Network Defence Strategy Evaluation: Simulation vs. Live Network

Jana Medková*†, Martin Husák*†, Martin Drašar*
*Institute of Computer Science, †Faculty of Informatics
Masaryk University
Brno, Czech Republic
{medkova, husakm, drasar}@ics.muni.cz

***Abstract*—A lot of research has been dedicated to finding an optimal strategy to defend network infrastructure. The proposed methods are usually evaluated using simulations, replayed attacks or testbed environments. However, these evaluation methods may give biased results, because in real life, attackers can follow a suboptimal strategy or react to a defence in an unexpected way. In this paper, we use a network of honeypots as a testing environment for evaluating network defence strategies. The honeypot network provides the opportunity to test a defence strategy against real attackers and is not as time and resource consuming as using white hat hackers. In our experiment, we use two different strategies to defend a group of honeypots in a live network and we compare these results to the results of a simulation with replayed attacks. We show that the results of the strategies in the simulation significantly differ from the results on the honeypot network which implies simulations are not sufficient for strategy evaluation. We also investigate how the attacker adapts to the responses taken by a defence strategy and how this change in behaviour affects the evaluation results.**

I. Introduction

A lot of research has been dedicated to finding an optimal strategy to defend network infrastructure [1]. There are many theoretical approaches, each having its advantages and disadvantages. Although many of the proposed methods work well in theory, there are too few cases of them being evaluated and verified in practice.

Network defence strategies are usually evaluated in a simulated environment. Deployment in a real environment requires a lot of effort to set up and is risky, because the test network could be abused by the attacker. Internal evaluation with a white hat hacker would mitigate the risk, however, it is even more time-consuming and getting a significant number of results for a detailed analysis is almost impossible.

Simulated scenarios are easily set up and do not pose any risks. A defence strategy is tested against selected attacks and the efficiency of the defence strategy's response is evaluated. The attacks are usually based on attacks from public datasets, attack vectors or, sometimes, an optimal attack strategy obtained as a by-product of defence strategy synthesis.

The evaluation in a simulated environment is extremely useful for the initial evaluation of a defence strategy and has a lot of advantages. It is very easily set up, the results are almost immediately available, the datasets provide a wide variety of labelled attacks and a simulated environment allows for varying simulation parameters and, therefore, it easily tests different aspects of the strategy. However, there are a couple of potential drawbacks.

The first drawback is that the attacks in the datasets may be outdated and may not contain the latest attacks. The second drawback is that the simulated scenarios probably do not consider changes in an attacker's behaviour in response to the actions taken by the strategy. In reality an attacker may adapt to the defence and change their actions in an unexpected way. This could significantly affect the evaluation's results. Using replayed attacks would stop being a sufficient evaluation option. Implementing an attacker's logic in the simulated attacks, based on attack vectors, is very demanding and prone to incompleteness.

In this paper, we focus on the consequences of the second drawback. We investigate whether the evaluation results are significantly affected by the attacker's change in behavior. We show that the evaluation gives different results in a simulated environment compared to on a live network facing real attackers. To formalize the scope of our work, we pose two research questions which we shall answer:

1) What are the differences between defence strategy evaluation in simulated and real environments?
2) Does the attacker change his behaviour based on the defender's actions?

In our experiment, we use a honeypot network to evaluate two response strategies in a live network against real attackers. Then, we use the same setup in a simulated scenario with replayed attacks, which were observed on the honeypot network. We found significant differences between the results in the real and simulated environment.

The benefit of using the honeypot network is that the network is attacked by a wide variety of attackers with no risk of actual damage being done. Therefore, we gain a high number of black hat hackers working towards our goal with minimal effort and resources required on our side. The honeypots also provide very detailed logging, because their purpose is to analyse attackers' actions. Therefore, estimating the impact of the attack on a honeypot is much easier than on a regular host. Moreover, we may be sure the traffic consists solely of attacks, since legitimate users would never access the honeypot. The disadvantage is that attackers sooner or later recognize they are interacting with a honeypot. Thus, we cannot test the response strategy against multi-stage attacks.

This paper is organised into five sections. Related work is surveyed in Section II. We describe the methodology of the experiment and the evaluated strategies in Section III. The results of the evaluation of the strategies are shown in Section IV. The paper is concluded in Section V

II. Related Work

We are unaware of a paper that would focus solely on the topic of strategy evaluation, however, the area is touched upon in mostly every paper on Intrusion Response Systems (IRS). Therefore, we mention the methodology of strategy evaluation in those papers. We took a list of the latest intrusion response system papers from [1] and went through the techniques used for evaluating network based IRSs.

The evaluation techniques can be classified based on the environment used, source of attacks and on evaluated aspects of the strategy, such as performance against the attack (how well does the strategy defend the network), sensitivity (how much are the results dependent on inputs) and scalability (what size networks could the strategy defend).

