perceptron, CNN: convolutional neural network, SVM: support vector machine, NB: naive Bayes, ANN: artificial neural network, DT: decision tree, AKF: adaptive Kalman filter, GRU: gated recurrent unit, DRL: deep reinforcement learning.

Table 2. Overview of the mitigation methods.

Authors	Year	Attributes	Limitations	Eval. Measure
Gopal et al. [27]	2018	Establish a security wall based on white listing	Lack of protection by other mechanisms	Router Model
Manso et al. [28]	2019	Construct a software-defined security system	Cause congestion of DNS controller	Quality of service
Ceron et al. [29]	2019	Modify the malicious traffic at network layer	Lack of dynamic malware analysis	Packet number

2.2. Botnet Defense System

The BDS is a network security system based on the concept of “fight fire with fire”. It uses white-hat botnets to fight against malicious botnets like Mirai. Figure 1 illustrates the configuration of the BDS and an IoT system defended by the BDS. The upper part represents the structure of the BDS. It consists of four parts: Monitor, Strategy Planner, Worm Launcher, and C&C Server. The lower part represents the IoT system. The inner graph represents the network of the system. Each vertex represents a network node, and each node has a single device. Each edge represents a connection between network nodes. A black circle (●) and a white circle (○) respectively represent a malicious bot and a white-hat bot. When the Monitor detects a malicious botnet, the Worm Launcher will send the white-hat worm according to the strategy specified by the Strategy Planner. Yamaguchi also put forward three basic strategies. These strategies can adjust the number of white-hat worms according to not only the ability of the worms, but also the structure of the network. However, there is no discussion on the placement of white-hat worms; it should be reasonably considered and configured.

**Figure 1.** The overview of the Botnet Defense System.

Pan et al. [12] proposed a machine learning white-hat Worm Launcher for the BDS. It basically has two phases: decision and placement. The first phase uses a feedforward neural network (FNN) to decide an appropriate placement tactic. The second phase uses a multitask learning model to predict white-hat worms’ appropriate position based on the predicted tactic. Furthermore, this Launcher takes a divide-and-conquer approach so that it can be adapted to large-scale IoT networks. The procedure of the Launcher is given as Procedure 1. The Launcher divides a given network into subnetworks. For each subnetwork, it predicts a tactic and, then, the white-hat worms’ placement based on the tactic.

Procedure 1 WORM-LAUNCHER(N, D_{Mirai}, M_1, M_2).

Input: Mirai bots' distribution D_{Mirai} in network N , machine learning model M_1 to decide tactics, machine learning model M_2 to place white-hat worms

Output: White-hat worms' placement D_W for D_{Mirai}

▷ *Step 1: Divide Stage*

1: Divide N into n subnetworks $N_1, N_2, \dots, N_n$ by network scale

▷ *Step 2: Conquer Stage*

2: **for** each subnetwork N_i in $N_1, N_2, \dots, N_n$ **do**

3: Use M_1 to decide a tactic T_i for N_i

4: Use M_2 to make white-hat worms' placement D_{W_i} on N_i based on T_i

5: **end for**

▷ *Step 3: Combining Stage*

6: Combine $D_{W_1}, D_{W_2}, \dots, D_{W_n}$ with the whole placement D_W

7: Output D_W and stop.

We illustrate the procedure with an example shown in Figure 2. Twenty devices are infected by Mirai in a 10×10 grid network. As a preparation, we trained two machine learning models, i.e., the FNN model M_1 and the multitask learning model M_2 , for a small network. The small network actually corresponds to the subnetwork divided in the procedure:

- Step 1 divides the network into subnetworks by the network scale. Assume $n = 4$ in this example. That is, the 10×10 network is divided into four 25×25 subnetworks.
- Step 2 predicts a tactic with M_1 and, then, the white-hat worms' placement based on the tactic with M_2 for each subnetwork.
- Step 3 combines the predictions of all subnetworks into the final white-hat worm's placement.

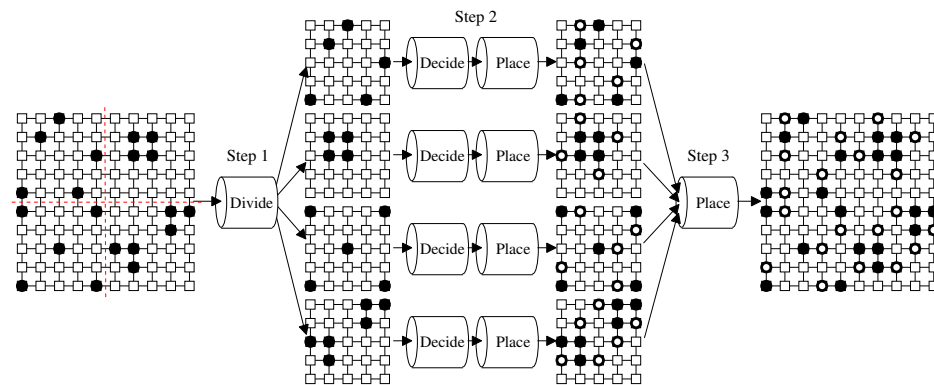


Figure 2. The operation of Procedure WORM-LAUNCHER. After dividing a given 10×10 network into four 25×25 subnetworks, for each subnetwork, the Launcher predicts the placement of white-hat worms by machine learning models.

3. White-Hat Worm Launcher for Tactical Response by Zoning

3.1. Zoning

The Mirai botnet is not always spread uniformly over the network. There may be some bias in the distribution of the Mirai bots. The BDS should deal with the Mirai botnet based on its bots' distribution. We define a density of bots in some area of the network as follows.

Definition 1. (Bots density) Given a distribution of bots on a network, for a specific area of the network, the density δ of the bots is given as follows:

$$\delta \stackrel{\text{def}}{=} \frac{\text{Number of bots}}{\text{Number of devices in the area}} [\%].$$

We introduce the concept of zoning to the BDS. The BDS divides a given IoT network into some zones and plans a tactic for each zone. Note that the zoning depends not on the network scale, but on the Mirai bots' density. We can expect the BDS to predict a tactic appropriate for the bots' density in each zone and to build a white-hat botnet effectively and efficiently.

For zoning, we used a density-based clustering algorithm called density-based spatial clustering of applications with noise (DBSCAN) [30]. Density-based clustering extracts areas that have a higher density than the rest as clusters. DBSCAN takes two parameters: ϵ and minPts . ϵ is the radius from a point. minPts is the minimum number of points needed to form a dense region. Given a set of points, DBSCAN classifies each point into three categories: Core, Border, and Noise. Core points are points with at least minPts neighbors within ϵ . Border points are points with no more than minPts neighbors within ϵ , but with Core points within ϵ . Noise points are points with no neighbors within ϵ . Finally, DBSCAN creates clusters from a collection of Core points and Border points. We formalize DBSCAN as follows.

