INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES LYON

UNIVERSITY OF PASSAU

Game-theoretic optimizations for honeypot deployment in AMI networks.

William Occelli

Supervised by:
Prof. Dr. Lionel Brunie
Prof. Dr. Harald Kosch

Project submitted in partial fulfillment of the requirements for the Degree of:
Master of Engineering,
Information and communication systems
at:
Institut National des Sciences Appliquées de Lyon,
and Universität Passau

September 2020

Abstract

The Smart Grid is an advanced energy network providing a two-way flow of information for power control and consumption data. Smart Meters, Intelligent Electronic Devices (IEDs) and Advanced Metering Infrastructures (AMI) are new assets of the power grid that require to be safely integrated. Their integration leads to new cybersecurity challenges which stimulate research. One of the available solution to prevent potential attackers to penetrate the power network is the use of deception techniques such as the deployment of honeypots, acting as dummies to lure attackers away from the critical servers. In this document, we use game theory to evaluate the best strategies in the deployment of honeypots in an AMI network. We propose a simplified non-cooperative game model with incomplete information between an attacker and a defender. From this game model, we derive optimal strategies thanks to the study of the Nash equilibria. We also propose detailed simulations to find the optimal honeypot proportion to deploy in a power network. Starting from our simplified game model, we introduce resource constraints for the defender and we extract the best theoretical defense configuration. Since our game model implies rationality of the players, we evaluate the optimal configurations found under an evolutionary based game model with irrational players. Results on the theoretical optimal honeypot proportion give valuable information to potential network administrators. However, important limitations are observed when analyzing the evolutionary-based simulations.

Acknowledgements

I would like to express my gratitude to my supervisor, Lionel Brunie, for his support, remarks and open-mindedness through the entire process of this master thesis. His positivism and love for academic research helped me to face any obstacle during these two past years. Furthermore I would like to thank Harald Kosch and Axelle Cheney for their support on the way. They always were available to kindly answer my questions and accompany me in the beautiful city of Passau. Also, I would like to thank Gabriele Gianini for his precious comments, wise advice and admirable knowledge on game theory. Without his engagement in my work, this document could not have been written. At last, I would like to thank my loved ones, who have supported me throughout entire process, especially during the difficult period of the COVID-19 pandemia where all my plans were questioned. Their support and patience allowed my to adapt quickly and finally propose this last work of my master's studies.

Contents

1 Introduction

Introduced by John Nash in 1951 [1], non-cooperative games study the strategies and payoffs between two or more players in competition. Particularly adapted to cyber-security contexts, game theory is used to help decision makers adopt the optimal strategies for their defense against cyber-attacks. This analytical approach is notably interesting in a recent research field which studies the use of honeypots as defense mechanisms in energy distribution networks called Smart Grids. Being relatively undocumented, game theory applied to the deployment of honeypots in Advanced Metering Infrastructure (AMI) networks allows us to propose a novel and more comprehensive non-cooperative game model. This game model is designed to study the best strategies for defenders (network administrators) in the deployment of honeypots as dummies for potential attackers. Our goal is to provide useful information about the optimal proportion of honeypots to deploy in an AMI network along with the efficiency evaluation of this setup. Thanks to game theory, we want to offer an AMI network administrator the opportunity to calculate the number of honeypots to deploy in order to obtain the best chances of luring and deceiving potential attackers. The simplified game model presented in this document will be evaluated under cost and efficiency constraints, meaning that we will propose simulations in order to assess a theoretically optimized honeypot proportion in a network when the resources of the network administrator are limited and when the number of honeypots influences the lure efficiency of these latter. Since we are using non-cooperative game models and Nash equilibria, we assume that players are rational and informed about heir opponents payoffs. The rationality of players is a strong assumption that doesn't reflect real-life cases. Hence, we will propose an evaluation of the optimal honeypot proportion found thanks to our static game model, with the use of evolutionary based simulations where players are irrational. Several strategies, represented by populations of players, will encounter and confront themselves over several generations. We will observe the evolution of the populations so as to better understand the relationships between them and assess the resilience of the population representing our optimized honeypot proportion against other various populations. Discussions and conclusions will be drawn from these results.

This document is organized as follows: we will first present the context of our work, our motivations and an overview of the the parties involved in our work. Then, we will detail the existing work on game models applied to honeypots in Advanced Metering Infrastructure (AMI) networks. In part 4.1, we will propose a formal description of our considered game model. In part 4.2, we will compute our game model's equilibria. In part 4.3 we will use simulations to better understand these equilibria and we will improve these simulations in part 5 where we will assess cost and efficiency constraints. Part 6 presents the evolutionary based evaluation of our results along with the new implemented model. Finally, we will evaluate and conclude on the results obtained, in part 7.

2 Context

2.1 Motivations

Energy is the most valuable resource on our planet. Electricity is at the basis of the most complex systems and it is now essential to our lives. However, most of the electricity distribution networks, called grids, have changed very little over the years. We entirely rely on networks that are vulnerable, since they were not originally designed to withstand attacks, and since they are incapable of resilience. That's why the energy sector is slowly evolving towards a more distributed network that will allow two-way flow of information in order to perform context-aware power control.

The concept of Smart Grid designates the use of technology to achieve a predictive, adaptive, sustainable, self-healing and resilient power network. To reach these requirements, Smart Grids must deeply modify the power network topology from a centralized system, simply distributing to customers, to an interconnected and decentralized system. This change implies the integration of new constituents that are able to communicate with other devices, through various communication technologies. However, the apparition of new components, produced by many different stakeholders, leads to interoperability and security issues. The connected devices do not usually implement sufficient security, often constituting openings into the network for malicious users.

The paramount importance of power systems in a society or a country, makes the power grid particularly interesting for ill-intentioned people. The increasing number of cyber-attacks on power systems leads institutions to focus their efforts on cyber-security. The new challenges brought by the integration of connected components into the grid stimulates the researches on the topic.

Out of the various techniques used to protect the electrical grid, deception techniques are still recent solutions. Deception techniques are non-destructive and non-intrusive techniques used to lure potential attackers towards dummy and uncritical systems. Deception can be combined with Intrusion Detection Systems to provide additional information on potential attacks. This combination of detection and deception is especially done thanks to the use of honeypots in the networks. Honeypots are security mechanisms, disguised as legitimate vulnerable assets of the system, which are configured to send appropriate reports to log collectors in order to detect and analyze attacks. The implementation of honeypots in Smart Grid systems is rare and the researches are recent whereas it allows to both deceive and discover attacks. Hence, the motivation to develop deception techniques applied to the Smart-Grid's cyber-defense and most notably, to local metering networks which represent easy entry points for potential attackers.

2.2 Advanced Metering Infrastructures

The Smart Grid possesses a huge and complex infrastructure composed of thousands of subnetworks. Some of them, such as the inner networks of power stations are usually well protected against cyber-attacks. However, it is impossible to deploy those security measures to a large scale and public network where millions of new electronic devices are introduced each year. Hence, the Advanced Metering Infrastructure becomes the most critical infrastructure to protect with the development of the Smart Grid. Advanced Metering Infrastructure or AMI is a configured infrastructure that integrates a large number of technologies [2]. Used to support safe, efficient and reliable electricity distribution along with advanced functionalities offered to the customer [3], the AMI comprises Smart Meters in the Home Area Network (HAN), Data Aggregation Points (DAP) in the Neighborhood Area Network (NAN) and Meter Data Management Systems (MDMS) in wider networks

(extended NAN or WAN). The Smart Meters communicate with the Data Aggregation Points (DAP) which collect data that will be sent via black-haul communication to the Meter Data Management System. Implementing AMI is a basic and mandatory step towards modernization of the electrical grid [4]. AMI provides customer with detailed, near real-time, information about power usage, cost and billing. AMI also supports net-metering for the integration of DERs in the network. There exists many implementations of AMI such as infrastructure with mesh topology and multi-point topology [3]. Mesh topology allows Smart Meters to communicate with each other in order to exchange data; only a subset of Smart Meters communicate with the DAP. Multi-point topology consists of a DAP communicating directly with each isolated Smart Meter. The communication set up in AMI varies depending on the characteristics of the Home Area Network (HAN). The technologies commonly used are :

- Cellular
- Power Line Carrier (PLC)
- Radio frequencies (GPRS)
- Zigbee
- Broadband over Power Lines (BPL)
- Bluetooth
- Optical Fiber
- WiFi
- WiMax

Since the data is not necessarily time-critical and since the devices in the HAN are easily movable and separated by short distances, wireless communication technologies are the most used : WiFi, ZigBee, HomePlug [2]. Technologies such as ZigBee and Homeplug have low energy consumption but low bandwidth. Wired technologies are used for more performance-related communications.