The most basic evaluation used is the verification of the strategy's decision logic. The evaluation is only theoretical as the strategy is not implemented in any environment. Several attack scenarios are considered based on existing attack vectors. The authors show which actions are chosen by the strategy in each scenario and argue that those actions are the most advantageous in the situation [2], [3], so the only evaluated aspect of the strategy is performance. This type of evaluation is very limited and is not sufficient. It is used also by Kanoun et al. [4], however, the authors are preparing for a more extensive evaluation in a live deployment.

Most of the strategies are evaluated in a simulated environment. In the simulation scenarios, the sources of the attacks are usually either replayed attacks [5], [6], [7] or simulated attacks based on attack vectors [8], [9]. All of the works evaluate only the strategy performance except for Strasburg et al. [6], who evaluate the performance and scalability of the strategy, and Wang et al. [9], who vary simulation parameters to test the performance, sensitivity and scalability. The replayed attacks come from various public DARPA datasets. The simulated attacks, based on attack vectors, cover only a few scenarios.

The most realistic way to evaluate a defence strategy is a real environment [10], [11]. The strategies are tested in a small network setup with detection tools. The attacks against the network are based on attack vectors and are performed in-house. Because of the effort required, only a few different attacks are used. Both works evaluate not only the strategy's performance, but also vary the setup to test other aspects of the strategy.

Apart from intrusion response systems, the evaluation of intrusion detection systems (IDS) is also a much-investigated topic [12], [13]. The evaluation of IDSs is also facing many issues [14], however, those issues are quite different. Intrusion detection evaluation suffers from a lack of labelled, up-to-date datasets, problems with the description and generation of "normal" traffic, and a lack of zero-day attacks data.

TABLE I: Defended honeypots.

IP address	Domain name	Username	Password
xxx.xxx.xxx.2	test	admin	password
xxx.xxx.xxx.3	-	root	raspberry
xxx.xxx.xxx.4	www	root	123123
xxx.xxx.xxx.5	mail	admin	admin
xxx.xxx.xxx.6	db	root	password1

III. Methodology

This section describes the experimental environment, strategies, and evaluation process. First, we describe the network of honeypots used in the experiment and the process of attack detection and response. Then, we present the evaluated strategies, including their theoretical background and parameters given by the experimental environment. Finally, a simulation run and the process of strategy evaluation are described.

A. Experimental Environment

We used five high-interaction honeypots in our experiment, the list of which can be seen in Table I. The honeypots were chosen because they have no production value and a certainty that any incoming traffic is, by nature, suspicious [15]. Thus, we did not have to take false positive detections into consideration and the production environment was spared from changes in configuration.

The honeypots were part of an existing honeynet, which has been running for several years [16]. The historical records of attacks against the honeynet were used in a simulation and cost estimations. The honeypots were placed in the same subnet and were assigned consequent IP addresses. Thus, we were able to observe attackers which targeted several victims at the same time or one by one. The honeypots were also assigned a domain name to lure potential attackers. The role of each honeypot corresponds to its domain name (www, mail, db).

Each honeypot runs a SSH server and responded to authentication attempts. The usernames and passwords were established for each service so that the attackers could successfully guess the combination, see Table I. The probability of a successful attack was based on historical records of authentication attempts against the honeynet.

B. Attack Detection and Response

The honeypots were deployed in a honeynet with central logging mechanisms and a database of authentication attempts. Each honeypot captured the credentials used in an authentication attempt and sent them to the central database. The database record consists of a timestamp, attacker's IP address, honeypot's IP address, username, and password. These records were used as a detection tool for the defence strategy. The attacker is identified by a single IP address and the attack is detected if at least one attempt from the IP is observed in the database.

The network traffic of the honeynet flows through a gate that had the capability to manipulate the traffic, e.g., by blocking access to certain services for certain hosts. Every minute, a script went through the new records in the database

TABLE II: Values of services for attacker and defender.

Target	Value of service		Unavailability cost
	for attacker	for defender	
xxx.xxx.xxx.2	0.03	500	1
xxx.xxx.xxx.3	0.29	500	0
xxx.xxx.xxx.4	0.02	2000	10
xxx.xxx.xxx.5	0.38	3000	2
xxx.xxx.xxx.6	0.28	4000	5

and computed an optimal strategy for the current situation. The experiment had several phases using different strategies which are discussed later. If a strategy resulted in restricting access to the honeypot for the attacker, the honeynet gate was commanded to block the attacker from accessing a specific honeypot. The blocking was implemented using a firewall rule that dropped the traffic only between the attacker and one of the honeypots. Thus, restricting the attacker's access to one of the targets didn't influence the attacker's access to other targets unless the interaction between the attacker and the other target was also found to be malicious. The restrictions could also be lifted in case of the rule being recognized as unneeded at the moment by the strategy.