Definition 2. (DBSCAN) For a set P of points, a radius ϵ from a point, and the minimum number minPts of points needed to form a dense region, DBSCAN is given as a mapping:

$$\text{DBSCAN} : (P, \epsilon, \text{minPts}) \mapsto C_1, C_2, \dots, C_m, \quad (1)$$

where $C_1, C_2, \dots, C_m$ are clusters of the points.

The reason for using DBSCAN is that it can find non-linearly separable clusters unlike k-means or Gaussian mixture EM clustering. It can find even a cluster that is surrounded by others. This is very important for zoning. Another reason is it is easy to control since there are only two parameters.

Combining clusters with a similar density, we define an obtained area as a zone.

Definition 3. (Zone) Let C be any cluster obtained by DBSCAN and $\delta(C)$ be the bot density of C . For some range of bots density, a zone is the union of cluster C such that $\delta(C)$ is in the range.

We used the following three zones in this paper:

- Danger Zone Z_{Danger} : contains a cluster C such that δ is $\underline{\delta_{\text{Danger}}} \leq \delta(C) \leq \overline{\delta_{\text{Danger}}} (=100\%)$. Almost all devices in this zone are infected by Mirai.
- Safe Zone Z_{Safe} : contains a cluster C such that $\underline{\delta_{\text{Safe}}} (=0\%) \leq \delta(C) \leq \overline{\delta_{\text{Safe}}}$. Almost no devices in this zone are infected.
- Unknown Zone Z_{Unknown} : contains a cluster C such that δ is $\underline{\delta_{\text{Safe}}} < \delta(C) < \overline{\delta_{\text{Danger}}}$. The devices in this zone are at risk of being infected.

We propose a procedure for zoning, which divides the given network into the three zones. The procedure is given as Procedure 2. Simply put, we ran DBSCAN twice for zoning. The first is for finding the Danger Zone, where a point represents the position of an infected device. The second is for finding the Safe Zone, where we regard a point to represent the position of a normal (uninfected) device. The remaining areas are incorporated into the Unknown Zone.

Procedure 2 DBSCAN-BASED-ZONING(N, D_{Mirai}, D_{normal}).

Input: Mirai bots' distribution D_{Mirai} in network N , normal devices' distribution D_{normal} on N .

Output: Danger Zone Z_{Danger} , Safe Zone Z_{Safe} , Unknown Zone $Z_{Unknown}$.

▷ *Step 1: Finding Danger Zone*

- 1: $P_{Mirai} \leftarrow \{\text{the point of each Mirai bot in } D_{Mirai} \text{ in the space of } N\}$.
- 2: $C_1^{Mirai}, C_2^{Mirai}, \dots, C_m^{Mirai} \leftarrow DBSCAN(P_{Mirai}, \epsilon_{Mirai}, minPts_{Mirai})$.
- 3: $Z_{Danger} \leftarrow \emptyset$.
- 4: **for** each cluster C in $C_1^{Mirai}, C_2^{Mirai}, \dots, C_m^{Mirai}$ **do**
- 5: **if** $\delta(C) \geq \overline{\delta_{Danger}}$ **then**
- 6: $Z_{Danger} \leftarrow Z_{Danger} \cup C$.
- 7: **end if**
- 8: **end for**

▷ *Step 2: Finding Safe Zone*

- 9: $P_{normal} \leftarrow \{\text{the point of each normal device in } D_{normal} \text{ in the space of } N\}$.
- 10: $C_1^{normal}, C_2^{normal}, \dots, C_n^{normal} \leftarrow DBSCAN(P_{normal}, \epsilon_{normal}, minPts_{normal})$.
- 11: $Z_{Safe} \leftarrow \emptyset$.
- 12: **for** each cluster C in $C_1^{normal}, C_2^{normal}, \dots, C_n^{normal}$ **do**
- 13: **if** $\delta(C) \leq \overline{\delta_{Safe}}$ **then**
- 14: $Z_{Safe} \leftarrow Z_{Safe} \cup C$.
- 15: **end if**
- 16: **end for**

▷ *Step 3: Finding Unknown Zone*

- 17: $Z_{Unknown} \leftarrow (D_{Mirai} \cup D_{normal}) \setminus (Z_{Danger} \cup Z_{Safe})$.

▷ *Step 4*

- 18: **Output** $Z_{Danger}, Z_{Safe}, Z_{Unknown}$ and **stop**.

The DBSCAN algorithm is known to have difficulty determining the parameter values for a dataset with large density differences. To solve this problem, we decided to run the DBSCAN algorithm many times, depending on the density. In fact, Procedure DBSCAN-BASED-ZONING runs the DBSCAN algorithm twice.

We illustrate the proposed zoning procedure with an example shown in Figure 3. Figure 3a represents the Mirai bots' distribution in a 10×10 grid network. In this example, we set $\epsilon_{Mirai} = 1$ and $minPts_{Mirai} = 3$. This means that if Mirai has infected a device and it is surrounded by more than two neighbors, we regard them to be in danger because they form a cluster. Meanwhile, we set $\epsilon_{normal} = 1$ and $minPts_{normal} = 5$. This means that if an uninfected device is surrounded by four uninfected devices, we regard them to be safe because there is no risk of infection. We can determine these values according to the security policy and network configuration. For example, if $\epsilon_{Mirai} = 1$ and $minPts_{Mirai} = 2$, we identify Danger Zones with greater sensitivity. This may launch more white-hat bots than necessary. On the other hand, if $\epsilon_{Mirai} = 1$ and $minPts_{Mirai} = 5$, we identify Danger Zones with lower sensitivity. This may build weak a white-hat botnet. There is a trade-off between the detection and the disinfection ability. These parameters provide adaptability to balance between them:

- Step 1 finds the Danger Zone. As a result of DBSCAN with $\epsilon_{Mirai} = 1$ and $minPts_{Mirai} = 3$, we obtain two clusters C_1^{Mirai} (red area in the upper right of Figure 3b) and C_2^{Mirai} (red area in the lower right of Figure 3b). Since $\delta(C_1^{Mirai}) = \delta(C_2^{Mirai}) = 100\% \geq \overline{\delta_{Danger}}$, these clusters comprise the Danger Zone, i.e., $Z_{Danger} = C_1^{Mirai} \cup C_2^{Mirai}$.
- Step 2 finds the Safe Zone. As a result of DBSCAN with $\epsilon_{normal} = 1$ and $minPts_{normal} = 5$, we obtain one cluster C_1^{normal} (blue area in the left of Figure 3b). For $\overline{\delta_{Safe}} = 0$, since $\delta(C_1^{normal}) = 0\% \leq \overline{\delta_{Safe}}$, the cluster becomes the Safe Zone, i.e., $Z_{Safe} = C_1^{normal}$.
- Step 3 finds the Unknown Zone. The remaining areas comprise $Z_{Unknown}$ (yellow areas of Figure 3b).