Smart meters Smart Meters are the end points of the Smart Grid. Located in a Home Area Network (HAN) they constitute the first brick of the system. Smart meters differ in their functions but they all possess common requirements and features [2, 3]:

- Quantitative measurement : Smart Meters must accurately measure the quantity of medium using different physical principles, typologies and methods. Producing data about temperature, electricity consumption, periods of consumption and other various measures, Smart meters will allow efficient monitoring and operations.
- Control and calibration : Smart Meters need to compensate small variations in the system to avoid outages or unnecessary energy consumption. Load limiting, tamper and energy theft detection are part of these actions.
- Communication : unlike Automatic Meter Reading (AMR), AMI provides a bidirectional flow of information. Smart meters must communicate with other devices to send metering data, but they also need to receive operational commands and act accordingly. They must be able to receive upgrades of firmware to prevent security breaches.

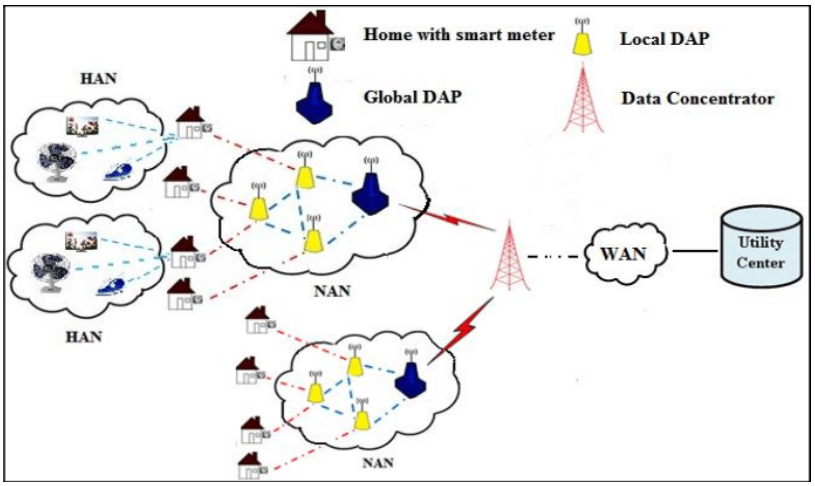

Figure 1: Smart Grid AMI network [5]

- Power management : Smart Meters connected to the electrical grid must be able to detect loss of power, power outages and power over-consumption. They allow power quality monitoring and some of them must maintain their main functionalities if no current is received.
- Display : Customers are provided with data visualization and information to match energy costs and billing.
- Synchronization : Smart Meters need to perform time synchronization with Data Aggregation Points (DAP) [6] and other devices to ensure a reliable data transmission especially for wireless communication.

Data Aggregation Points Data Aggregation Points (DAP) are communication nodes that bridge Smart Meters and the Meter Data Management System (MDMS). DAP are usually installed in electrical substations, inside buildings or in pole-mounted containers [3]. Smart Meters and DAP are connected through the Neighborhood Area Network (NAN). DAP is a gateway between the NAN and the power grid data network. Because DAPs are single points of failure by collecting all data from Smart Meters located in the NAN, it is recommended that Smart Meters have the possibility to access several DAP as explained in [6].

Meter Data Management Systems The Meter Data Management System (MDMS) is the central module of the operations management system provided with the analytical tools required for intelligent communication and decision making [2]. MDMS implementation differs depending on the utility provider but it usually features data import, data validation, data cleansing and process in order to prepare it for billing and analysis. MDMS will provide application programming interfaces (APIs) to the applications depending on the metering data (billing applications, power management systems, load balancing applications). MDMS also have the ability to verify power status, to perform power restoration verification, to remotely connect to Smart Meters and to read remote meters on demand.

2.3 Cyber deception in the Smart Grid

To reach common security objectives, organizations have worked to define high-level security requirements to the participants of the grid. The NIST-IR [7], NERC-CIP, IEC (IEC-61850) [8], CEN-CENELEC-ETSI [9] and other organizations have agreed to provide standards based on three main requirements: *availability, integrity* and *confidentiality*. To give a more complete approach of the high-level requirements, *authentication, authorization, auditability, accountability* and *non-repudiability* can be addressed [10, 11].

To fulfill these security requirements and challenges, many security countermeasures have been studied over the years. Models have been created to organize and separate the different methods used in cyber-security. Namely the "7D model", defined in [12] and detailed in [13], has been proposed to better understand the scope of each cyber-security countermeasure. The 7D security model is composed of seven actions :

- Discover
- Detect
- Deny
- Disrupt
- Deceive
- Degrade
- Destruct

Discovery, detection and denial belong to the pre-attack phase defined in [11] whereas disruption, deception, degradation and destruction are deployed when the system is infected, meaning, during the attack. Discovery is one of the most discussed research field for Smart Grids. The goal of discovery is to identify the vulnerabilities of the system by locating sensitive data or testing communication protocols for example. The knowledge gained from discovery allows to better design security solutions and to ensure that they respond to the evolving needs of the grid through audit. Detection corresponds to the uncovering of cyber-threats thanks to data gathered by the nodes of the grid. Intrusion Detection Systems (IDS) represent unavoidable tools for detection. Denial of attacks is the prevention of cyber-threats. It mainly consists of securing data, denying access to unauthorized users and making sure that only authorized users can access secured data. Disruption is the mitigation of attacks and is typically addressed by game-theory approaches. Deception is a fairly under-reported research field which goal is to mislead the attackers into a trap. The attackers will gather false information that will orient their attack into known (and less critical) areas of the system. Degradation and destruction are the two last phases of the security and consist of destroying the effects of the attack. Their irremediable aspect and the resources needed to perform hostile actions against the attackers (who are usually unknown at the time of the attack) definitely reduce the possibilities for degradation and destruction. In this document, we will focus on the deception part of the cyber-security defense. It is important to note that every action of the "7D model" is complementary and the best defense will be achieved with a wide range of techniques rather than one specific and over-tuned technique.

Deception techniques, since they almost entirely rely on the behavior of the adversary, are usually studied thanks to game theory. Researchers will try to analyze and anticipate the participants' actions in order to propose optimal strategies. To do so, equations derived from the game models' equilibria will be developed to build algorithms able to determine the best strategy for a participant, depending on initial parameters. In the next section, we will detail the few papers applying game theory to the deployment of honeypots in an AMI network.

3 Related Work

Presented as a whole by the literature survey proposed by Dalamagkas et al. [14], the use of honeypots in Smart Grid networks is the subject matter of several recent publications. The emerging need for a better understanding of the available strategies for network administrators is directly linked to the increasing popularity of honeypots and deception techniques as cyber-defense mechanisms. The different equations extracted from the analysis of equilibria in the various game models are used to build simulations that analyse the impact of some parameters (number of honeypots, efficacy of the honeypots) on the different players' payoffs. Related-works propose simulations: the energy consumption and detection rate of attackers as a function of honeypot/anti-honeypot distribution [15], players' payoff as a function of the maintenance costs of honeypots [16] and player's utility as a function of the attacker's detection probability, the number of honeypots in the network and the attacker's belief of facing a real system [17]. However, to the best of our knowledge, no existing work proposes a detailed analysis of the honeypot distribution under cost and efficiency constraints with an evaluation of the results under an iterative version of the game model.

