A Mathematical Model for Analyzing Honeynets and Their Cyber Deception Techniques

Amir Javadpour*‡, Forough Ja'fari†§, Tarik Taleb*¶, and Chafika Benzaïd*‖
*Faculty of Information Technology and Electrical Engineering, University of Oulu, Oulu, 90570, Finland
†Department of Computer Engineering, Sharif University of Technology, Tehran, Iran
‡a.javadpour87@gmail.com §azadeh.mth@gmail.com ¶tarik.taleb@oulu.fi ‖chafika.benzaid@oulu.fi

***Abstract*—As a way of obtaining useful information about the adversaries behavior with a low rate of false detection, honeypots have made significant advancements in the field of cybersecurity. They are also powerful in wasting the adversaries resources and attracting their attention from other critical assets in the network. A deceptive network with multiple honeypots is called a honeynet. The honeypots in a honeynet aim to cooperate in order to increase their deception power. Professional adversaries utilize strong detection mechanisms to discover the existence of the honeypots in a network. When an adversary finds that a deception mechanism is deployed, it may change their behavior and cause malicious effects on the network. Therefore, a honeynet has to be deceptive enough in order not to be identified. This paper aims to review the techniques that are designed for the honeynets to make them improve their deception performance. The recent related surveys do not focus on the honeynet-specific techniques, and also have no comparison analysis. The main presented techniques in this paper are fully investigated through comparative analysis and simulation scenarios. Some suggestions on the research gap are also provided. The results of this paper can be used by the honeynet developers and researchers to improve their work.**

***Index Terms*—Deception performance, Honeynet efficiency, Honeypot deployment, Professional adversary, Network security.**

I. Introduction

The number of new cyber threats targeting the critical assets in industrial, governmental, and personal networks is growing yearly. Moreover, these threats release different variants with improved malicious features that made them more and more complicated and hard to detect [1, 2, 3, 4, 5]. According to the seriousness of the effects of the cyber threats on computer networks, the defenders attempt to apply deception techniques to their network, by which they can detect and analyze the unknown threats, and also prevent the dangerous ones [6, 7]. Even though different security mechanisms, such as intrusion detection and prevention systems, are designed for mitigating the attacks, they are not efficient enough (1) to detect zero-day and unknown threats, and (2) to closely analyze the adversary's behavior. Honeypot is a deceptive tool that is capable of helping the network defenders to achieve the two above-mentioned goals [8, 9, 10].

Honeypots confuse the adversaries and waste their resources, creating the ambiguity of achieving the goal in the adversary. They use deception techniques, and hence, they are always one step ahead of the adversaries [11]. There are different types of honeypot deployment methods in a network, such as Minefield, Shield, and Honeyfarm. Generally speaking, a deceptive network with one or more honeypots is called a honeynet despite the different deployment types. The honeypots can provide different fake services such as web and database-related services. Traffic to the honeypots can first pass through a honeywall, such as Roo. Iptables can also restrict the communication between the honeypots and the real systems [12].

This paper presents a wide range of researches in the field of honeynets and their deception techniques. Several surveys exist in this field, among them are the recent researches performed by Fraunholz et al. [13], Razali et al. [14], Zobal et al. [15], Seungjin et al. [16], and Lackner [17]. However, these researches do not mention the deception techniques that are specifically used by the honeynets, and also lack a comparative analysis of the deception methods. This paper aims to give a comprehensive review of honeynet researches and their introduced deception techniques along with investigating these researches based on comparative and simulation results. The main contributions of this paper are as follows:

- Suggesting a model for presenting general honeynets and their parameters, that also helps in comparing the current honeynet researches.
- Categorizing the deception techniques of honeynets and comparing them with simulation scenarios.
- Giving practical and detailed suggestions according to the research gap and future directions of honeynets.

This paper is structured as follows. In section II, we present the general model of representing the honeynets. Then, we explain the honeynets deception techniques in section III. The comparison of these techniques is also mentioned. In section IV, we provide some suggestions to improve the performance of honeynets deception techniques. Finally, the paper concludes in section V.

II. Honeynets Model

A honeynet is a deceptive network of decoys that places several honeypots in different network locations to take advantage of the collaboration between these honeypots. This community of honeypots is more effective than using a single decoy in a network. Now, we suggest a general representation of honeynets, which can match with the current works in this field. By this new representation, all the game models mentioned in this section can be compared. A honeynet, $\mathcal{H}$,

978-1-6654-4407-1/21/$31.00

can be written as $\mathcal{H} = \{\mathcal{N}, \mathcal{S}, \mathcal{C}, \mathcal{B}\}$, where $\mathcal{N}$ is the network characteristics, $\mathcal{S}$ is the detail of the players strategies, $\mathcal{C}$ is the set of costs that the players may pay, and $\mathcal{B}$ is the set of benefits that the players may obtain.