- Step 4 outputs Z_{Danger} , Z_{Safe} , $Z_{Unknown}$ as the Danger, Safe, and Unknown Zones.

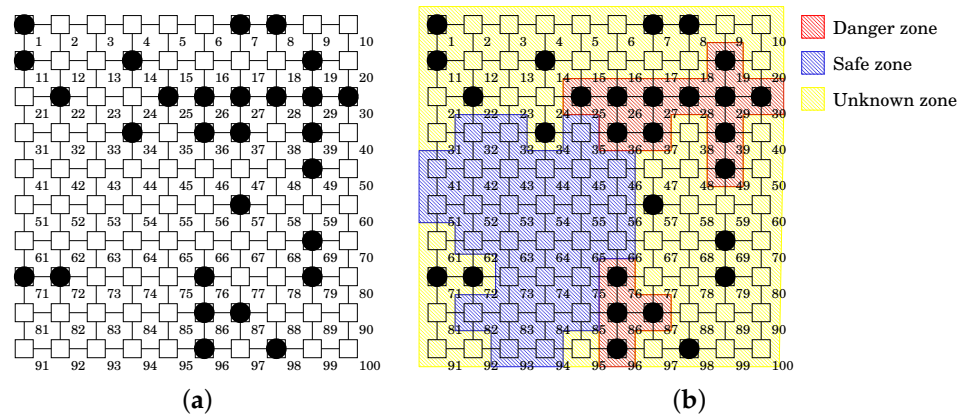


Figure 3. Application example of Procedure DBSCAN-BASED-ZONING. (a) Mirai bots' distribution. (b) Result of zoning.

3.2. Tactics by Zone

We propose tactics for sending white-hat worms by zone to build the white-hat botnet efficiently and effectively. The tactic for each zone is as follows.

3.2.1. Surrounding Tactic $T_{Surrounding}$ for Danger Zone

In the Danger Zone, Mirai bots are clumped. The BDS should confine those Mirai bots within the zone. The surrounding tactic [31] places white-hat worms around the zone and prevents the Mirai bots from spreading. See Figure 4.

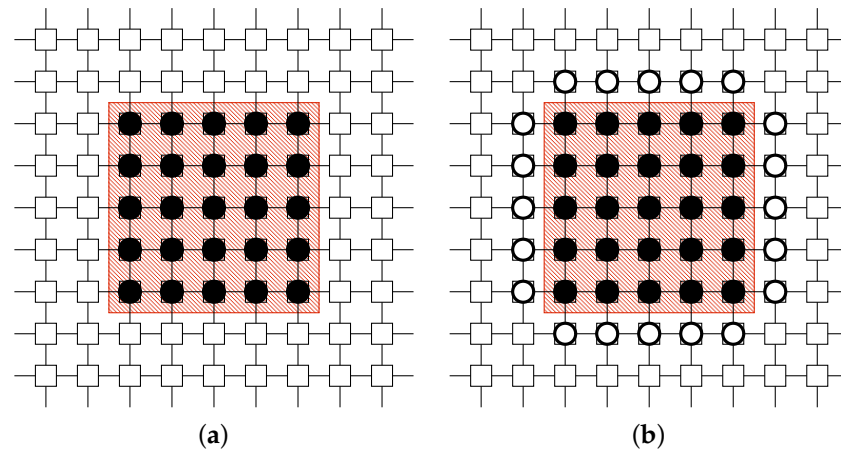


Figure 4. Application example of surrounding tactic for Danger Zone. (a) Before. The red area represents a Danger Zone. (b) After. The white-hat worms are placed to surround the zone.

3.2.2. Protecting Tactic $T_{Protecting}$ for Safe Zone

In the Safe Zone, there are few Mirai bots. Therefore, the BDS should prevent the Mirai botnet from entering this area. The protecting tactic [32] places white-hat worms on the border of the area to protect them like a firewall and prevent the bots from entering. See Figure 5.

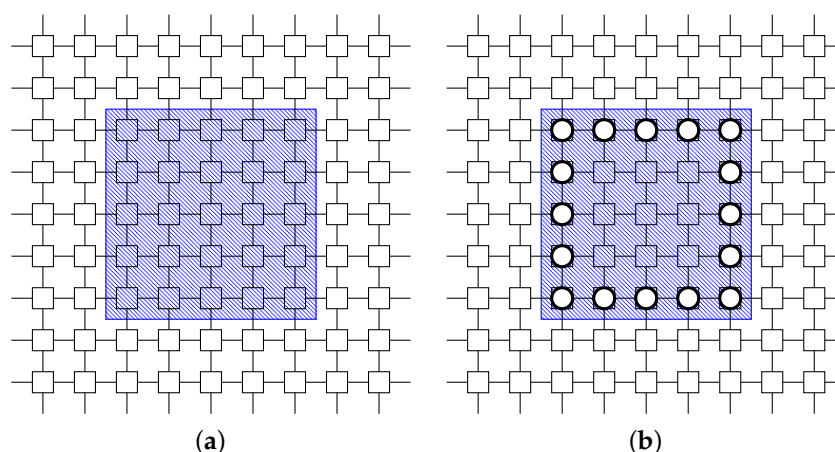


Figure 5. Application example of protecting tactic for Safe Zone. (a) Before. The blue area represents a Safe Zone. (b) After. The white-hat worms are placed to frame the zone.

3.2.3. Machine Learning Tactic T_{ML} for Unknown Zone

This tactic places white-hat worms according to the result of the machine learning models. See Figure 6. We used a multitask learning model to implement this tactic in the same way as the second phase of [12].