C. Strategies

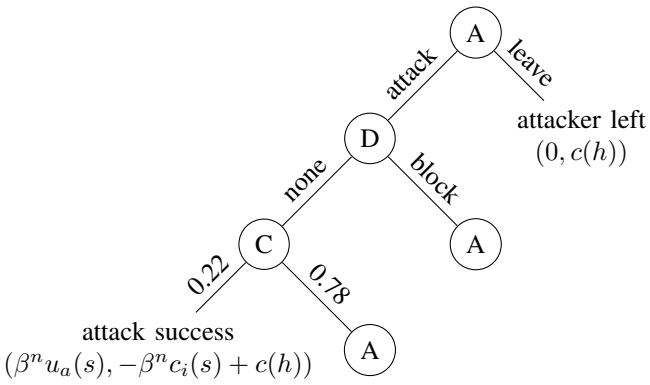
We implemented two different strategies, which were evaluated on the network of honeypots. We chose two concepts which are most common in intrusion response systems - game theory and cost analysis. The requirements placed on the defence are as follows:

- Each service has a value for the defender. The value indicates the damage the defender suffers when the service is successfully attacked. The values of each service are listed in Table II.
- Each service should be available all the time. For each minute the service is not available, the defender suffers a loss. This unavailability penalty is also listed in Table II for each service.
- The frequent reconfiguration of iptables and a high number of iptable rules are undesirable, so for each added/removed iptable rule, the defender receives a reconfiguration penalty. The penalty was set to 10.

The value and the cost of unavailability for each service were established prior to the experiment start. We based these values on the domain names of the honeypots. We treat the honeypot with domain name www as a web server, therefore it has the highest cost of unavailability and a smaller value. On the other hand the honeypot with the domain name db is treated as a database server, therefore it has the highest value and a lower value of unavailability. The honeypot with the domain name mail has a medium value and unavailability cost. The two remaining honeypots have both low value and unavailability cost, the one with domain name test is treated as a test web server, the other as a work station.

1) Game theory: In the first scenario, we used a game-theory based strategy. Game theory takes into account both the attacker's and defender's goals. We used the Nash equilibria to find the optimal defender's strategy. We modelled the interaction between attacker and defender as a finite, non-zero, two player game in an extensive form. Due to the computational complexity, we were forced to model an attack against each service as a separate attack and limit the number of rounds to five rounds, which corresponds to a five-minute interval. Since the upper quartile of the length of attack was less than five minutes, we accepted this limitation.

Fig. 1: One game round.



The n-th round with history h. The A nodes mark the attacker's moves, D node marks the defender's move and C node marks a chance move. The values in brackets denote the attacker's and defender's reward in terminal nodes. $c(h)$ is the accumulated discounted defender's reward based on history h, $p_s = 0.22$ is the probability of attack success, β is the discount factor, $c_i(s)$ is the value of the service for the defender, $u_a(s)$ is the value of the service for the attacker.

The two players (attacker and defender) select an action in turns. One game round is depicted in Figure 1. The actions available to the attacker are to leave the game, gaining nothing, or attack the target. The actions available to defender are to block the attacker from the target by reconfiguring the IP tables or not to block the attacker from the target. If the attacker chose to attack and the defender chose not to block, the chance node is reached. In the chance node the game ends with the probability of $p_s = 0.22$ for a successful attack and continues with the probability of $p = 0.78$. The probability p_s was estimated as a weighted average of probabilities of success for each service, based on the observed frequency of the combination of username and password and average attempts per minute. The reward in the terminal node is discounted by a factor of $\beta = 0.9$ for each round and is computed as follows:

- If the game ends because of a successful attack, the attacker gains the discounted value of the service, based on his utility function. The defender gains the discounted penalty for service lost and the discounted penalty for reconfiguration and unavailability based on the history of the final state
- If the attacker leaves, he gains nothing and the defender gains the discounted penalty for reconfiguration and unavailability based on the history of the final state
- If the game reaches the final state after five rounds,

the attacker gains nothing and the defender gains the discounted penalty for reconfiguration and unavailability based on the history of the final state

The discount factor express that the gain/loss, which is further in future has a lower value than the same gain/loss received sooner. The values of each service for attackers are listed in Table II. The value of a service for the attacker is the reward an attacker receives upon a successful attack against the service. Given that the difficulty of attacking each target is exactly the same and assuming the attackers choose their targets based on the subjective target value, we based the value for the attacker on the popularity of each target before the experiment.

We found the mixed Nash equilibria of the game using the Gambit tool [17] and implemented a strategy based on the equilibria strategy.

2) Cost Sensitive: The second implemented strategy was based on cost evaluation. It considers the immediate cost associated with each action from the defender's point of view and does not include the attacker's goals into consideration. It performs the action with the lowest associated cost.

The defender has the following responses available: to block the attacker from the target or not to block the attacker from the target. Each action, a, is described by two parameters: the type of the action, $type(a)$, which is either to block or not and the target of the action, $target(a)$, which denotes the service the attacker will be blocked from or not. This means a total of 10 actions is considered.