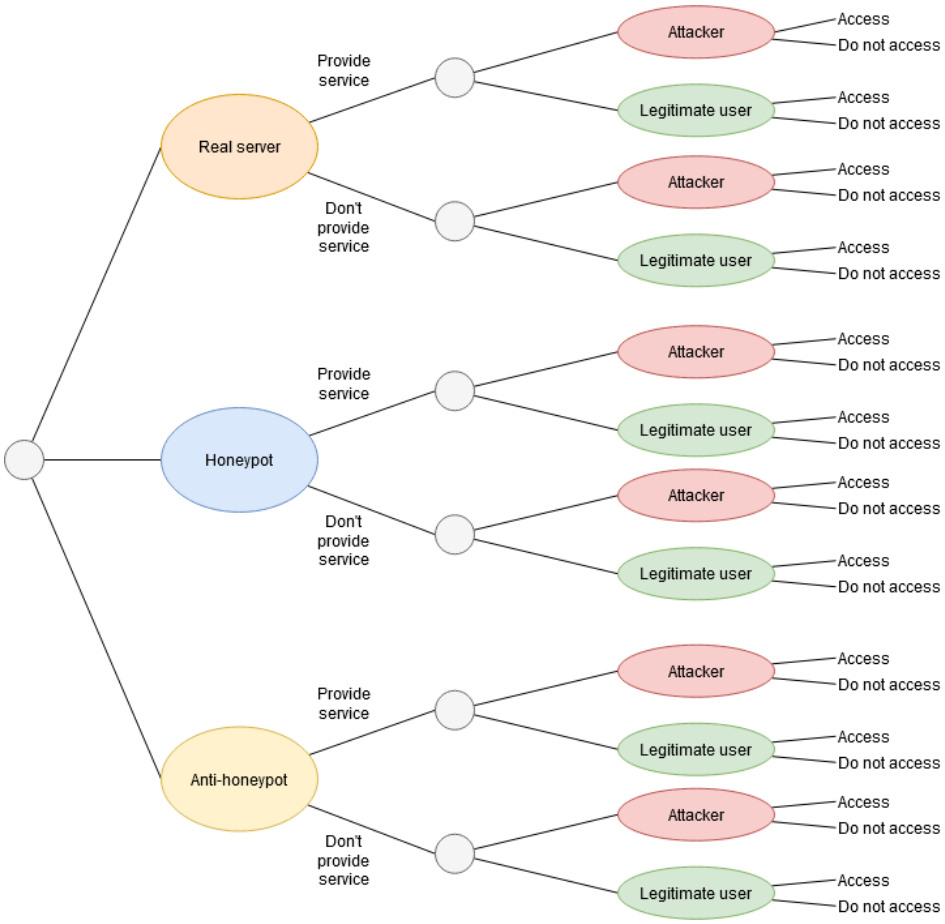

Figure 2: Simplified game tree from the attacker's perspective in Wang et al. [15]

Wang et al. propose a strategic Honeypot Game Model for Distributed Denial of Service (DDoS) Attacks in the Smart Grid [15]. In their Game Model, visitors, which can be legitimate users or attackers, can face 3 different services : honeypot, real servers and anti-honeypot. The visitor can chose to access (or not) the service without knowing which type of service he's facing. Simultaneously, the Service Provider can decide to provide (or not) the service to the visitor, without knowing which type of visitor it is facing (legitimate user or attacker). By analyzing the Bayesian-Nash Equilibria (BNE) they define optimal strategies for balancing the detection rate of attackers and the energy consumption of the deployment of honeypots. They evaluate their proposal thanks to an AMI testbed, plotting detection rates and energy consumption for various strategies.

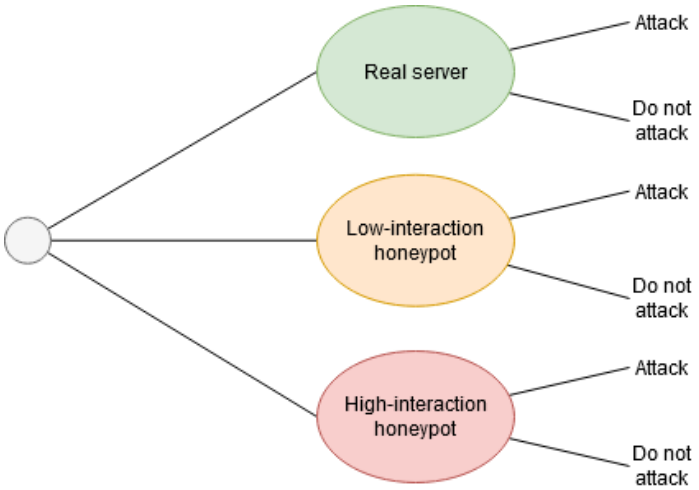

Figure 3: Simplified game tree from the attacker's perspective in Boumkheld et al. [17]

In the scope of the H2020 SPEAR project, Boumkheld et al. propose a Honeypot Type Selection Game for Smart Grid AMI Networks [17]. This sequential game with complete but imperfect information called the Honeypot Type Selection Game (HTSG) allows AMI network administrators to chose the best type of honeypot solution between low-interaction honeypots and high-interaction honeypots. In their Game Model, while deploying a new system, the defender can choose to deploy a high-interaction honeypot, a low-interaction honeypot or a system with no honeypot (real system). Attackers who face these three services can choose to attack or not to attack. The proposed work takes into account the cost of deployment and maintenance of the different types of honeypots compared with the costs of a successful attack. The Bayesian-Nash Equilibria (BNE) are simulated to study the attacker's detection probability and the optimal proportion of the different types of honeypots in the network.

Tian et al. [16] also propose a Honeypot Type Selection Game (HTSG). They go further with the analysis of insufficient defense resources that forces defenders to favour some honeypots over others. In their Game Model, the attacker faces two services : low-interaction honeypots and high-interaction honeypots. As in [15], the service provider can decide to provide (or not) the services to the user. The attacker can choose between a weak offensive attack and a strong offensive attack. For each attack, the attacker chooses to launch (or not) the attack. Bayesian-Nash Equilibria (BNE) are simulated to provide evaluation on the defender's payoffs and the attacker's detection probability.

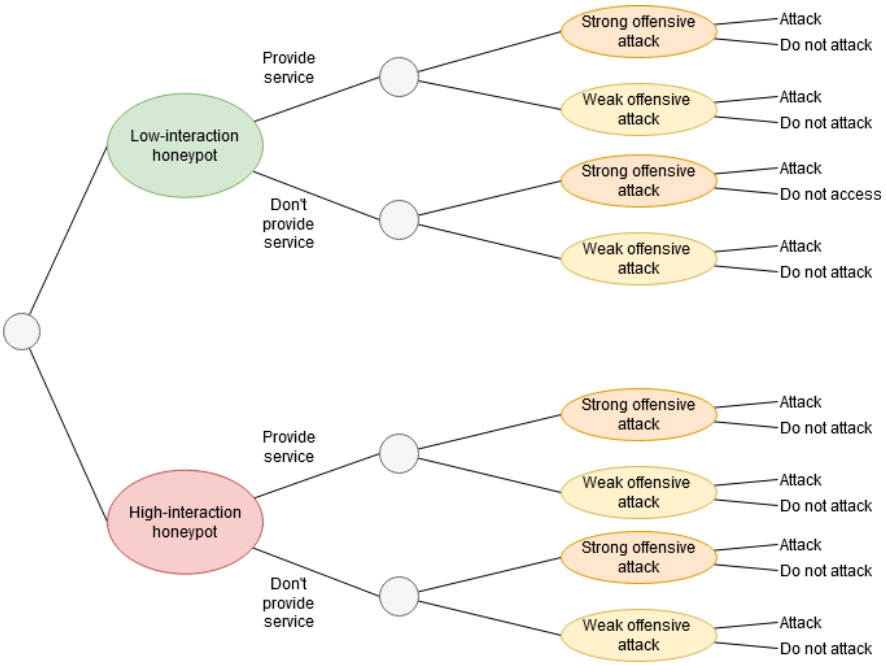

Figure 4: Simplified game tree from the attacker's perspective in Tian et al. [16]