$\mathcal{N}$ can be written as $\mathcal{N} = \{h, r, z, D\}$, where h, r, z are the number of honeypots, the number of real hosts that are vulnerable to cyber attacks, and the number of other safe hosts in the network, respectively. D is a list of hosts' degrees, where d_i is the i^{th} host degree and d_{max} is the maximum degree value among all the hosts.

We have $\mathcal{S} = \{ma, ah, sr_h, sr_r, ar_a, sr'_h, sr'_r, ar'_a, oh\}$, where ma is the maximum number of attack attempts that can be performed by the adversary and ah is the adversary's acceptable number of connections to honeypots. sr_h and sr_r are the service rate of honeypots and real hosts, respectively. ar_a is the adversary's attack rate and the symbols with a quote are the reduced factor of that symbol after applying a specific strategy. oh is the optimal number of required honeypots found in the defender's best strategies.

The set of costs for each operation can be written as $\mathcal{C} = \{pc_h, pc_r, ac_h, ac_r, pc'_h, pc'_r, ac'_h, ac'_r, cc, dc, rc, mc\}$, where pc_h and pc_r are the adversary's cost of probing a honeypot and a real host, respectively. ac_h and ac_r are the adversary's cost of attacking a honeypot and a real host, respectively. The symbols with a quote are the same cost for the defender. cc is the adversary's cost of being caught by honeypots, dc is the defender's cost of deploying a honeypot in the network, and rc is the defender's cost of responding to the adversary's attempts. Finally, mc is the maximum acceptable attacking cost for the adversary.

For the benefits set, we have $\mathcal{B} = \{ab_h, ab_r\}$, where ab_h and ab_r are the benefits of successfully attacking a honeypot (some adversaries can compromise honeypots to control them) and a real host, respectively.

III. Deception Improvement Techniques

In this section, we have presented deception techniques that are used to improve honeynets performance. In addition, we have mentioned the suggested strategies in each research represented with our model, $\mathcal{H}$. We have also simulated comparable works to analyze their performance. A summarization of the related researches in this field is presented in Table I. It is worth noting that for some of the researches, that has no names, we have chosen a related name.

A. Optimizing the Honeypots

One of the most significant network parameters in honeynets is the number of deployed honeypots. Placing a few number of honeypots in the network is not sufficient to lure the adversary and protecting the entire network. On the other hand, using too much honeypots in the network is costly, and in some cases, it may warn the adversaries about the deception mechanism they are facing. Therefore, finding the optimal number of honeypots for a network is necessary.

Rowe et al. [18] have proposed a mathematical model, TBM (Tolerance-Based Model), to calculate the adversary's cost and benefit in attacking a honeynet. Using TBM, the network defender can set an appropriate number of honeypots to make the attacking cost higher than the adversary's expected cost.

The network defender in the previous model must have the information about the adversary's perceived cost and benefit. Another model is proposed by Crouse [19] to find the optimal number of honeypots without adversary's perceived parameters. A honeynet is modeled as an urn that contains three types of beads with different colors. We call this model as URN. When the adversary attacks a host, a bead is removed from the urn. This model assumes that the adversary is successful if it could launch an attack against at least one of the vulnerable hosts. The network defender can calculate the number of required honeypots to quantify the adversary's successful attack rate threshold.

For some professional adversaries, connecting to a few number of honeypots does not reveal their identity. Hence, they accept to communicate with a specific number of honeypots. Crouse et al. [20] have proposed a similar urn model, called URNt (URN with a threshold), that also considers an acceptable number of connections to the honeypots. The probability of a successful attack is calculated, and then, the network defender can make the best decision to choose the appropriate deception method by examining the probability of the adversary's victory. The previous urn models do not consider the probing process that the adversary performs before launching the attacks. Therefore, Fraunholz and Schotten [21] have proposed a game, called GBO (Game-Base Optimizer), that also models the probing process and it is solved based on Stackelberg competition. The model suggests setting the number of honeypots so that the attacking payoff is equal to the not attacking payoff. In this situation, the adversary is forced to not attack the hosts.

To compare the performance of the researches mentioned in this section, we have simulated several random networks with different characteristics in Python, and apply the optimal metric suggested by each model. Each network has a total number of 100 nodes, and the number of honeypots varies in each scenario from zero to 50. The adversary probes the network and his/her goal is to compromise at least 20 real hosts in the network. Connecting to more than 4 honeypots leads to the adversary's failure. The details about the adversary's attacking scenario are presented in Algorithm 1.

The results are shown in Figure 1. The changes in the number of successful attacks, which is indicated by "success", and the changes of the four models' metrics, are shown in this figure. The growth and decay of the metrics compared with the adversary's success metric show that URNt model can get better results than the other three models in our scenarios.