The cost of action a consists of two parts $C(a) = C_n(a) - C_p(a)$. The cost $C_n(a) = C_r(s,a) + C_u(a)$ sums the negative impacts of the action on the system and the cost $C_p(a)$ sums the positive impacts of the action. The negative impacts of the action are the cost of reconfiguration $C_r(a)$ and the cost of unavailability $C_u(a)$. The cost of reconfiguration $C_r(a)$ is computed based on the current configuration and the considered action as follows:

$$C_r(s,a) = \begin{cases} 0 & \text{if the current state is the same as the state of block of action } a \\ 10 & \text{otherwise} \end{cases}$$

The cost of unavailability depends on the target of the action and its associated availability value c_A. It is computed for one round, which corresponds to one minute.

$$C_u(a) = \begin{cases} 0 & \text{if the action is not to block} \\ c_A(target(a)) & \text{otherwise} \end{cases}$$

Finally, the positive impacts of the actions are estimated as the potential damage that was mitigated by the defensive action. The potential damage depends on the probability of an attack's success, $p_s = 0.22$. It was estimated as a weighted average of probability of an attacks's success against each service. This is based on the observed frequency of username/password combinations and average attempts per minute.

$$C_p(a) = p_s \cdot c_i(target(a))$$

This cost estimation follows the cost estimation in [6], which uses the operational cost, impact of the response on the system, and response goodness to calculate the total cost.

Each round, a script is called which considers all actions for all attacks in terms of their cost. The action with the lowest cost is selected and performed.

D. Simulation Run

In order to decide whether there is a difference between how the strategy performs in a semi-real environment of a honeypot network and in a simulated environment with replayed attacks, we created a simulation scenario for each strategy. In the simulation, the attacks observed before the experiment were replayed against the strategy and the strategy's responses were logged. The strategy employed the same logic as it did when deployed on the honeypot network, however, the attacker was deprived of the possibility to react to the responses. To simulate the reconfiguration of iptables, the attempts that would have occurred during active blocking were not considered.

E. Strategy Evaluation

In order to evaluate the strategies, we first separated the observed attempts into attacks. An attack is a sequence of attempts by the same attacker, regardless of the target, which closely followed after one another. We assume that if there are no attempts by the attacker for at least one hour, the attacker has stopped the attack. For gaps lower than one hour, we were looking into the pace of the attack. If the gap between the attempts was uncharacteristically long, we also assumed the attacker stopped their attack. Based on these requirements a gap between the last attempt and the first attempt in the following attack must satisfy one of the following conditions:

1) The gap between the attempt and the next one is longer than one hour
2) The gap between the attempt and the next one is longer than five minutes and is more than five times the average gap between the previous five attempts

For attacks that occurred during the experiment, we added an additional condition that the gap was not caused by a block. We were also checking if the combination of username and password was used by the attacker before (restarted attacks). A graph of attack lengths is shown in Figure 2, illustrating the distribution of attacks before the experiment.

The performance of the strategy was evaluated for each separate attack. We computed three indicators for each attack:

- the number of reconfigurations of iptables, n_r, made in response to the attack
- the total time, $n_u(s)$, the service was blocked (in minutes) for each service s
- whether the service was compromised for each service s

The overall score, $e(a)$, of the strategy per attack, a, was computed as follows:

$$e(a) = n_r \cdot 10 + \sum_{s \in S} n_u(s) \cdot c_A(s) + \sum_{s \in S} n_{succ}(s) \cdot c_I(s),$$

where $c_A(s)$ is the availability value of service s, $c_I(s)$ is the value for the defender of service s and $n_{succ}(s) = 1$ if service s was successfully attacked, $n_{succ}(s) = 0$ otherwise.