Publication	Attack strategies	Network constituents	Simulations
Wang et al. [15]	Attack No attack	Real server Honeypot Anti-honeypot	Energy consumption and detection rate vs. distribution of HP
Boumkheld et al. [17]	Attack No attack	Real server Low-interaction HP High-interaction HP	player's utility vs. attacker's detection probability, number of honeypots, attacker's belief
Tian et al. [16]	Strong attack Weak attack No attack	Low-interaction HP High interaction HP	players' payoff vs. maintenance costs of honeypots

Table 1: Summary of the related publications on Game Theory models for honeypot deployment in AMI networks

4 Simplified Game model

4.1 Game model

Our simplified game model is formally described in this section.

4.1.1 Context

The proposed game is a non-cooperative 2 players game with incomplete information. An attacker (hacker) is facing a defender (Service Provider, network administrator) in a network composed of two types of devices: Real Servers (RS) and Honeypots (HP). Real Servers (RS) are considered as the substation's (DAP, MDMS) servers in an AMI network. The defender needs to ensure that the RS are providing services in order to have a functioning network. Stops in the service production of RS can be triggered by the defender shutting services down or by the attacker launching an attack on RS. Honeypots (HP) are placed in the network to lure attackers. HP have a cost of deployment and maintenance for the defender but they allow to reduce the impact of an attack performed by the attacker since their attack doesn't impact final legitimate users. HP can provide useful information about the attacker, as an Intrusion Detection System (IDS) would do. This detection feature will be studied in more advanced versions of our game model. The attacker's goal is to maximize his payoff by attacking Real Servers (RS) and trying to avoid attacks on Honeypots (HP). The defender's goal is to maximize his payoff by providing services with Real Servers (RS) and by limiting the impact of a potential attack thanks to Honeypots (HP). The attacker has no information about the type of service he's facing. The defender has no prior information about the attacker's choice. However, each player knows his adversary's available strategies. This simplified toy model is used as a core work for future and more advanced versions of the game.

4.1.2 Game definition

We define the game model as a tuple: $G \triangleq \{D, A, F_A, F_D, U_D, U_A\}$.

$D \in \{D_H, D_R\}$ is the type of network device deployed by the defender D, with D_H a Honeypot (HP) and D_R a Real Server (RS).

A is the Attacker.

$F_D \in \{\Omega_1, \Omega_2\}$ is a binary strategy used by the defender D, where Ω_1 indicates that D provides services and Ω_2 indicates that services are not provided.

$F_A \in \{\Lambda_1, \Lambda_2\}$ is a binary strategy used by the attacker A, where Λ_1 indicates that the attack is launched and Λ_2 indicates that the attack is called off.

U_D and U_A indicate the payoffs for the defender D and the attacker A respectively.

We assume that $D \in \{D_H, D_R\}$ follows a distribution rule such that $\{P(D_H) = \theta, P(D_R) = 1-\theta\}$. Meaning that θ is the proportion of HP in the network. Since there is at least one real server in the network, we know that $\theta < 1$.

The attacker has 2 information sets. The first one corresponds to the defender providing services Ω_1. In this case, the attacker is facing a responding system and he is unable to know which system he's facing (HP or RS). The second information set corresponds to the defender not providing services Ω_2. In this case, the attacker is facing a non-responding system and when the attacker decides to launch the attack Λ_1, we consider that he will attack the remaining systems (HP when RS do not provide services and RS when HP do not provide services), independently of the state of the remaining system.

Symbols	Descriptions
D	Defender
D_H	Honeypot (HP)
D_R	Real Server (RS)
A	Attacker
Ω_1	Services provided
Ω_2	Services not provided
Λ_1	Attack launched
Λ_2	Attack not launched
U_D	Defender's payoff
U_A	Attacker's payoff
α	Reward of an attack on RS ($0 < \alpha$)
β	Reward of RS working properly ($0 < \beta$)
μ	Reward of an attack on HP ($0 < \mu < \alpha$)
δ	Cost of maintenance of HP ($0 < \delta$)
γ	Cost of an attack ($0 < \gamma << \alpha$)
θ	Proportion of HP ($0 \leq \theta < 1$)

Table 2: List of symbols in the paper

The defender D has 4 strategies, namely $\{(\Omega_1, \Omega_1), (\Omega_1, \Omega_2), (\Omega_2, \Omega_1), (\Omega_2, \Omega_2)\}$.
The attacker A has 4 responses, namely $\{(\Lambda_1|\Omega_1), (\Lambda_2|\Omega_1), (\Lambda_1|\Omega_2), (\Lambda_2|\Omega_2)\}$.

This leads to 8 possible outcomes shown in figure 5.

4.1.3 Payoffs

In the following, we will explain the different payoffs encountered in our game model. These payoffs can be better understood thanks to the game tree (see figure 5).

Ω_2 for real servers
When the defender D does not provide services $\{\Omega_2\}$, he shuts the services down for both attackers and legitimate users. When the defender D stops providing services for a real system $\{D_R\}$, his payoff suffers $-\beta$ since the server is no longer able to provide useful services to legitimate users. Additionally, we consider that when the attacker launches the attack $\{\Lambda_1\}$, he will launch the attack on the remaining system. When the current system doesn't respond, the attacker will focus on the other systems. Hence, the attacker will attack honeypots if the real servers do not provide services. Hence, when an attack is launched, the defender's payoff is $-\beta + \mu$, corresponding to the loss due to unprovided services in addition to the reward of an attack against a honeypot. When he launches the attack $\{\Lambda_1\}$, the attacker's payoff is $-\mu - \gamma$ corresponding to an attack against a honeypot (the remaining system) and the cost of an attack. When no attack is launched $\{\Lambda_2\}$, the defender's payoff corresponds to the lack of real services $-\beta$, whereas the attacker's payoff is null.

Ω_2 for honeypots
Similarly, when the defender D does not provide services $\{\Omega_2\}$ for honeypots D_H, and the attacker launches the attack $\{\Lambda_1\}$, then the attacker will attack remaining systems, meaning the real servers.