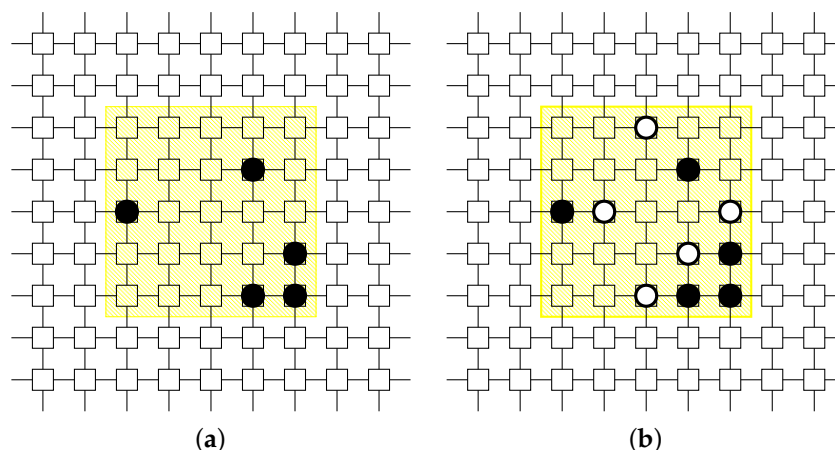


Figure 6. Application example of machine learning tactic for Unknown Zone. (a) Before. The yellow area represents an Unknown Zone. (b) After. The white-hat worms are placed according to the prediction of the given machine learning model.

3.3. White-Hat Worm Launcher by Zoning

Combining the zoning procedure and the tactics for sending white-hat worms by zone, we propose a new white-hat Worm Launcher that can make tactical responses for the divided zones. The procedure of the Launcher is given as Procedure 3. The first step uses Procedure DBSCAN-BASED-ZONING to divide a given network into three zones, i.e., the Danger, Safe, and Unknown Zones. The second step makes the placement of white-hat worms according to the tactic for each zone, i.e., the surrounding tactic for the Danger Zone, the protecting tactic for the Safe Zone, and the machine learning tactic for the Unknown Zone. The third step combines those results for the whole placement.

We illustrate the proposed procedure with an example shown in Figure 7. The network and Mirai bots' distribution are the same as the example of Figure 3:

- Step 1 divides the network into three zones by using Procedure DBSCAN-BASED-ZONING. As stated before, we obtain the result of zoning shown in Figure 3b.

- Step 2 places white-hat worms in each zone. Each zone takes a different tactic, i.e., the surrounding tactic for the Danger Zone, the protecting tactic for the Safe Zone, and the machine learning tactic for the Unknown Zone.
- Step 3 combines those results for the whole placement.

Procedure 3 ZONING-BASED-WORM-LAUNCHER($N, D_{Mirai}, D_{normal}, M_2$).

Input: Mirai bots' distribution D_{Mirai} in network N , normal devices' distribution D_{normal} on N , machine learning model to place white-hat worms M_2 .

Output: White-hat worms' placement D_W for D_{Mirai} .

▷ Step 1: Zoning Stage

1: $Z_{Danger}, Z_{Safe}, Z_{Unknown} \leftarrow \text{DBSCAN-BASED-ZONING}(N, D_{Mirai}, D_{normal})$.

▷ Step 2: Placing Stage

2: **for** each zone Z in $Z_{Danger}, Z_{Safe}, Z_{Unknown}$ **do**

3: Place white-hat worms as follows:

4: **if** Z is Z_{Danger} **then**

5: Use the surrounding tactic and store the placement result in $D_{W_{Danger}}$.

6: **end if**

7: **if** Z is Z_{Safe} **then**

8: Use the protecting tactic and store the placement result in $D_{W_{Safe}}$.

9: **end if**

10: **if** Z is $Z_{Unknown}$ **then**

11: Use the machine Learning tactic and store the placement result predicted by M_2 in $D_{W_{Unknown}}$.

12: **end if**

13: **end for**

▷ Step 3: Combining Stage

14: Combine $D_{W_{Danger}}, D_{W_{Safe}}, D_{W_{Unknown}}$ with the whole placement D_W .

15: Output D_W and stop.

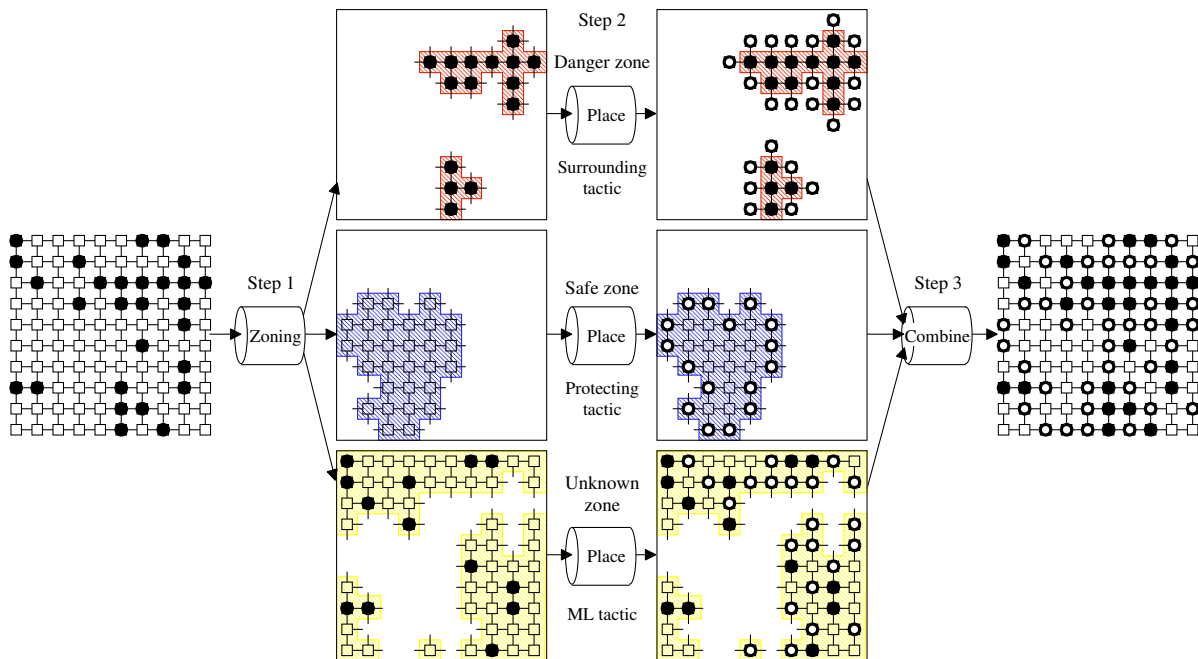


Figure 7. The operation of Procedure ZONING-BASED-WORM-LAUNCHER. After dividing a given 10×10 network into three zones based on the Mirai bots' density, the Launcher places white-hat worms according to the tactic for each zone.