B. Diversifying the Honeypots

The type of honeypots that are located in the network is another major parameter that affects the honeynet performance. There are different types of honeyepots with various capabilities. Hence, the network defender must utilize appropriate types according to their deployment cost. Moreover, this

TABLE I
COMPARING THE RESEARCHES IN THE FIELD OF DECEPTION IN HONEYNETS.

Research	Purpose	Key Characteristic	Model Type
TBM [18]	Optimizing	Considering the adversary's tolerance	Mathematical model
URN [19]		Considering the safe hosts in the model	Mathematical model
URNt [20]		Considering a threshold for connecting to a honeypot	Mathematical model
GBO [21]		Considering the number of probes before the attack	General two-player game
HSGp [22]	Diversifying	Assigning each host a numerical importance value	Zero-sum game
DHG [23]		Assigning limited importance values to the hosts	General two-player game
DHGu [24]		Assuming the adversary does not know the honeypot types	General-sum game
DHD [25]		Considering honeypot detection techniques	Zero-sum game
POSG [26]	Locating	Considering unknown attack graph safety level for the defender	Zero-sum game
POSGm [27]		Letting the defender to place multiple honeypots on the attack graph	Zero-sum game
DD [28]	Dynamizing	Assuming contiguously located honeypots	General two-player game
SGM [29]		Considering the attacks with and without probing	Signaling game
HDG [30]		Considering multiple probes for the adversary	Extensive form game
CSG [31]		Considering adversary's obtained evidence	Signaling game
SGE [32]		Considering adversary's obtained evidence based on hosts activities	Signaling game
eHDG [33]		Assuming that both the adversaies and the legitimate users may probe	General two-player game
DTG [34]	Shaping	Considering different topological characteristics	Differential game
VTG [35]		Considering various network topologies	Mathematical model

Algorithm 1 The adversary's attack process in the simulations for comparing the researches about optimizing the honeypots.

```
honeypots ← the list of honeypot nodes
hosts ← the list of real host nodes
network ← honeypots + hosts
compromised, checked ← two empty lists
payed ← 0
accpeted_cost = 20 × host_cost + 4 × honeypot_cost
while True do
    if size(compromised) = 20 then
        log a successful attack
        break
    if size(checked) = size(network) then
        log an unsuccessful attack
        break
    if payed > accepted_cost then
        log an unsuccessful attack
        break
    target ← a random node from network − checked
    add target to checked
    if target ∈ honeypots then
        payed ← payed + honeypot_cost
    else
        payed ← payed + host_cost
        add target to compromised
```

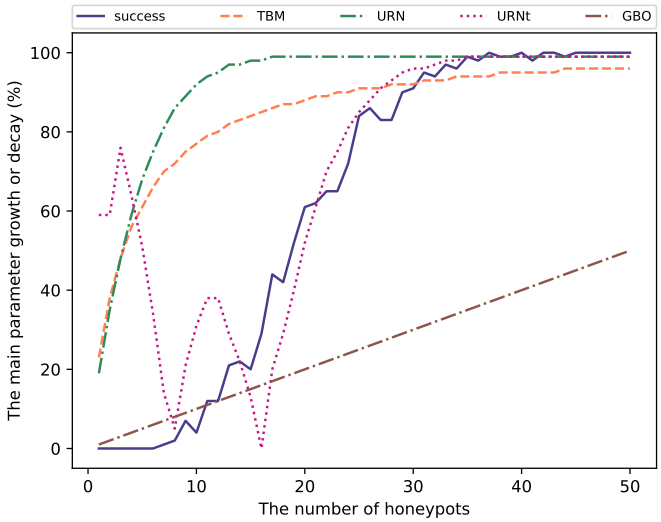

Fig. 1. Comparing the researches about optimizing the honeypots.

diversity helps the defender prevent more honeypot detection attacks. Because if the adversary finds a way to detect a honeypot of type t, the other type t honeypots can be detected in a similar way. But, if the honeypots are of different types, detecting one of the honeypots may not lead to simply detecting all of them.

Píbil et al. [22] have proposed a zero-sum game model, called HSG (Honeypot Selection Game), for a honeynet with different types of honeypots. A numerical value is assigned to each real host and honeypot. The importance values for the honeypots are fake as they pretend to be that much important. The adversary aims to attack a real host, while the defender tries to optimally deploy a fixed number of honeypots but with different types to increase the probability of a honeypot being attacked. The adversary is not able to probe the hosts before attacking them. Hence, in the same research, another model called HSGp is proposed to support the probing process, in which the adversary's resources are limited. This research has suggested to find the optimal strategy for the defender by Nash equilibrium concepts.