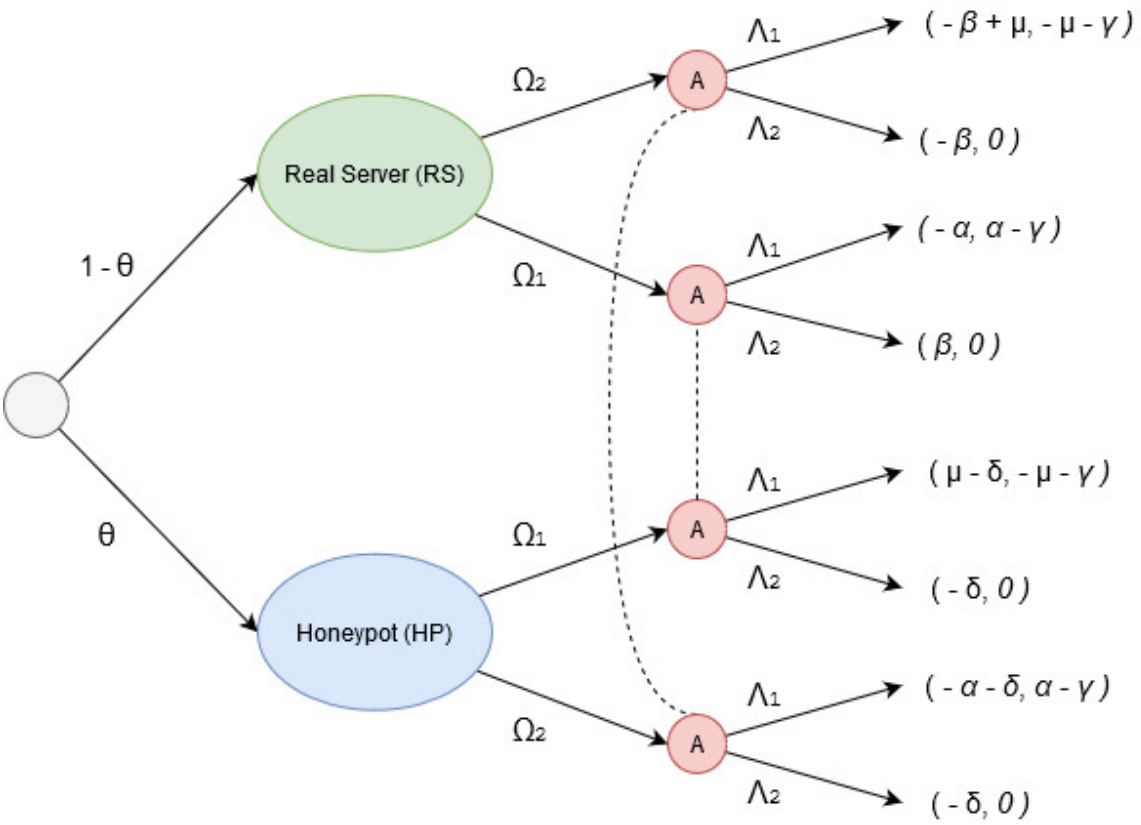

Figure 5: Our game tree from the attacker's perspective

This attack leads to a negative defender's payoff $-\alpha - \delta$ corresponding to the loss of an attack against real servers in addition to the cost of deployment of honeypots which is unavoidable since the defense infrastructure (hence the defense costs) is set at the beginning of the game. When he launches the attack $\{\Lambda_1\}$, the attacker's payoff is $\alpha - \gamma$ corresponding to an attack against a real server (the remaining system) and the cost of an attack. When no attack is launched $\{\Lambda_2\}$, the defender's payoff corresponds to the cost of deployment of honeypots $-\delta$, whereas the attacker's payoff is null.

Ω_1 for real servers
When Real Servers $\{D_R\}$ provide services $\{\Omega_1\}$ and the attacker A decides to launch the attack $\{\Lambda_1\}$, then the defender D has a payoff $-\alpha$ since the RS suffers from the attack, and the attacker A has a payoff $\alpha - \gamma$, corresponding to the reward of this successful attack minus the cost of an attack. If A does not launch the attack $\{\Lambda_2\}$, the defender D has a payoff β since the RS is able to provide services to legitimate users, whereas the attacker A has a 0 payoff.

Ω_1 for honeypots
When Honeypots $\{D_H\}$ provide services $\{\Omega_1\}$ and the attacker A decides to launch the attack $\{\Lambda_1\}$, then the defender D has a payoff $\mu - \delta$ corresponding to the reward of an attack against HP minus the cost of maintenance, and the attacker A has a payoff $-\mu - \gamma$, corresponding to the loss of this

unsuccessful attack minus the cost of an attack. An attack against a honeypot is considered as an unsuccessful attack since the network is not damaged and the defender is able to gather information about the attacker's strategy (detect the attacker). If A does not launch the attack $\{\Lambda_2\}$, the defender D has a payoff $-\delta$ since HP entail maintenance costs, whereas the attacker A has a 0 payoff.

Because the attacker's information set contains two nodes, we cannot use backward induction to solve the game. This naturally leads us to the idea of converting the game to strategic form and looking for a Bayesian Nash equilibrium using a game matrix [18]. This introduces the next section.

		Attacker	
	(HP, RS)	Λ_1	Λ_2
Defender	(Ω_1, Ω_1)	$\theta(\mu-\delta)+(1-\theta)(-\alpha)$, $\theta(-\mu-\gamma)+(1-\theta)(\alpha-\gamma)$	$\theta(-\delta)+(1-\theta)\beta$, 0
	(Ω_1, Ω_2)	$\theta(\mu-\delta)+(1-\theta)(-\beta+\mu)$, $-\mu-\gamma$	$\theta(-\delta)+(1-\theta)(-\beta)$, 0
	(Ω_2, Ω_1)	$-\alpha-\theta\delta$, $\alpha-\gamma$	$\theta(-\delta)+(1-\theta)\beta$, 0
	(Ω_2, Ω_2)	$\theta(-\alpha-\delta)+(1-\theta)(-\beta+\mu)$, $\theta(\alpha-\gamma)+(1-\theta)(-\mu-\gamma)$	$\theta(-\delta)+(1-\theta)(-\beta)$, 0

Table 3: Payoff matrix

4.2 Equilibrium analysis

The assessment of equilibria in a Bayesian game model boils down to the comparison of the different possible outcomes for each player of the game. From this comparison, we extract dominant strategies, meaning strategies leading to maximized payoffs regardless of the strategy of the other players. In most cases, there is no dominant strategy for both players in non-cooperative games. Hence, we need to study mixed strategies where players alternate strategies depending on parameters in order to maximize their payoff. In this section, we will first look for dominant strategies, then we will study mixed strategies and finally we will set our assumptions for the simulation.

First, we will look for dominant strategies for the defender. To do so, we will compare the payoffs of the different strategies.

When the attacker launches the attack $\{\Lambda_1\}$:

$$U_D((\Omega_1,\Omega_1)|\Lambda_1) - U_D((\Omega_1,\Omega_2)|\Lambda_1) = (1-\theta)(\beta-\alpha-\mu) \tag{1}$$

$$U_D((\Omega_1,\Omega_1)|\Lambda_1) - U_D((\Omega_2,\Omega_1)|\Lambda_1) = \theta(\alpha+\mu) > 0 \tag{2}$$

$$U_D((\Omega_1,\Omega_1)|\Lambda_1) - U_D((\Omega_2,\Omega_2)|\Lambda_1) = \theta(\alpha+\mu) + (1-\theta)(\beta-\alpha-\mu) \tag{3}$$

$$U_D((\Omega_1, \Omega_2)|\Lambda_1) - U_D((\Omega_2, \Omega_1)|\Lambda_1) = \alpha + \mu - (1-\theta)\beta \tag{4}$$

$$U_D((\Omega_1, \Omega_2)|\Lambda_1) - U_D((\Omega_2, \Omega_2)|\Lambda_1) = \theta(\alpha + \mu) > 0 \tag{5}$$

$$U_D((\Omega_2, \Omega_1)|\Lambda_1) - U_D((\Omega_2, \Omega_2)|\Lambda_1) = (1-\theta)(\beta - \alpha - \mu) \tag{6}$$

When the attacker does not launch the attack $\{\Lambda_2\}$:

$$U_D((\Omega_1, \Omega_1)|\Lambda_2) - U_D((\Omega_1, \Omega_2)|\Lambda_2) = 2\beta(1-\theta) > 0 \tag{7}$$

$$U_D((\Omega_1, \Omega_1)|\Lambda_2) - U_D((\Omega_2, \Omega_1)|\Lambda_2) = 0 \tag{8}$$

$$U_D((\Omega_1, \Omega_1)|\Lambda_2) - U_D((\Omega_2, \Omega_2)|\Lambda_2) = 2\beta(1-\theta) > 0 \tag{9}$$

$$U_D((\Omega_1, \Omega_2)|\Lambda_2) - U_D((\Omega_2, \Omega_1)|\Lambda_2) = -2\beta(1-\theta) < 0 \tag{10}$$

$$U_D((\Omega_1, \Omega_2)|\Lambda_2) - U_D((\Omega_2, \Omega_2)|\Lambda_2) = 0 \tag{11}$$

$$U_D((\Omega_2, \Omega_1)|\Lambda_2) - U_D((\Omega_2, \Omega_2)|\Lambda_2) = 2\beta(1-\theta) \tag{12}$$

From equations 2 and 8 we observe that (Ω_2, Ω_1) is dominated by (Ω_1, Ω_1). Similarly, from equations 5 and 11 we observe that (Ω_2, Ω_2) is dominated by (Ω_1, Ω_2).