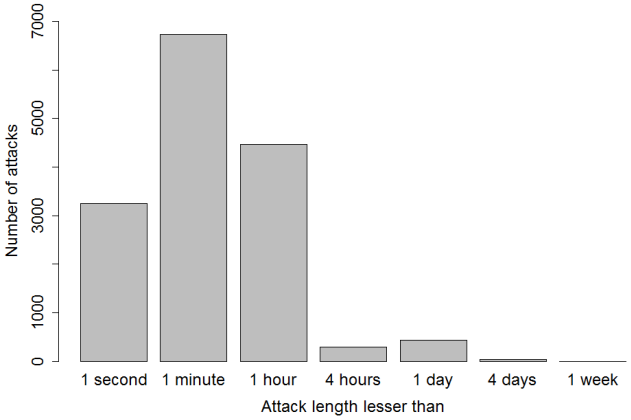


Fig. 2: Attacks lengths

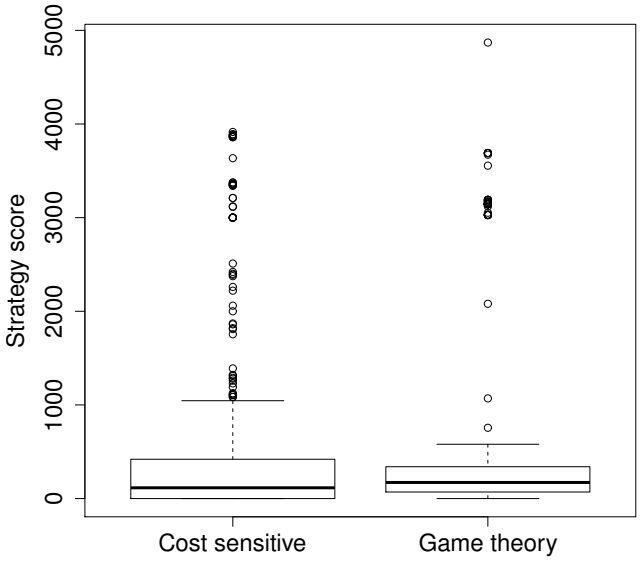


Fig. 3: Comparison of Cost Sensitive and Game Theory Strategy

IV. Results and Discussion

The experiment consisted of two steps, simulation run and live evaluation. We ran the live evaluation in the honeynet in August 2016. Each strategy was deployed for at least two weeks. In Table III, we summarize the overall statistics of the experiment, i.e., the number of attacks, successful attacks, reconfigurations, and time interval under which the machine was unavailable. We observed a total of 207 attacks during the experiment with the game theory based strategy and 437 attacks during the experiment with the cost sensitive strategy.

In the simulation with replayed attacks, we replayed a vast collection of attacks against each strategy. The attacks were reconstructed from the historical records of authentication attempts against the honeynet. An attack was a series of authentication attempts that had the same source and target IP. We used a collection of 2,148,805 authentication attempts in 15,214 attacks observed from December 2011 till June 2016.

A. Strategy Comparison

We evaluated the strategies as described in Section III-E. Figure 3 shows a comparison of the results for both strategies. The overall statistics of the results for each strategy are in summed up in Table IV.

The cost sensitive strategy did better than the game theory based strategy in the semi-real environment of honeypot networks. It had the advantage of longer minimal block duration over the game theory based strategy. The average length of attack observed prior to experiment was around 26 min. The short duration of 5 minutes led to a high number of repeated reconfigurations resulting in a high penalty. This is because in the case of longer attacks, if the block is removed, the attack is detected again and the block has to be put back in place, resulting in two additional reconfigurations. It also prevented a small window of opportunity for attackers to perform a successful attack after the block was removed and before the attack was detected again (and the block was put back in place). As you can observe in Table III the cost sensitive strategy has a lower relative number of successful attacks than the game theory based strategy. The cost sensitive strategy is also easily scalable, which could give it an even greater advantage on larger networks.

B. Simulated and Semi-real Execution

Next, we compared how each strategy performed in the simulated environment and the semi-real environment. The overall statistics for each strategy in the simulation run are presented in Table IV. It shows that both strategies performed differently in the simulated environment than in the semi-real environment of the honeypot network. The mean results of the strategies in the semi-real environment are lower than the mean results of the strategies in the simulated environment, and the standard deviation is also lower in the semi-real environment. Moreover, the change is in the opposite direction. In the experiment, the game theory based strategy performed worse than the cost sensitive strategy. In the simulation run, it was the other way around.

To confirm the difference in the outcomes of the strategies, we want to show that the results in the simulated environment do not come from the same distribution as the results in the semi-real environment. We used the Wilcoxon–Mann–Whitney two-sample rank-sum test [18] to test the following null hypotheses:

1) the distribution of the game theory based strategy's results in the semi-real environment does not significantly differ from the distribution of game theory based strategy's results in the simulated environment
2) the distribution of the cost sensitive strategy's results in the semi-real environment does not significantly differ from the distribution of the cost sensitive strategy's results in the simulated environment

TABLE III: Experiment Statistics

Strategy	# attacks	# reconfigurations	# minutes blocked	# successful attacks	Start	End
Game theory	207	2374	5294	55	2016-08-01 00:14	2016-08-16 03:21
Cost sensitive	437	1029	22467	62	2016-08-16 03:41	2016-08-30 17:44

TABLE IV: Strategy Results per Attack Statistics

Strategy	Count	Mean strategy score	Stdev
Game-theory	207	803.37	1279.24
Cost sensitive	437	489.41	937.52
None	15214	1004.47	2407.20
Game-theory*	15214	1006.363	2371.38
Cost sensitive*	15214	1109.63	2343.42

* simulation run

We were able to reject both null hypotheses with a significance level of $\alpha = 0.01$, which indicates that for both strategies there is a statistically significant difference between how the strategy performs in a simulated environment and how the strategy performs in a honeypot network.

We tried to explain why the strategies would perform better facing real attackers than replayed attacks. We suspect the difference is tied to the difference in attack lengths as discussed in detail in Section IV-C1. The defender's actions probably forced the attacker to split his attacks over time and the result is several attacks of medium length instead of one long attack. This leads to better result in our evaluation scheme, however, it might be argued, whether this is the desired result of employing a network defence.