Only specific numbers are allowed to be assigned to the honeypots importance values. As a result, Durkota et al. [23] have proposed a game model, called DHG (Diversifying Honeynet Game), for presenting a honeynet with different but limited honeypot types. The adversary does not know which host is a honeypot. It creates the attack graph of the honeynet and analyzes it to find the optimal attack path. The best defensive strategy can be calculated by Stackelberg equilibrium. In DHG, the adversary pays a specific cost when a honeypot is attacked, and gets reward for successfully attacking a real

host. Since DHG assumes that the adversary knows the types of honeypots, another model, called DHGu (DHG with an unaware adversary), is proposed by Durkota et al. [24] that assumes the adversary is unaware of the honeypot types. DHGu presents a honeynet with a general-sum game. The first move is performed by the defender to optimally place a honeypot. Deploying these honeypots may have different costs and bring different security levels. The adversary knows the number of honeypots, but does not have any information about their type and location. However, by calculating the probability of accessing a honeypot and the cost of a successful attack, a host is selected to be compromised. This research has proposed approximate solutions to find the best strategies.

To analyze the honeypot detection attacks, Sarr et al. [25] have proposed a zero-sum game model, called DHD (Diversifying to mitigate Honeypot Detection), in which the defender attempts to reduce the chance of successful honeypot detection attacks by increasing the cost of detecting all honeypot types. The defender pays a specific cost for deploying each type of honeypots. On the other hand, the adversary obtains a reward by detecting a honeypot, but pays the same deployment cost of a type t honeypot as the detection cost. This research suggested not deploying same type honeypots and diversify them randomly in the network to decrease the chance of honeypot detection.

C. Locating the Honeypots

The honeypots location is also important in deceiving the adversary. Two honeynets with the same number of honeypots may have different defensive performance according to their placement strategy. The attack graph of a network can be used to finding the appropriate location for the honeypots. An attack graph is a directed graph that represents the beginning and the ending states of different intrusions in the network. The edges in this graph show the process of exploiting the vulnerabilities. The honeypots must be located between two nodes of this graph that are connected with an edge, in a way that the network seems to be more vulnerable.

Anwar et al. [26] have modeled the problem of placing the honeypots as a zero-sum stochastic game, and the model is called POSG (Partially Observable Stochastic Game). Both the players know the attack graph in POSG, however, the adversary does not know which network node is a honeypot, and the defender is not aware of which vulnerability has been exploited by the adversary. In each step, the defender can choose to place a single honeypot on a specific edge, and the adversary selects a host to exploit. The POSG researchers have not mentioned a specific strategy for the players, and they have suggested using approximate methods to find the optimal one.

Since the defender can place only a single honeypot in each step of POSG, Anwar et al. [27] proposed another zero-sum game model, called POSGm (POSG with multiple honeypots), to make the model more realistic by placing multiple honeypots. In POSGm, first a linear equation is suggested to calculate the reward function of the players, and then a progressive algorithm is proposed to check all possible game states in future steps and find the best strategies.

D. Dynamizing the Honeypots

One of the deception techniques that can be utilized in honeynets is changing the behavior of real hosts or honeypots and responding to the adversary dynamically. When an adversary probes one of the hosts, the network defender decides whether to show that host as a real host or as a honeypot. This technique increases the adversary's uncertainty about the hosts types. There is a trade-off between the deception level obtained by lying and the cost of configuring related mechanisms.

Cai et al. [28] have proposed a two-player game model, in which the honeypots are contiguously located in the address space of the network. The number of lies in this model are limited. The adversary tries to find the block of honeypots with the lowest possible number of probes, and the defender aims to increase the number of these probes. This research has suggested the defenders to use a strategy, called Delay-Delay (DD), in which the honeypots always tell lie until their limitation is exceeded.

The honeypots are not always contiguously located within the address space. Carroll and Grosu [29] have proposed another model, called SGM (Signaling Game Model), based on signaling games, in which the network utilizes the honeypots in random places. When the adversary probes a host, the defender can respond with 'h' or 'r' to show the probed host is a honeypot or a real system, respectively. SGM introduces specific situations in which the defender must respond with 'r', and in other situations, it is better to respond with 'h'.

In SGM, the adversary can only probe a single host in each step, while in reality, several hosts may be probed. Hence, Garg and Grosu [30] have proposed another game model, called HDG (Honeypot Deception Game), to better represent the honeynets in dynamizing scenarios. HDG is an extensive form game, the players of which move alternatively until the adversary probes a specific number of hosts. At the final step, the adversary decides to attack one of the hosts or not. HDG suggests for the defender to respond in such a way that the probability of getting a true or lie response be equal.