Procedure ZONING-BASED-WORM-LAUNCHER can run in a few seconds on a PC with an Intel Core i7 2.6 GHz and 16 GB RAM. Zoning in Step 1 takes about one second. Step 2 decides the placement of white-hat bots for each zone. Zoning omits predictions by the learning model in some zones and allows for a significant reduction in computation time. As a result, it can run in one second. Combining in Step 3 hardly takes any time.

4. Simulation Evaluation

4.1. Experiment

We modeled the BDS with the proposed zoning-based Worm Launcher by using an agent-oriented Petri net PN^2 and, then, carried out experiments to evaluate the effect of the Launcher through the simulation of the PN^2 model. For the detail of the modeling by PN^2 , refer to [9].

We configured a 10×10 grid network N and considered four kinds of initial distributions of Mirai. The first three distributions and the result of zoning are illustrated in Figure 8:

- Distribution 1: Has a low density of Mirai bots in the whole network, i.e., $\delta(N) = 31\%$. This is illustrated in Figure 8a. The infected network is divided into 25% Danger Zone, 50% Safe Zone, and 25% Unknown Zone.
- Distribution 2: Has a medium density of Mirai bots in the whole network, i.e., $\delta(N) = 37\%$. This is illustrated in Figure 8b. The infected network is divided into 25% Danger Zone, 25% Safe Zone, and 50% Unknown Zone.
- Distribution 3: Has a high density of Mirai bots in the whole network, i.e., $\delta(N) = 56\%$. This is illustrated in Figure 8c. The infected network is divided into 50% Danger Zone, 25% Safe Zone, and 25% Unknown Zone.
- Distribution 4: Set to be random. It has almost the same density as Distribution 1, i.e., $\delta(N) = 30\%$; unlike Distribution 1, all the Mirai bots are placed randomly.

We used the tool PN^2 Simulator [33] for the simulation. We took Mirai's infection rate R_{Mirai} as a quantitative index. R_{Mirai} is given by $R_{Mirai} = \#_{Mirai} / \#_{device}$, where $\#_{Mirai}$ is the number of devices infected by Mirai and $\#_{device}$ is the number of all devices in N . The other simulation parameters are given as follows:

- Lifespan $\ell = 2$;
- Delay time $\delta = 7$ until rebooting;
- Secondary infection probability $\rho = 50\%$.

4.2. Simulation Result

We compared the proposed Launcher with the Launcher of [12]. The notable difference between them having or not having zoning, i.e., the proposed Launcher has zoning, while the Launcher of [12] has no zoning. The simulation results can be seen in Table 3. Each value represents the mean of R_{Mirai} at 1, 10, 100, 1000, and 10,000 steps for 1000 trials. In every distribution and step, the infection rate of the Launcher with zoning had to a lower value than the Launcher without zoning. Figure 9 charts these results to make it easier to compare these Launchers. From the graphs, we found that in the early stage, the Launcher with zoning removed significantly more Mirai bots than the Launcher without zoning. Especially in the case of Distribution 1, which contains 50% Safe Zone, the protecting tactic effectively prevented the devices in the zone from infection. However, the white-hat botnet behaves autonomously. Thus, the formation will collapse with the passage of time, and therefore, the effect of the initial launch position will disappear. This implies that the tactic has a time limit for its effect.

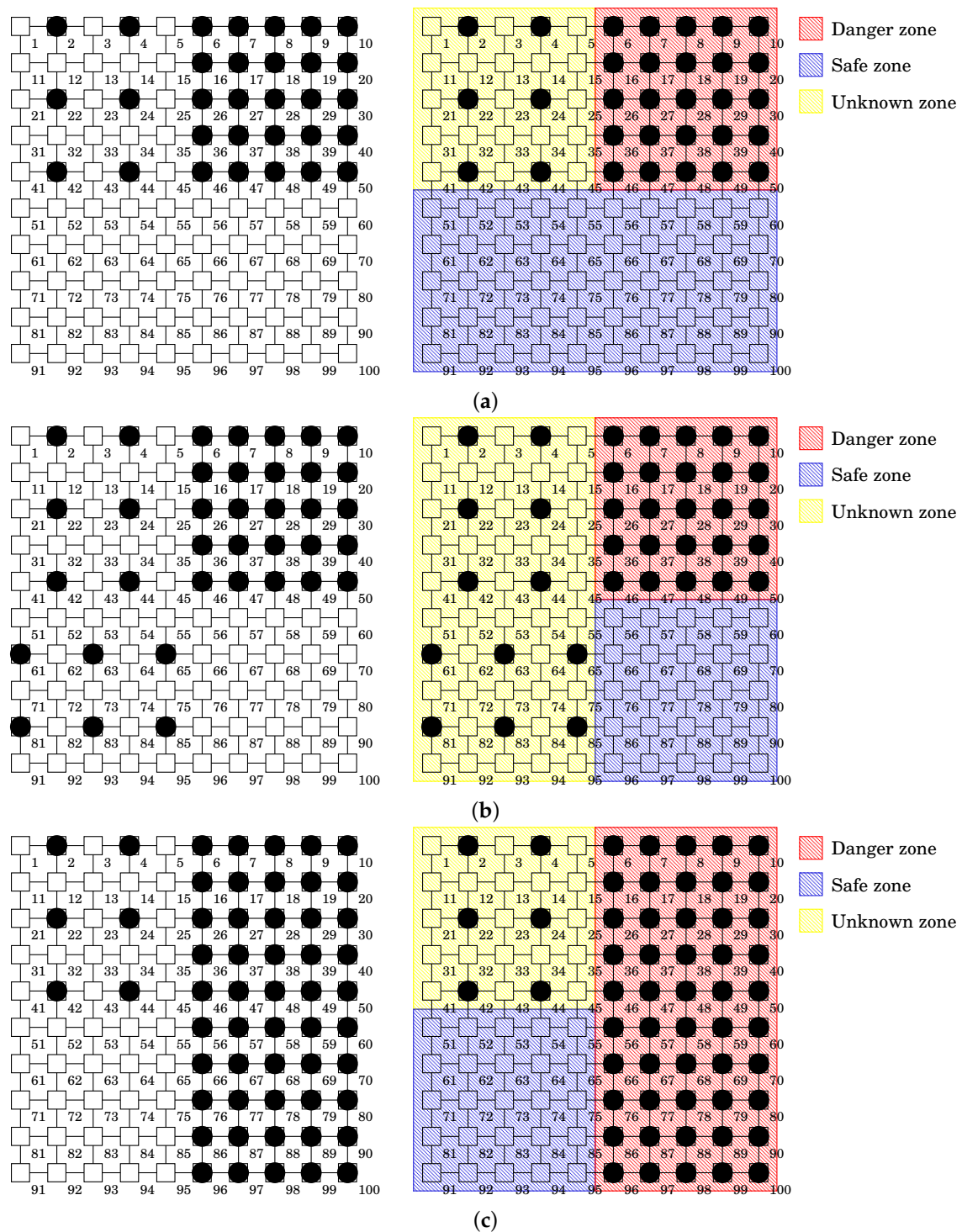


Figure 8. Three kinds of Mirai distributions in a 10 × 10 grid network used for simulation evaluation. (a) Distribution 1 and the result of zoning. (b) Distribution 2 and the result of zoning. (c) Distribution 3 and the result of zoning.