C. Attacker's Behaviour

We tried to investigate whether the attacker's behaviour changes based on the defender's actions. We found several statistics from the data gathered during the experiment: the lengths of the attacks, correlation between attack lengths and strategy evaluation scores, ratio of returning attackers and time needed to succeed. Then we compared these statistics between the semi-real environment runs when attacker can change his behaviour based on the defender's response and the simulated environment run, when this option is taken from him.

1) Attack Lengths: We compared the lengths of the attacks before the experiment and throughout the experiment. The estimated density of the lengths of the attack for both strategies and historical records is shown in Figure 4. We can see that the lengths are lower in the historical records than in the semi-real environment. We suspect the longer attacks could be caused by an attacker waiting for the defender to stop blocking him to finish the attack. The relatively high values for the cost-sensitive strategy were caused by several very slow attacks that occurred during the experiment. Although, the attacker made only several attempts, the time span between the attempts was almost one hour.

To show that there is a significant difference between the lengths of attacks in a semi-real and simulated environment, we used the Mann-Whitney-Wilcox test with following null hypotheses:

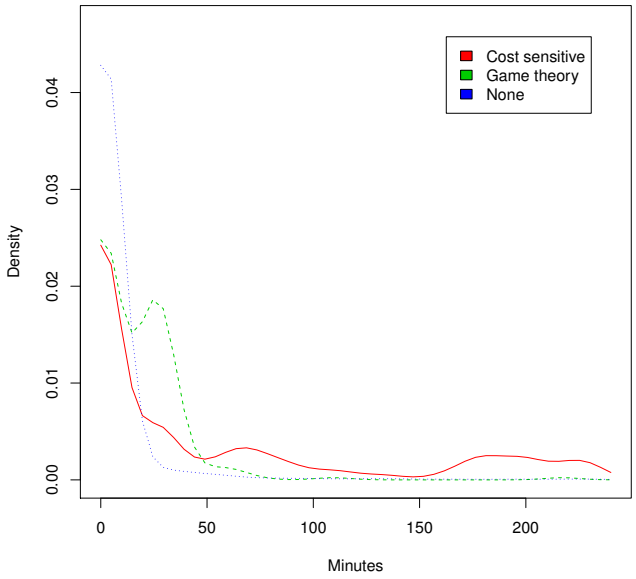


Fig. 4: Attacks lengths

1) the distribution of attack lengths observed during experiment with a game theory based strategy in a semi-real environment does not significantly differ from the distribution of attack lengths prior to the experiment
2) the distribution of attack lengths observed during experiment with a cost sensitive strategy in a semi-real environment does not significantly differ from the distribution of attack lengths prior to the experiment

We were able to reject both hypotheses with a significance level of $\alpha = 0.01$. This confirms that the attackers were indeed behaving differently during the experiment (when they were actively opposed) than prior to the experiment (when they did not meet any resistance).

2) Correlation between attack length and strategy result: We were interested in how much the overall penalty depends on the length of the attack. Since the defender receives the penalty for unavailability, we would expect these two numbers to be correlated. The correlation coefficients are shown in Table V. There is almost no correlation between the attack length and the strategy result when deployed in the semi-real environment for the game theory based strategy or the cost sensitive strategy, and in the results of no strategy. A slightly higher correlation was observed in the simulated environment for the cost sensitive strategy and game theory strategy. This means that the correlation differs for data captured during the experiment when the attacker can react to the defender's actions (since no strategy is how the honeypot network was defended up to now) and the data from the simulated environ-

ment with replayed attacks.

TABLE V: Correlation between attack length and strategy result

Strategy	Correlation	95% Confidence interval
Game-theory	0.11	[-0.02, 0.25]
Cost sensitive	0.06	[-0.03, 0.15]
None	0.05	[0.03, 0.07]
Game-theory*	0.35	[0.33, 0.36]
Cost sensitive*	0.41	[0.39, 0.42]

* simulation run

The overall low correlation is caused by the minimal block duration for each strategy. The low correlation between penalty and attack length for no strategy indicates that the attack's success is dependent mostly on whether the attacker has the right combination of username and password in his dictionary, and not on the length of the attack.

3) The Return of the Attacker: Another interesting aspect of the attacker's behaviour is the tendency to repeat the attack. For both strategies, we computed the average ratio of attackers who repeated their attack in a period of the next 10 days. We compared the results with the same values computed for attacks observed before the experiment. A boxplot with results is shown in Figure 5. The average ratio of returning attackers is 0.24 during the experiment with the game theory based strategy, 0.26 during the experiment with the cost sensitive strategy and 0.11 in the historical data.