Some professional adversaries seek for evidence to probabilistically check whether a response is lie or truth. For example, a honeypot may simulate mouse movements to act normal. However, the adversary finds out the fake movement with uncommon patterns. Two models have been proposed that take the adversary's evidence into consideration, both of which are based on cheap-talk signaling games. In the first model, called CSG (Cheap-talk Signaling Game) and proposed by Pawlick and Zhu [31], the adversary finds evidences, and then, decides whether to launch an attack or not. CSG suggests finding the optimal strategy using perfect Bayesian Nash equilibrium. However, an exact optimal strategy is not explained. Pawlick et al. [32] have proposed the other game, called SGE (Signaling Game with Evidence). If a host has a high activity level, it is probably a real host, and low activity level is the main clue of a honeypot. Though, the defender can

Algorithm 2 The process of the simulated scenario for comparing the researches about dynamizing the honeypots.

```
network ← the list of the network nodes
checked, belief ← two empty lists
for i from 1 to 40 do
    target ← a random node from network − checked
    add target to checked
    response ← probe(network, target)
    accept ← a random number between 0 and 100
    if accpet < true_evidence_rate then
        if response = a real host then
            add target to belief
    else
        if response = a honeypot then
            add target to belief
if size(belief) > 0 then
    target ← a random node from belief
    if target is a real host then
        log a successful attack
    else
        log an unsuccessful attack
else
    log an unsuccessful attack
```

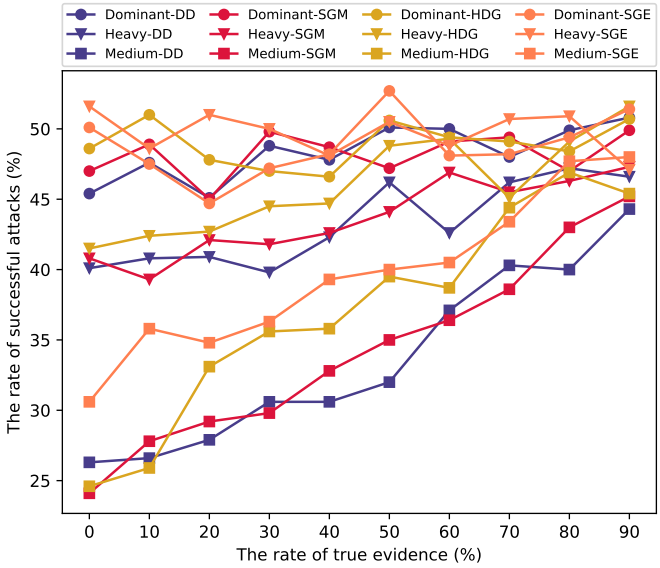

Fig. 2. Comparing the performance of DD, SGM, HDG, and SGE strategies in different scenarios.

change the activity level to lure the adversary. The optimal strategy for the defender in SGE is to make the adversary's evidence useless. This model considers three different states (i.e., dominant, heavy, and middle), according to the number of real hosts and honeypots, and then suggests the optimal strategy in each of them.

The previous models have assumed that only the adversary probes the networks. In real situations, some benign hosts also probe the network and communicate with other hosts. So, Bilinski et al. [33] have extended HDG model, and proposed a Bayesian game model, called eHDG (extended HDG), in which the first player is the network defender, and the second player is an adversary or a benign node with a specific probability. The suggested strategy consists in balancing the number of lies between all the hosts.

To compare the main models, we have simulated several scenarios in Python and apply the optimal strategy of each of these four models. Each simulated network is in one of the states introduced in SGE, which are Dominant, Heavy, and Middle states. Each network contains 100 nodes, and 5, 20, and 50 nodes are honeypots in Dominant, Heavy, and Middle states, respectively. The honeypots are randomly distributed in the network, and the adversary also randomly scans the network. The adversary probes up to 40 hosts, and the network defender can lie at most 30 times in our scenarios. Finally, the adversary selects a target among the probed hosts and launches an attack against it. If the target is a real host the attack is successful, and otherwise, the adversary fails. The detailed process of the simulated scenarios is explained in Algorithm 2. The probe() function that is mentioned in Algorithm 2 is the target node type, which may be a lie or a truth, according to the dynamizing model. The results of each scenario are shown in Figure 2. We can see that on average, DD and SGM exhibit higher performance than HDG and SGE in our scenarios.

E. Shaping the Honeynet

A honeynet topology is another factor to be considered in analyzing its performance. The connections between the hosts can have a significant influence on deception power. For example, in a network with full mesh topology, all the hosts are directly connected to each other, and the propagation of a malware in such networks is fast. Hence, the network defender must pay attention to the honeynet topology to get better performance.