		Attacker	
	(HP, RS)	Λ_1	Λ_2
Defender	(Ω_1, Ω_1)	$\theta(\mu-\delta)+(1-\theta)(-\alpha)$, $\theta(-\mu-\gamma)+(1-\theta)(\alpha-\gamma)$	$\theta(-\delta)+(1-\theta)\beta$, 0
	(Ω_1, Ω_2)	$\theta(\mu-\delta)+(1-\theta)(-\beta+\mu)$, $-\mu-\gamma$	$\theta(-\delta)+(1-\theta)(-\beta)$, 0

Table 4: Payoff matrix with dominated strategies removed

Now between strategies (Ω_1, Ω_1) and (Ω_1, Ω_2), we observe from equation 1 that (Ω_1, Ω_1) dominates if and only if $\beta - \mu > \alpha$. We can easily understand the underlying practical condition: "Is an attack more damaging than not providing services?". Since players are rational in Bayesian games, the defender would prefer to stop providing services if there is a risk that an attack deals more damage than not providing services. In real life, we cannot imagine that the AMI network does not provide services on a regular basis. Moreover, the defender does not always know that there is an attack currently occurring (stealth attack), he is not able to stop providing services only when an attack is discovered. Hence we might make assumptions on the values of β, μ and α.

However, we will first take a look at mixed strategies for both players since pure Bayesian strategies are compromised under the current parameters' domains (we don't yet assume that $\beta > \alpha + \mu$). For a player to be willing to mix over two strategies, he must be indifferent between them; otherwise, he would play his preferred pure strategy [18]. This must be true for each player, and this insight enables us to find the equilibrium mixing probabilities. To keep each opponent indifferent, we have the following two equations:

Let p the probability that D chooses strategy (Ω_1, Ω_1) (the defender provides services for both entities)
Let q the probability that A chooses strategy Λ_1 (the attacker launches the attack).

$$\begin{aligned} & p[\theta(-\mu-\gamma)+(1-\theta)(\alpha-\gamma)]+(1-p)[-\mu-\gamma] = 0p+0(1-p) \\ \Rightarrow & p(1-\theta)(\alpha+\mu)-\mu-\gamma = 0 \qquad (0 \le \theta < 1, \gamma < \alpha) \\ \Rightarrow & p = \frac{\mu+\gamma}{(1-\theta)(\mu+\alpha)} \end{aligned} \tag{13}$$

$$\begin{aligned} & q[\theta(\mu-\delta)+(1-\theta)(-\alpha)]+(1-q)[\theta(-\delta)+(1-\theta)\beta] \\ = & q[\theta(\mu-\delta)+(1-\theta)(-\beta+\mu)]+(1-q)[\theta(-\delta)+(1-\theta)(-\beta)] \\ \Rightarrow & q(1-\theta)(\beta-\alpha-\mu)+(1-q)(1-\theta)2\beta = 0 \qquad (0 \le \theta < 1) \\ \Rightarrow & q = \frac{2\beta}{\beta+\alpha+\mu} \end{aligned} \tag{14}$$

We observe from equation 14 that to have an equilibrium, we need : $\beta < \alpha + \mu$, otherwise, there is no BNE. This condition means that it is preferable for the defender D to stop providing services in order to avoid the damages caused by an attack. This rational behavior does not fit reality since the goal of the defender (network administrator) is to provide services to legitimate users. The defender cannot simply shut services down in order to avoid attacks.

Hence, from now on, we will assume that it is more costly to deny services of a real server than to undertake an attack. We simulate the fact that if the defender does not provide services, legitimate users will go away.
Set $\beta > \alpha + \mu$.

Thanks to this new condition, we now have a pure dominant strategy for the defender D, namely (Ω_1, Ω_1). The payoff matrix can be simplified to table 5.

		Attacker	
		Λ_1	Λ_2
Defender	(Ω_1, Ω_1)	$\theta(\mu-\delta)+(1-\theta)(-\alpha)$, $\theta(-\mu-\gamma)+(1-\theta)(\alpha-\gamma)$	$\theta(-\delta)+(1-\theta)\beta$, 0

Table 5: Payoff matrix with dominated strategies removed under assumption $\beta > \alpha + \mu$

Now the equilibrium for the attacker A boils down to :

$$\begin{aligned} &\theta(-\mu-\gamma)+(1-\theta)(\alpha-\gamma)=0 \\ \Rightarrow &\theta = \frac{\alpha-\gamma}{\alpha+\mu} \end{aligned} \tag{15}$$

For practical applications, it is interesting to know which proportion of honeypots θ makes the attacker A choose strategy Λ_2 (not launch the attack). From this simplified model, we have the condition : $\theta \geq \frac{\alpha-\gamma}{\alpha+\mu}$.

4.3 Simulations

In this section, we present the results of our simulations which were established by manually setting some parameters and by varying other parameters. These simulations are designed to study the variations of θ under various constraints. The Game Model used to build these simulations is extremely simple and the results don't pretend to perfectly reflect reality. However, the results give a rough idea of optimal solutions concerning the deployment of honeypots in an AMI network.

First, we evaluate the impact of the variation of the attack's reward α and the honeypot's reward μ on the minimal value of θ for an attacker to call off his attack (cf. equation 15). To do so, we manually set some parameters according to common sense. The absolute values don't matter in this case since we essentially look at ratios. In the first simulation (see fig. 6), we make α vary between 1 and 1000. We consider that the cost of an attack is approximately the tenth of the reward, hence $\gamma = \frac{1}{10}\alpha$. The values of the reward of an attack against a honeypot are discrete and belong to $\mu \in [1, 250, 500, 750, 1000]$.

Hence we study :
$\theta = \frac{\alpha-\gamma}{\alpha+\mu}$, with: $\alpha \in [1:1000]$, $\gamma = \frac{1}{10}\alpha$, and $\mu \in [1, 250, 500, 750, 1000]$

From this simulation, shown in figure 6, we observe that the reward of an attack on a honeypot μ has an important impact on the minimal proportion of honeypots in the network θ. When honeypots don't act as Intrusion Detection Systems (IDS), meaning that they don't provide a positive payoff for the defender ($\mu \approx 0$), the equilibrium value of θ rapidly increases with the damages caused by the attack α, as shown in figure 6. Moreover, we notice that a proportion of three honeypots for one real-system ($\theta = 3/4$) is greater than all recommended values of θ with $\mu \geq 200$. This gives a rough idea of a practical honeypot distribution.