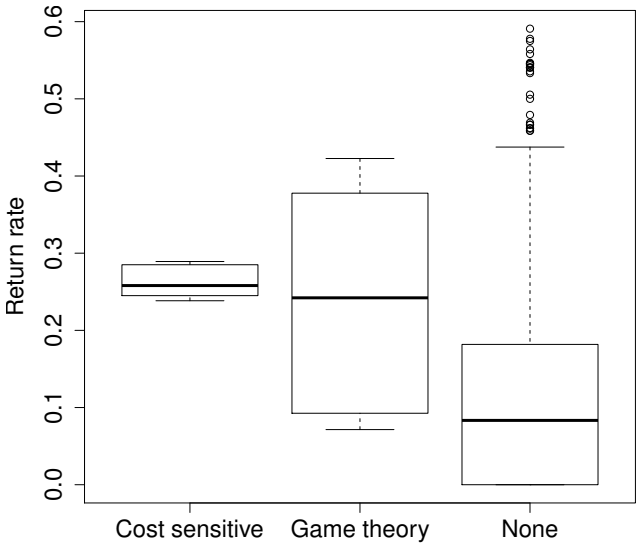


Fig. 5: Attacker's Return Ratio

We can see that the return rate of the attacker is higher when the attacker has a chance to react to the defender's actions. One explanation might be that the attacker gives up after being blocked, however, returns later to finish the attack. We also investigated whether the returning attackers were starting from where they left off (meaning the combinations of usernames and passwords were not observed in the previous attack by the same attacker). We saw only a very low number of restarted attacks, which supports the hypothesis that the attackers paused their attacks when blocked and continued the attack later.

4) Time to Succeed: We investigated how long the attacker needs to successfully attack the service. We took all the successful attempts and computed the time from the beginning of the attack. The statistics of the measured times are shown in seconds in Table VI. The average values are rather small due to weak passwords. The highest average value was computed for no strategy, which is not very surprising since this is the only strategy that allows the attacker to succeed over the length of the whole attack. Overall these statistics do not point to any change of attacker behaviour.

TABLE VI: Time to succeed

Strategy	Mean time to succeed	Stdev
Game-theory	238.80 s	546.39 s
Cost sensitive	855.03 s	2068.91 s
None	1055.36 s	7561.65 s
Game-theory*	431.64 s	3938.81 s
Cost sensitive*	300.08 s	7561.65 s

* simulation run

D. General Discussion

In this section, we would like to summarize comments and lessons learned during the experiment's setup and run. The following comments might not relate directly to the research questions, however, we find them relevant to the subject of strategy verification.

The data shows that the attacker does indeed change his behaviour if he faces an active defence. We found out that the active defence does not deter the attacker. On the contrary, the attackers were more persistent in their attacks during the experiment and often returned to continue the attack. We have two most probable hypotheses for the attacker's motivation for this change of behaviour. Either the act of defending suggests that there is something worth defending, which increases the attacker's motivation, or we were dealing with automated attacks scripted so that the given sequence of username/password combinations is always finished and the observed change in behaviour was caused by retry mechanisms in the script. Since we cannot confirm or reject our hypothesis with the attackers, we cannot find the true reasoning behind their actions.

The computational complexity of the strategies is often not reflected in the evaluation. We encountered this limitation with the game theory based strategy. If the strategy cannot be scaled with reasonable computational complexity, it might perform very well, however, it could not be used in real deployment since networks tend to be of a larger scale than in experiments. Although game theory has performed well, it could not (in its current form) be used in practice. However, this attribute could be detected during the simulation verification since we can easily scale our network in simulated environment.

The combination of one-time costs and cost per unit of time has to be very carefully considered. Specifically, in the security triad Confidence, Integrity, Availability, the former two are one-time costs and the latter is a cost per unit of time. Since we were running an experiment, we defined these values without the need to justify them. However, in deployment in a real network, the system administrator would be hard pressed to estimate these parameters.

Deployment in a real environment forces the authors to address all aspects of the strategy, such as an estimation of parameters, what inputs are available, whether the assumptions about other components are realistic, and so on. The strategy is viable only if all parameters can be estimated, inputs supplemented and all assumptions met. In our case, we needed to estimate the probability of an attack's success, the values of the services and we also had to take into account that, by blocking, we lose detection ability.

V. Conclusions

In this paper, we have evaluated two network defence strategies in a simulation run and on a live honeypot network to compare evaluation in a real-life and simulated environment.

Our first research question regarded the difference between defence strategy evaluation in simulated and real environments. When we compared the results of the simulation run and the live evaluation, we could see that both strategies were performing better in the live evaluation than in the simulation run. We attribute this difference to the change in attackers' behaviour. This suggests there are significant differences between theoretical and practical evaluation that have to be taken into account if we want to evaluate new defence strategies.

To answer our second research question concerning the change in attackers' behaviour when facing an active defence, we studied various statistics from the data gathered during the experiment and before the experiment. We found that the attacks were on average longer during the experiment and that a higher number of attackers repeated their attack. We concluded that an active defence does not deter the attacker. On the contrary, the attackers are more persistent in their attacks and often return to continue the attack.