Ren and Zhang [34] have proposed a differential game, called DTG (Differential Topological Game), to analyze this effect. In DTG, the adversary attempts to find the best rate of infecting the network hosts to launch a DDoS attack, and the honeynet tries to reduce the propagation rate of the adversary's malware with the lowest cost. It is stated in this research that the topological degree of each honeypot can significantly influence its performance. For example, higher-degree honeypots can capture more attacks than the lower-degree ones. But, the lower-degree honeypots are more appropriate for recovery processes. This research also suggested that for scale-free networks, in which the degrees follow a power law distribution, higher exponent is more efficient in preventing the attacks.

In addition to scale-free networks, other topologies must also be considered. The influence of some typical topologies on deception performance is investigated by Ren et al. [35], which has proposed a model, called VTG (Various Topologies to model the Game), in which the honeynet is partially infected by a malware and its spreading speed is checked in different network topologies such as ring, star, tree and scale-free. The results of this research stated that if the maximal characteristic value of the honeynet adjacency matrix (λ_{max}) is less than the ratio of recovery rate of the infected hosts

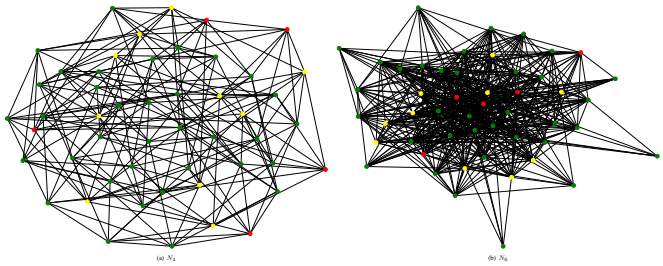

Fig. 3. The simulated networks with different topologies.

Algorithm 3 The malware propagation process in the simulations for comparing the researches about shaping the honeynet.

```
honeypots ← the list of honeypot nodes
infecteds ← the list of infected nodes
for each infection interval do
    for i ∈ infecteds do
        neighbors ← the list of node i's neighbor nodes
        flag ← a random number between 0 and 10
        if flag < infection_rate × 10 then
            for j ∈ neighbors do
                if j ∈ honeypots then
                    remove i from infecteds
                    break
                add j to infecteds
```

to the malware infection rate, the honeynet can experience a high level performance. The research suggested keeping the greatest host degree low for having an efficient honeynet.

To analyze the researches performed in the field of honeynets topologies, we have simulated six networks (i.e., N_1 to N_6) in Python with different topologies which are infected by a malware. As an example, N_4 and N_6 are shown in Figure 3. N_1, N_2, and N_3 use ring, star, and tree topologies, respectively. N_4 uses k-regular topology, in which all the nodes have seven neighbor nodes. Finally, N_5 and N_6 are scale-free networks, and N_5 has a lower exponent than N_6. All the simulated networks have 50 nodes, among which 10 nodes are honeypots and five nodes are initially infected by the malware. The honeypots and the initially-infected hosts are placed randomly in the network, and they are shown by yellow and red nodes in Figure 3, respectively. The infected hosts can connect to normal hosts and exploit them with a specific probability. If an infected host connects to a honeypot, it will be recovered. The detail of the malware propagation process is mentioned in Algorithm 3.

The results of these simulations is shown in Figure 4. N_6 has the lowest number of infected hosts and it can better prevent the spreading of a malware in our scenarios. Since N_6 and N_5 are scale-free networks, and the exponent in N_6 is higher, we can say that the suggestion of DTG model in our scenarios is acceptable. On the other hand, the values of λ_{max} for N_1 and N_4 are two and seven, and for N_2 this value is greater than seven. λ_{max} for N_3 is greater than two and less than seven. However, a direct relation between λ_{max} and the final number of infected hosts is not observed.

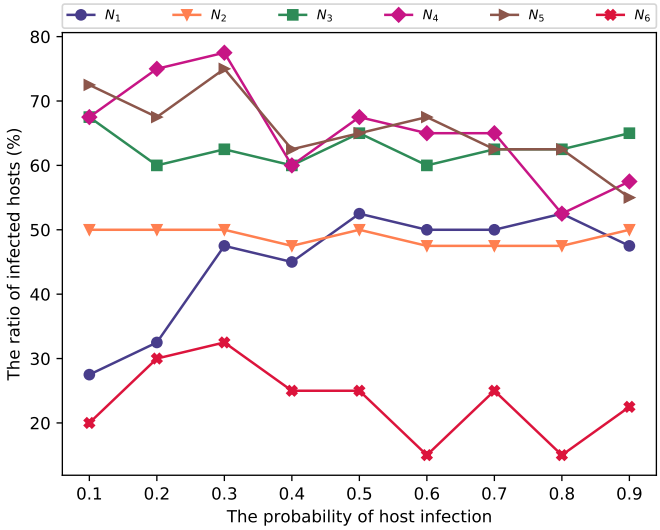

Fig. 4. Comparing the malware propagation process in different scenarios.