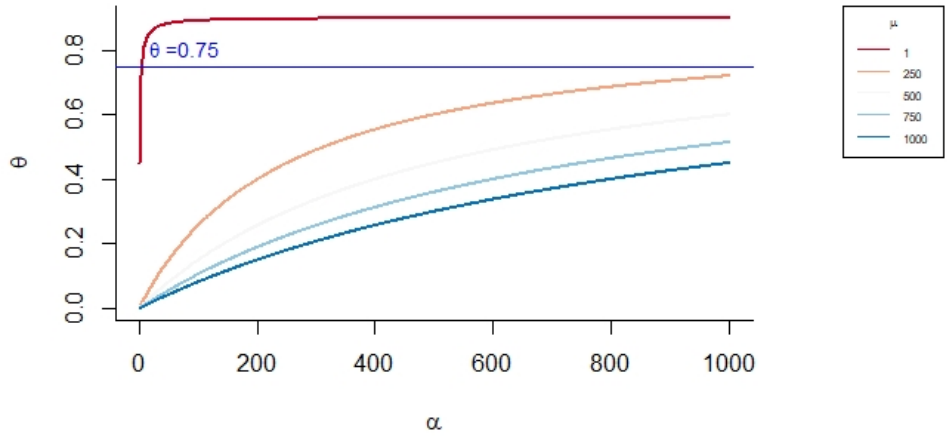

Figure 6: Equilibrium value of θ vs. the reward of an attack α under various values of μ

Now we study the impact of the cost of an attack γ on the equilibrium value of θ. The relationship between the two is linear hence the results of the simulation are extremely predictable.
For this simulation we set : $\theta = \frac{\alpha-\gamma}{\alpha+\mu}$, with: $\gamma \in [1:1000]$, $\alpha = 1000$, and $\mu \in [100, 250, 500]$

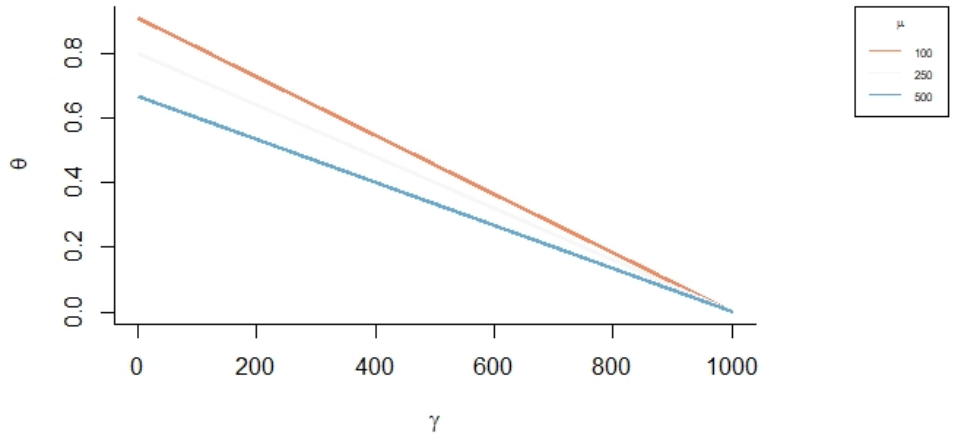

Figure 7: Equilibrium value of θ vs. the cost of an attack γ under various values of μ

From this simulation (fig. 7) we visualize the relationship between the cost of an attack γ ($0 < \gamma < \alpha$) and the proportion of honeypots in the network θ. Practical conclusions cannot be

drawn from this plot but it gives a graphical help to better understand the importance of maximizing the cost of an attack, if the goal is to prevent attacks.

Finally, we assess the defender's payoff U_D as a function of the attack's reward α considering various values of the honeypot distribution θ. Depending on the attacker's strategy $\{\Lambda_1, \Lambda_2\}$, the defender's payoff calculation varies. Since the players are rational in Bayesian games, we consider that the attacker will choose his dominant strategy under some parameters values. Meaning that we use equation 15 to build an *if else* statement. If $\theta > \frac{\alpha-\gamma}{\alpha+\mu}$, then the attacker will prefer not to attack $\{\Lambda_2\}$. On the contrary, if $\theta \leq \frac{\alpha-\gamma}{\alpha+\mu}$, then the attacker will prefer to attack $\{\Lambda_1\}$. We compute the defender's payoff thanks to the payoff matrix 5.

For this simulation we set :
$U_D = ifelse(\theta > \frac{\alpha-\gamma}{\alpha+\mu}, \theta(-\delta) + (1-\theta)\beta, \theta(\mu-\delta) - (1-\theta)\alpha))$
$\alpha \in [1:1000]$, $\theta \in [0:0.9, by = 0.1]$, $\gamma = \frac{1}{10}\alpha$, $\delta = 250$, $\beta = \alpha + \mu + 1$ and $\mu = 180$

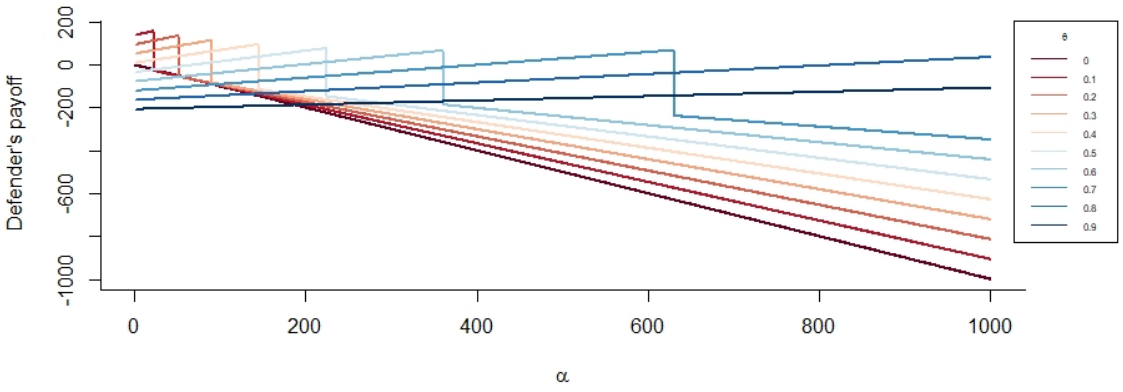

Figure 8: Defender's payoff value U_D vs. the reward of an attack α under various honeypot distributions θ

From this simulation (fig. 8) we observe a pattern for payoff values depending on the proportion of honeypots θ. Indeed, the reward for the defender is increasing until a threshold value of α, meaning that the distribution of honeypots θ "prevents" attacks until this value, and even helps the defender to identify the attacker. This threshold value of α is the minimal reward value of an attack on a real system which becomes beneficial for the attacker. Exceeding this threshold value of α, the defender suffers significant payoff losses.

The simulation shows that our model will tend to maximize the value of θ, in order to keep the attacker from attacking (Λ_2) for all values of α. In other words, we will select a value of θ that puts the threshold value of α outside the considered α range. In order to maximize the defender's payoff, we look at the first value of θ that forces the attacker to choose Λ_2 for all values of α. Figure 8 shows this situation where $\theta = 0.8$ prevents attacks for all considered values of α, and is preferred to $\theta = 0.9$ which also prevents all attacks but leads to a lesser defender's payoff.

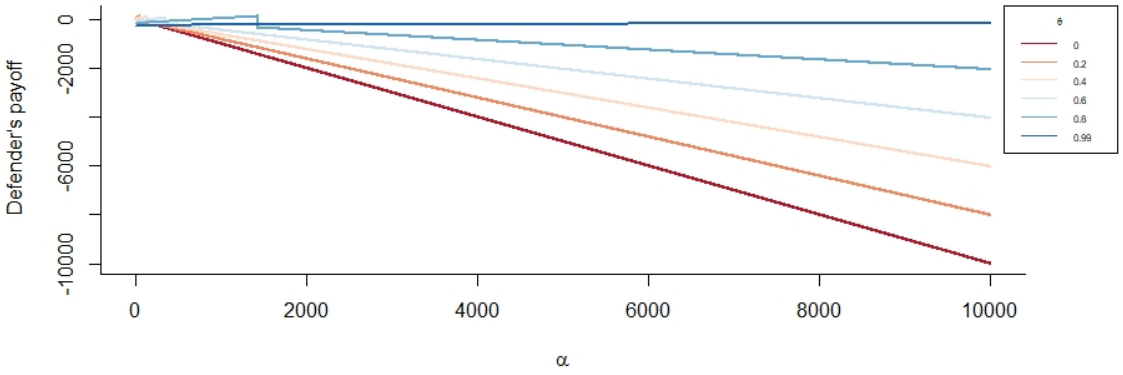

Figure 9: Defender's payoff value U_D vs. the reward of an attack α under various honeypot distributions θ

However, from this simulation we can understand that our game model will always allow the defender to find a θ value that protects him against all attacks. We provide the defender with an "easy-win" situation that is illustrated in figure 9, where attack's reward α goes up to 10000 and a proportion of honeypot $\theta = 0.99$ is still able to prevent any attack, even if it means to deploy hundreds of honeypots in the network. This shows the limitations of our game model and simulations. We need to assess cost and efficiency constraints to a greater extend in order to avoid such unrealistic results.