We observed notable differences between the evaluation in a simulated environment and on a honeypot network. We conclude that using replayed attacks for evaluating strategy is not sufficient. Since implementing an attacker's behavior in the simulation scenarios is going to be a very demanding task, it is clear that the field of strategy evaluation faces a considerable challenge.

In our future work, we are going to focus more deeply on the attacker's reaction to a defence. We would like to expand our experiment so that the attacker can observe and react to the success of his attack. However, for this, we would need a high-interaction honeypot that allows successful intrusion. It would also be very useful to find ways how to distinguish between scripted attacks and human-driven attacks. Since scripted attacks follow a very simple logic, the defence could be tailored to counter this logic, and therefore, could be much more efficient.

Acknowledgements

This research was supported by the Security Research Programme of the Czech Republic 2015 - 2020 (BV III / 1 VS) granted by the Ministry of the Interior of the Czech Republic under No. VI20172020070 Research of Tools for Cyber Situational Awareness and Decision Support of CSIRT Teams in Protection of Critical Infrastructure.

References

[1] A. Shameli-sendi, M. Cheriet, and A. Hamou-lhadj, "Taxonomy of intrusion risk assessment and response system," *Computers & Security*, vol. 45, pp. 1–16, 2014.

[2] N. Kheir, "Response policies and countermeasures: Management of service dependencies and intrusion and reaction impacts," Ph.D. dissertation, PhD thesis, Ecole Nationale Supérieure des Télécommunications de Bretagne, 2010.

[3] M. Papadaki and S. Furnell, "Achieving automated intrusion response: a prototype implementation," *Information management & computer security*, vol. 14, no. 3, pp. 235–251, 2006.

[4] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens, and J. Araujo, "Automated reaction based on risk analysis and attackers skills in intrusion detection systems," in *2008 Third International Conference on Risks and Security of Internet and Systems*. IEEE, 2008, pp. 117–124.

[5] Z. Zhang, P.-H. Ho, and L. He, "Measuring ids-estimated attack impacts for rational incident response: A decision theoretic approach," *Computers & Security*, vol. 28, no. 7, pp. 605–614, 2009.

[6] C. Strasburg, N. Stakhanova, S. Basu, and J. S. Wong, "A framework for cost sensitive assessment of intrusion response selection," in *2009 33rd Annual IEEE international computer software and applications conference*, vol. 1. IEEE, 2009, pp. 355–360.

[7] N. Stakhanova, S. Basu, and J. Wong, "A cost-sensitive model for preemptive intrusion response systems." in *AINA*, vol. 7, 2007, pp. 428–435.

[8] W. Kanoun, N. Cuppens-Boulahia, F. Cuppens, and S. Dubus, "Risk-aware framework for activating and deactivating policy-based response," in *Network and System Security (NSS), 2010 4th International Conference on*. IEEE, 2010, pp. 207–215.

[9] S. Wang, Z. Zhang, and Y. Kadobayashi, "Exploring attack graph for cost-benefit security hardening: A probabilistic approach," *Computers & security*, vol. 32, pp. 158–169, 2013.

[10] C. Mu and Y. Li, "An intrusion response decision-making model based on hierarchical task network planning," *Expert systems with applications*, vol. 37, no. 3, pp. 2465–2472, 2010.

[11] B. Foo, Y.-S. Wu, Y.-C. Mao, S. Bagchi, and E. Spafford, "Adepts: adaptive intrusion response using attack graphs in an e-commerce environment," in *2005 International Conference on Dependable Systems and Networks (DSN'05)*. IEEE, 2005, pp. 508–517.

[12] N. J. Puketza, K. Zhang, M. Chung, B. Mukherjee, and R. A. Olsson, "A methodology for testing intrusion detection systems," *IEEE Transactions on Software Engineering*, vol. 22, no. 10, pp. 719–729, 1996.

[13] N. Athanasiades, R. Abler, J. Levine, H. Owen, and G. Riley, "Intrusion detection testing and benchmarking methodologies," in *Information Assurance, 2003. IWIAS 2003. Proceedings. First IEEE International Workshop on*. IEEE, 2003, pp. 63–72.

[14] P. Mell, V. Hu, R. Lippmann, J. Haines, and M. Zissman, "An overview of issues in testing intrusion detection systems," 2003.

[15] The Honeynet Project, *Know Your Enemy: Learning about Security Threats*, 2nd ed. Boston: Addison-Wesley, 2004.

[16] M. Husák and M. Drašar, "Flow-based Monitoring of Honeypots," in *Security and Protection of Information 2013*. Brno: Univerzita obrany, 2013, pp. 63–70.

[17] R. D. McKelvey, A. M. McLennan, and T. L. Turocy, "Gambit: Software tools for game theory, version 14.1. 0," 2014. [Online]. Available: http://www.gambit-project.org/

[18] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *The annals of mathematical statistics*, pp. 50–60, 1947.