IV. Suggestions

After reviewing the mentioned techniques, we face several research gaps in the field of honeynets. They can be further investigated by the researches, and our suggestions are as follows. The *diversifying* technique can be improved by machine learning approaches. One can train a learning model to predict the services that are commonly targeted by the current threat spread over the network using real-world datasets. This prediction helps honeypots to simulate the services that can attract the adversaries with a higher rate. We suggest using Moving Target Defense paradigm [36] to change the position of honeypots in a honeynet, which can lead to a lower cost of deployment and a higher efficiency in wasting the adversaries resources, and hence, a more powerful *locating* technique. In none of the mentioned techniques, the impact of the others is considered. For example, the *shaping* technique is not independent of the optimizing technique. If the resources are limited, and only two of the systems can act as a honeypot, the honeynet topology may be very different compared with a situation of having 20 honeypots. As a result, in order to improve a technique, we suggest considering the parameters of other ones, as well.

V. Conclusion

In this paper, we have presented a wide range of honeynet researches. We have explained the basic concepts about the honeynets and then suggested a general model for presenting them. Then, we have mentioned the deception techniques that have been used to improve the performance of honeynets. These techniques are categorized as optimizing, diversifying, locating, dynamizing, and shaping the honeypots. Since we have modeled these techniques based on the proposed general model, they have become comparable. For the main techniques in this field, we have simulated different scenarios in Python to analyze them.

ACKNOWLEDGMENT

This work was supported in part by the EU's HE research and innovation programme HORIZON-JU-SNS-2022 under the RIGOUROUS project (Grant No. 101095933), the Academy of Finland IDEA-MILL project under Grant No. 352428, and the Academy of Finland 6Genesis Flagship Project (Grant No. 346208).

REFERENCES

[1] F. Ja'fari, S. Mostafavi, K. Mizanian, and E. Jafari, "An intelligent botnet blocking approach in software defined networks using honeypots," *Journal of Ambient Intelligence and Humanized Computing*, vol. 12, no. 2, pp. 2993–3016, 2021.

[2] A. Javadpour, F. Ja'fari, T. Taleb, and C. Benzaïd, "Reinforcement learning-based slice isolation against ddos attacks in beyond 5g networks," *IEEE Transactions on Network and Service Management*, 2023.

[3] A. Javadpour, P. Pinto, F. Ja'fari, and W. Zhang, "Dmaidps: a distributed multi-agent intrusion detection and prevention system for cloud iot environments," *Cluster Computing*, vol. 26, no. 1, pp. 367–384, 2023.

[4] C. Benzaïd and T. Taleb, "Zsm security: Threat surface and best practices," *IEEE Network*, vol. 34, no. 3, pp. 124–133, 2020.

[5] C. Benzaïd, B. Mohammed, and T. Taleb, "Robust self-protection against application-layer (d)dos attacks in sdn environment," in *Proc. of IEEE Wireless Communnications and Networking Conference (WCNC)*. IEEE, 2020, pp. 1–6.

[6] A. Javadpour, F. Ja'fari, T. Taleb, and M. Shojafar, "A cost-effective mtd approach for ddos attacks in software-defined networks," in *GLOBECOM 2022-2022 IEEE Global Communications Conference*. IEEE, 2022, pp. 4173–4178.

[7] C. Benzaïd, T. Taleb, and M. Z. Farooqi, "Zsm security: Threat surface and best practices," *IEEE Network*, vol. 35, no. 3, pp. 212–222, 2021.

[8] J. M. J. Valero, M. G. Pérez, A. H. Celdrán, and G. M. Pérez, "Identification and classification of cyber threats through ssh honeypot systems," in *Handbook of Research on Intrusion Detection Systems*. IGI Global, 2020, pp. 105–129.

[9] C. Benzaïd, T. Taleb, and J. Song, "Ai-based autonomic and scalable security management architecture for secure network slicing in b5g," *IEEE Network*, vol. 36, no. 6, pp. 165–174, 2022.

[10] K. Samdanis, A. N. Abbou, J. Song, and T. Taleb, "Ai/ml service enablers & model maintenance for beyond 5g networks," *IEEE Network*, 2023.

[11] H. Wang, H. He, W. Zhang, W. Liu, P. Liu, and A. Javadpour, "Using honeypots to model botnet attacks on the internet of medical things," *Computers and Electrical Engineering*, vol. 102, p. 108212, 2022.

[12] R. Gautam, S. Kumar, and J. Bhattacharya, "Optimized virtual honeynet with implementation of host machine as honeywall," in *2015 Annual IEEE India Conference (INDICON)*. IEEE, Dec 2015, pp. 1–6.

[13] D. Fraunholz, S. D. Anton, C. Lipps, D. Reti, D. Krohmer, F. Pohl, M. Tammen, and H. D. Schotten, "Demystifying deception technology: A survey," *arXiv preprint arXiv:1804.06196*, 2018.