Table 3. Simulation result. Each value represents the mean of infection rate R_{Mirai} for 1000 trials.

(a) Distribution 1		
Step	Without Zoning	With Zoning
1	37.11%	36.97%
10	38.20%	36.99%
100	38.50%	31.89%
1000	8.51%	7.62%
10,000	1.49%	0.82%
(b) Distribution 2		
Step	Without Zoning	With Zoning
1	31.03%	31.01%
10	32.03%	30.70%
100	31.89%	20.71%
1000	6.78%	5.99%
10,000	0.38%	0.29%
(c) Distribution 3		
Step	Without Zoning	With Zoning
1	56.07%	55.89%
10	56.43%	54.86%
100	52.02%	45.47%
1000	11.56%	10.60%
10,000	0.62%	0.60%
(d) Distribution 4		
Step	Without Zoning	With Zoning
1	27.96%	27.82%
10	27.67%	25.99%
100	18.37%	12.20%
1000	6.58%	6.11%
10,000	1.00%	0.97%

In addition, from the graph of Figure 9d, the random distribution resulted in a similar tendency as the other distributions. That is, the infection rate curve with zoning is steeper than without zoning, which means that the infection rate decreases more significantly. We confirmed that the proposed Launcher can also show performance in a general environment and effectively remove Mirai bots.

4.3. Discussion

Table 4 shows the positioning of this study among the related research. Most of the related research focuses on detecting botnets and mitigating the threat, while only the last three research works focus on exterminating existing botnets. The three research works deal with the BDS, but propose different Launchers. The Launcher of [10] gives only a basic functionality, which sends white-hat worms to the infected network. That is, it places worms at random and does not specify concrete nodes. On the other hand, the Launcher of [12] specifies nodes to launch. This Launcher learns how to place white-hat worms based on machine learning. The Launcher also divides the given network into small networks and increases the accuracy of the prediction. The Launcher proposed in this paper introduces the concept of zoning and applies tactics for each zone. In general, there is a bias in the distribution of Mirai bots. Therefore, the given network should be divided by the Mirai bots' density rather than by network scale in a uniform manner. The proposed Launcher divides the given network into zones based on density and can use appropriate tactics depending on the density.

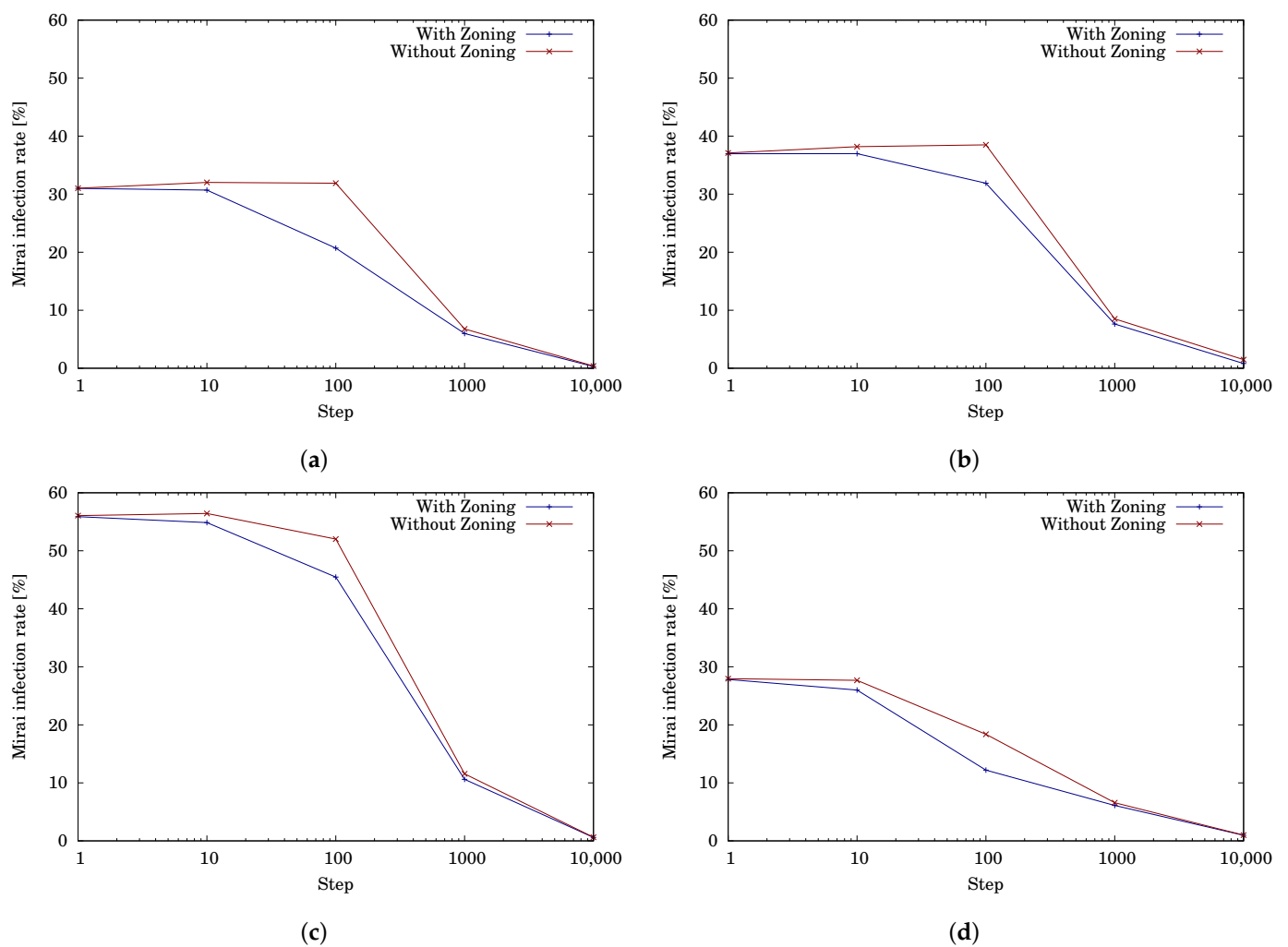


Figure 9. Comparison of the Launchers with and without zoning. (a) Distribution 1, (b) Distribution 2, (c) Distribution 3, and (d) Distribution 4.

Table 4. Positioning of this study among the related works.

Authors	Year	Detection	Mitigation	Extermination	Tactics	Zoning
Qu et al. [17]	2018	✓	×	×	×	×
Nagisetty et al. [14]	2019	✓	×	×	×	×
Vishwakarma et al. [20]	2019	✓	×	×	×	×
Adan et al. [22]	2019	✓	×	×	×	×
Soe et al. [15]	2020	✓	×	×	×	×
Guerra-Manzanares et al. [21]	2020	✓	×	×	×	×
Wang et al. [19]	2020	✓	×	×	×	×
Qu et al. [18]	2022	✓	×	×	×	×
May raju et al. [16]	2022	✓	×	×	×	×
Gopal et al. [27]	2018	×	✓	×	×	×
Manso et al. [28]	2019	✓	✓	×	×	×
Ceron et al. [29]	2019	×	✓	×	×	×
Yamaguchi [10]	2020	×	×	✓	×	×
Pan et al. [12]	2022	×	×	✓	✓	×
This study	2022	×	×	✓	✓	✓

Next, we discuss tactical response by zoning. For the Safe and Danger Zones, we observed that the effect of the tactics for those zones was obvious, especially in the early

stage, which showed that the white-hat botnet can effectively prevent the Mirai botnet from spreading as a “wall”. On the other hand, although it is not easy to isolate randomly distributed Mirai bots, our proposed Launcher showed excellent performance in the case of a random distribution. In general, the proposed Launcher can effectively reduce the Mirai infection rate and reduce the number of infected devices in the network. Zoning helps us make it easier to consider tactics because we separated the phenomenon. In addition, zoning saves time and cost during the design and runtime. This is important because machine learning often requires much computational time and cost.

The proposed Worm Launcher was designed to be scalable. In this paper, we used a 10×10 network to illustrate the proposed method. However, we can apply it to a network with a different size. At this time, we only have to perform padding or use the divide-and-conquer step given in our previous paper as the preprocess. This preprocess hardly takes any time. Actually, we performed a preliminary experiment using a larger network with 20×20 nodes and confirmed that the proposed Launcher is efficient. From the above, it can be said that the proposed Launcher is adaptable to large-scale networks.

5. Conclusions

In this paper, we proposed a novel machine learning white-hat Worm Launcher for tactical response by zoning in the Botnet Defense System. We first introduced the previous work and the BDS. Next, we proposed a new concept of zoning to grasp malicious botnets' spread with bias over IoT networks. We also proposed three tactics to respond to each zone. Then, we confirmed the effect of the proposed white-hat Worm Launcher through a simulation evaluation with agent-oriented Petri net models. The result showed that the proposed Launcher is effective and can reduce the number of infected IoT devices by about 30% overall.

In this paper, we focused on the case of the Mirai botnet infecting only once. However, the Mirai botnet may infect many times. Whenever the Mirai botnet appears, we should nip it in the bud as early as possible. However, when a few Mirai bots are scattered, the proposed Launcher may not work well because all of the network is identified as a Safe Zone. To resolve this problem, we can use several approaches. For example, we can predict future infection situations and make a proactive response based on the prediction. We also can make a white-hat botnet resident to continually disinfect Mirai bots [34].

Author Contributions: Conceptualization, S.Y.; Data curation, X.P.; Formal analysis, S.Y.; Funding acquisition, S.Y.; Investigation, X.P.; Supervision, S.Y.; Writing—original draft, X.P.; Writing—review & editing, S.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by JSPS KAKENHI Grant Numbers JP19K11965 and JP22K12028.

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

References

1. Vishwakarma, R.; Jain, A.K. A survey of DDoS attacking techniques and defence mechanisms in the IoT network. *Telecommun. Syst.* **2020**, *73*, 3–25. [CrossRef]
2. Antonakakis, M.; April, T.; Bailey, M.; Bernhard, M.; Bursztein, E.; Cochran, J. Understanding the mirai botnet. In Proceedings of the 26th USENIX Security Symposium (SEC'17), Ancouner, BC, Canada, 16–18 August 2017.
3. Margolis, J.; Oh, T.T.; Jadhav, S.; Bernhard, M.; Kim, Y.H.; Kim, J.N. An indepth analysis of the mirai botnet. In Proceedings of 2017 International Conference on Software Security and Assurance (ICSSA), Altoona, PA, USA, 1 July 2017.
4. Marzano, A.; Alexander, D.; Fonseca, O.L.H.M.; Fazzion, E.C.; Hoepers, C.; Jessen, K. The evolution of bashlite and mirai iot botnets. In Proceedings of 2018 IEEE Symposium on Computers and Communications (ISCC), Altoona, PA, USA, 1 June 2018.
5. OVH Suffers 1.1 Tbps DDoS Attack. Available online: <https://www.scmagazine.com/news/ddos/ovh-suffers-massive-1-1tbps-ddos-attack> (accessed on 27 September 2016).
6. Kolias, C.; Kambourakis, G.; Stavrou, A.; Voas, J. DDoS in the IoT: Mirai and other botnets. *Computer* **2017**, *50*, 80–84. [CrossRef]
7. Number of Internet of Things (IoT) Connected Devices Worldwide from 2019 to 2030 | Statista. Available online: <https://www.statista.com/statistics/1183457/iot-connected-devices-worldwide/> (accessed on 17 Mar 2022).
8. Edwards, S.; Profetis, I. Hajime: Analysis of a decentralized internet worm for IoT devices. *Rapidity Netw.* **2016**, *16*, 1–18.