[14] M. F. Razali, M. N. Razali, F. Z. Mansor, G. Muruti, and N. Jamil, "Iot honeypot: A review from researcher's perspective," in *2018 IEEE Conference on Application, Information and Network Security (AINS)*. IEEE, 2018, pp. 93–98.

[15] L. Zobal, D. Kolář, and R. Fujdiak, "Current state of honeypots and deception strategies in cybersecurity," in *2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. IEEE, 2019, pp. 1–9.

[16] L. Seungjin, A. Abdullah, and N. Jhanjhi, "A review on honeypot-based botnet detection models for smart factory," *International Journal of Advanced Computer Science and Applications*, vol. 11, no. 6, pp. 418–435, 2020.

[17] P. Lackner, "How to mock a bear: Honeypot, honeynet, honeywall & honeytoken: A survey," 2021.

[18] N. C. Rowe, E. J. Custy, and B. T. Duong, "Defending cyberspace with fake honeypots." *J. Comput.*, vol. 2, no. 2, pp. 25–36, 2007.

[19] M. B. Crouse, "Performance analysis of cyber deception using probabilistic models," Master's thesis, Wake Forest University Graduate School of Arts and Sciences, Winston-Salem, North Carolina, 2012.

[20] M. Crouse, B. Prosser, and E. W. Fulp, "Probabilistic performance analysis of moving target and deception reconnaissance defenses," in *Proceedings of the Second ACM Workshop on Moving Target Defense*. ACM, 2015, pp. 21–29.

[21] D. Fraunholz and H. D. Schotten, "Strategic defense and attack in deception based network security," in *2018 International Conference on Information Networking (ICOIN)*. IEEE, 2018, pp. 156–161.

[22] R. Píbil, V. Lisỳ, C. Kiekintveld, B. Bošanskỳ, and M. Pěchouček, "Game theoretic model of strategic honeypot selection in computer networks," in *International conference on decision and game theory for security*. Springer, 2012, pp. 201–220.

[23] K. Durkota, V. Lisỳ, B. Bošanskỳ, and C. Kiekintveld, "Optimal network security hardening using attack graph games," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.

[24] ——, "Approximate solutions for attack graph games with imperfect information," in *International Conference on Decision and Game Theory for Security*. Springer, 2015, pp. 228–249.

[25] A. B. Sarr, A. H. Anwar, C. Kamhoua, N. Leslie, and J. Acosta, "Software diversity for cyber deception," in *GLOBECOM 2020-2020 IEEE Global Communications Conference*. IEEE, 2020, pp. 1–6.

[26] A. H. Anwar, C. Kamhoua, and N. Leslie, "A game-theoretic framework for dynamic cyber deception in internet of battlefield things," in *Proceedings of the 16th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*, 2019, pp. 522–526.

[27] ——, "Honeypot allocation over attack graphs in cyber deception games," in *2020 International Conference on Computing, Networking and Communications (ICNC)*. IEEE, 2020, pp. 502–506.

[28] J.-Y. Cai, V. Yegneswaran, C. Alfeld, and P. Barford, "An attacker-defender game for honeynets," in *COCOON*. Springer, 2009, pp. 7–16.

[29] T. E. Carroll and D. Grosu, "A game theoretic investigation of deception in network security," *Security and Communication Networks*, vol. 4, no. 10, pp. 1162–1172, 2011.

[30] N. Garg and D. Grosu, "Deception in honeynets: A game-theoretic analysis," in *2007 IEEE SMC Information Assurance and Security Workshop*. IEEE, June 2007, pp. 107–113.

[31] J. Pawlick and Q. Zhu, "Deception by design: evidence-based signaling games for network defense," *arXiv preprint arXiv:1503.05458*, 2015.

[32] J. Pawlick, E. Colbert, and Q. Zhu, "Modeling and analysis of leaky deception using signaling games with evidence," *IEEE Transactions on Information Forensics and Security*, vol. 14, no. 7, pp. 1871–1886, 2018.

[33] M. Bilinski, R. Gabrys, and J. Mauger, "Optimal placement of honeypots for network defense," in *International Conference on Decision and Game Theory for Security*. Springer, 2018, pp. 115–126.

[34] J. Ren and C. Zhang, "A differential game method against attacks in heterogeneous honeynet," *Computers & Security*, vol. 97, p. 101870, 2020.

[35] J. Ren, C. Zhang, and Q. Hao, "A theoretical method to evaluate honeynet potency," *Future Generation Computer Systems*, vol. 116, pp. 76–85, 2021.

[36] C. Benzaïd and T. Taleb, "Ai for beyond 5g networks: a cyber-security defense or offense enabler?" *IEEE network*, vol. 34, no. 6, pp. 140–147, 2020.