9. Yamaguchi, S. White-Hat Worm to Fight Malware and Its Evaluation by Agent-Oriented Petri Nets. *Sensors* **2020**, *20*, 556. [[CrossRef](#)] [[PubMed](#)]
10. Yamaguchi, S. Botnet Defense System: Concept, Design, and Basic Strategy. *Information* **2020**, *11*, 516. [[CrossRef](#)]
11. Yamaguchi, S.; Ahmadon, M.A.B.; Ge, Q.W. Introduction of Petri nets: Its applications and security challenges. In *Handbook of Research on Modern Cryptographic Solutions for Computer and Cyber Security*; Gupta, B.B., Agrawal, D.P., Yamaguchi, S., Eds.; IGI Publishing: Hershey, PA, USA, 2016; pp. 145–179.
12. Pan, X.; Yamaguchi, S.; Kageyama, T.; Kamlin, M.H.B. Machine-Learning-Based White-Hat Worm Launcher in Botnet Defense System. *Int. J. Softw. Sci. Comput. Intell.* **2022**, *14*, 1–14. [[CrossRef](#)]
13. Wazzan, M.; Algazzawi, D.; Bamasaq, O.; Albeshri, A.; Cheng, L. Internet of Things Botnet Detection Approaches: Analysis and Recommendations for Future Research. *Appl. Sci.* **2021**, *11*, 5713. [[CrossRef](#)]
14. Nagisetty, A.; Gupta, G.P. Framework for detection of malicious activities in iot networks using keras deep learning library. In Proceedings of the 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC) IEEE, Surya Engineering College, Erode, India, 27–29 March 2019.
15. Soe, Y.N.; Feng, Y.; Santosa, P.I.; Hartanto, R.; Sakurai, K. Machine Learning-Based IoT-Botnet Attack Detection with Sequential Architecture. *Sensors* **2020**, *20*, 4372. [[CrossRef](#)] [[PubMed](#)]
16. May raju, P.; Gupta, G.P. Intrusion Detection Framework Using an Improved Deep Reinforcement Learning Technique for IoT Network. In *Soft Computing for Security Applications*; Ranganathan, G., Fernando, X., Shi, F., El Alloui, Y., Eds.; Advances in Intelligent Systems and Computing; Springer: Singapore, 2022; pp. 765–779.
17. Qu, Z.; Zhang, Y.; Qu, N.; Wang, L.; Li, Y.; Dong, Y. Method for quantitative estimation of the risk propagation threshold in electric power cps based on seepage probability. *IEEE Access* **2018**, *6*, 68813–68823. [[CrossRef](#)]
18. Qu, Z.; Bo, X.; Yu, T.; Liu, Y.; Dong, Y.; Kan, Z.; Wang, L.; Li, Y. Active and passive hybrid detection method for power CPS false data injection attacks with improved AKF and GRU-CNN. *IET Renew. Power Gener.* **2022**, *16*, 1490–1508. [[CrossRef](#)]
19. Wang, L.; Qu, Z.; Li, Y.; Hu, K.; Sun, J.; Xue, K.; Cui, M. Method for extracting patterns of coordinated network attacks on electric power cps based on temporal-topological correlation. *IEEE Access* **2020**, *8*, 57260–57272. [[CrossRef](#)]
20. Vishwakarma, R.; Jain, A.K. A Honeypot with Machine Learning based Detection Framework for defending IoT based Botnet DDoS Attacks. In Proceedings of the 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI), Tirunelveli, India, 23–25 April 2019; pp. 1019–1024.
21. Guerra-Manzanares, A.; Medina-Galindo, J.; Bahsi, H.; Nmm, S. MedBIoT: Generation of an IoT Botnet Dataset in a Medium-sized IoT Network. In Proceedings of the 2020 6th International Conference on Information Systems Security and Privacy (ICISSP), Berlin/Heidelberg, Germany, 25–27 February 2020; pp. 207–218.
22. María Fernanda Ojeda Adán. Designing an Internet of Things Attack Simulator. Bachelor's Thesis, Metropolia University of Applied Sciences, Vantaa, Finland, 26 February 2019.
23. Sagirlar, G.; Carminati, B.; Ferrari, E. AutoBotCatcher: Blockchain-based P2P botnet detection for the internet of things. In Proceedings of the 2018 IEEE 4th International Conference on Collaboration and Internet Computing (CIC), Philadelphia, PA, USA, 18–20 October 2018.
24. Falco, G.; Li, C.; Fedorov, P.; Caldera, C.; Arora, R.; Jackson, K. Neuromesh: Iot security enabled by a blockchain powered botnetvaccine. In Proceedings of the International Conference on Omni-Layer Intelligent Systems, Crete, Greece, 5–7 May 2019.
25. Hu, J.W.; Yeh, L.Y.; Liao, S.W.; Yang, C.S. A deep convolutional neural network model to classify heartbeats. *Comput. Secur.* **2019**, *86*, 238–252. [[CrossRef](#)]
26. Özçelik, M.; Chalabianloo, N.; Gür, G. Software-defined edge defense against IoT-based DDoS. In Proceedings of the 2017 IEEE International Conference on Computer and Information Technology (CIT), Helsinki, Finland, 21–23 August 2017.
27. Gopal, T. S.; Meerolla, M.; Jyostna, G.; Eswari, P.R.L.; Magesh, E. Mitigating Mirai Malware Spreading in IoT Environment. In Proceedings of 2018 International Conference on Advances in Computing, Communications and Informatics (ICACCI), Bangalore, India, 19 September 2018.
28. Manso, P.; Moura, J. Serrão, C. SDN-Based Intrusion Detection System for Early Detection and Mitigation of DDoS Attacks. *Information* **2019**, *10*, 106. [[CrossRef](#)]
29. Ceron, J.M.; Jessen, K.; Hoepers, C.; Granville, L.Z.; Margi, C.B. Improving IoT Botnet Investigation Using an Adaptive Network Layer. *Sensors* **2019**, *19*, 727. [[CrossRef](#)] [[PubMed](#)]
30. Ester, M.; Kriegel, H.P.; Sander, J.; Xu, X. A density-based algorithm for discovering clusters in large spatial databases with noise. *Kdd* **1996**, *96*, 226–231.
31. Kageyama, T.; Yamaguchi, S. On Tactics to Deploy White-Hat Worms in Botnet Defense System. In Proceedings of the 10th Global Conference on Consumer Electronics (GCCE 2021), Kyoto, Japan, 12–15 October 2021.
32. Ohsaki, K.; Yamaguchi, S. A Proposal of Heterogeneous White-Hat Botnet in Botnet Defense System. In Proceedings of the Sixth International Conference on Consumer Electronics (ICCE) Asia, Gangneung, Korea, 27–29 October 2021.
33. Nakahori, K.; Yamaguchi, S. A Support Tool to Design IoT Services with NuSMV. In Proceedings of the 25th International Conference on Consumer Electronics (ICCE 2017), Las Vegas, NV, USA, 4–8 December 2017.
34. Yamaguchi, S.; Makihara, D. On Resident Strategy for White-Hat Botnet in Botnet Defense System. In Proceedings of the 9th International Conference on Consumer Electronics—Taiwan (ICCE-TW 2022), Taipei, Taiwan, 6–8 July 2022.