5 Game model under cost and efficiency constraints

The game model presented in section 4.1 is deliberately simple and does not fully assess cost constraints for the defender. The results we found thanks to the equilibrium analysis and simulation will tend to maximize the proportion of honeypots in the network. The defender is provided with a tool (honeypot) that is greatly efficient against the attacker and which cost is negligible compared to payoff results. This "easy-win" solution consisting of deploying a proportion of honeypots $\theta = 0.99$ is not feasible in real life due to budgetary constraints suffered by the defender. Moreover, a too important proportion of honeypots in the network would give the attacker the opportunity to find similarities between the various honeypots, hence reducing the efficiency of the lure. Meaning that 9 honeypots presenting resemblance in a network with one real server would lead the attacker to target the isolated system. In this section, the goal is to assess these various constraints (cost and efficiency) by adding complementary information to the game model. We will start from a very simple parameter constraint, then we will include the notion of honeypot efficiency thanks to a probability function to finally study the results thanks to advanced simulations.

5.1 Cost constraint

In order to assess the fact that a defender cannot deploy an infinite number of honeypots in the network, we introduce a cost constraint in the game model. The constraint is simply defined as following :
The defender has a maximum budget Δ for his defense. The deployment and maintenance of one honeypot costs δ_{HP}. δ_{HP} is then the unit cost of a honeypot. There are N_{RS} real servers in the network. There are $N_{HP} = \frac{\theta}{1-\theta} N_{RS}$ honeypots in the network. The defense cost for the defender is $\delta = N_{HP} * \delta_{HP}$, corresponding to the number of honeypots in the network times the unit cost of a honeypot. The constraint is: $\delta \leq \Delta$.

Symbols	Descriptions
Δ	Defender's maximum budget $(0 < \Delta)$
N_{RS}	Number of real servers $(0 < N_{RS})$
N_{HP}	Number of honeypots $(0 \leq N_{HP})$
δ	Defender's defense cost $(0 \leq \delta \leq \Delta)$
δ_{HP}	Honeypot unit cost $(0 < \delta_{HP})$

Table 6: Additional symbols in the paper

From this very simple cost constraint we can develop conditions on the proportion of honeypots in the network θ. We have :

$$\begin{aligned} \delta =& N_{HP} * \delta_{HP} = \frac{\theta}{1-\theta} N_{RS} * \delta_{HP} \leq \Delta \quad (\delta \leq \Delta) \\ \Leftrightarrow& \theta \leq \frac{\Delta}{\Delta + \delta_{HP} * N_{RS}} \qquad\qquad (0 \leq \theta < 1) \end{aligned} \tag{16}$$

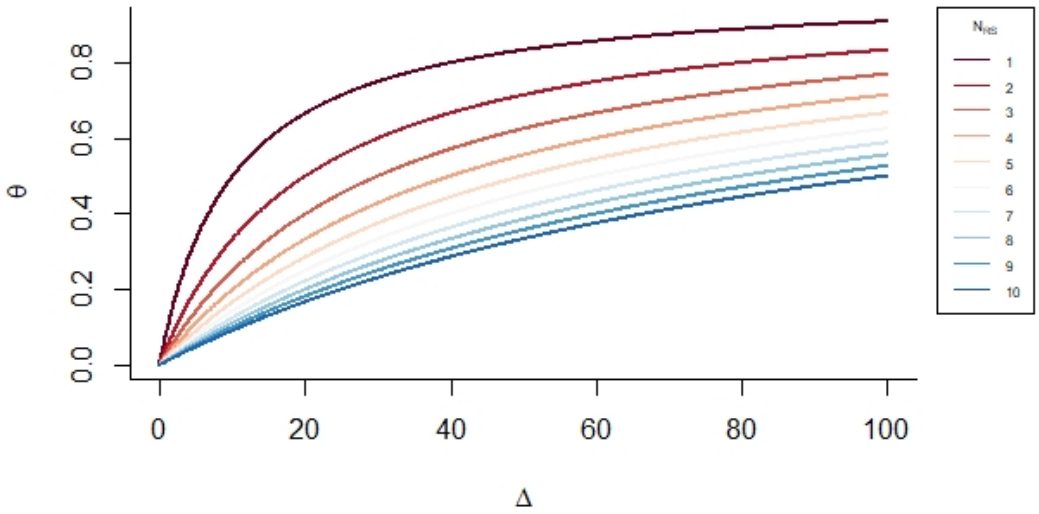

Figure 10: Maximum value of θ vs. maximum budget Δ for various numbers of real servers N_{RS} with $\delta_{HP} = 10$

Equation 16 is illustrated in figure 10. For this simulation, we consider values of Δ between 1 and 100 and a fixed value of 10 for δ_{HP}. Maximum values of θ are displayed for several N_{RS}.

This cost constraint doesn't affect the players' strategies since it doesn't change the attacker's payoff and we still consider that the defender cannot deliberately stop providing services. Hence, we can keep the same payoff matrix as presented in table 5. However, the cost constraint will affect the simulations. To better understand the impact of the cost constraint on the defender's payoff we perform various numerical applications. We plot the defender's payoff as a function of the attack's reward α, for various θ values (as done in figure 8). Then, we make parameters Δ, N_{RS} and δ_{HP} vary. From the results obtained, we extracted interesting cases for a practical deployment of honeypots. We'll notice that optimal values of θ are no longer necessarily maxima.

Figure 11 shows the defender's payoff when he allows a defense budget $\Delta = 100$, with a honeypot unit cost $\delta_{HP} = 10$, and 2 reals systems in the network. We observe that θ is bounded by 0.9. However, the results are similar to the results found in the previous sections. There exists a θ ($\theta = 0.8$) that forces the attacker to call off his attack for every value of α in the simulation. In this case, if we consider that the damages caused by an attack won't exceed 1000, the preferred solution would be the minimal value of θ that prevents all attacks.

Figure 12 shows the defender's payoff when he allows a defense budget $\Delta = 100$, with a honeypot unit cost $\delta_{HP} = 10$ and 7 reals systems in the network. By increasing the number of real systems

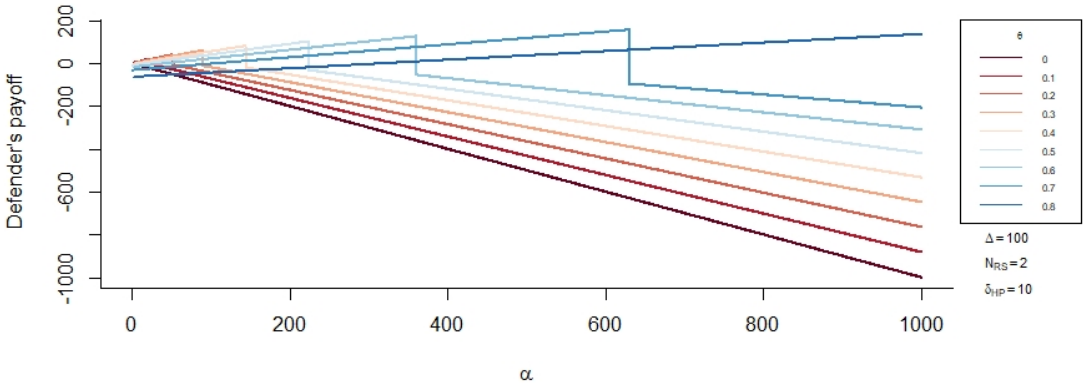

Figure 11: U_D vs. α for various θ with cost constraint and $\Delta = 100, N_{RS} = 2, \delta_{HP} = 10$

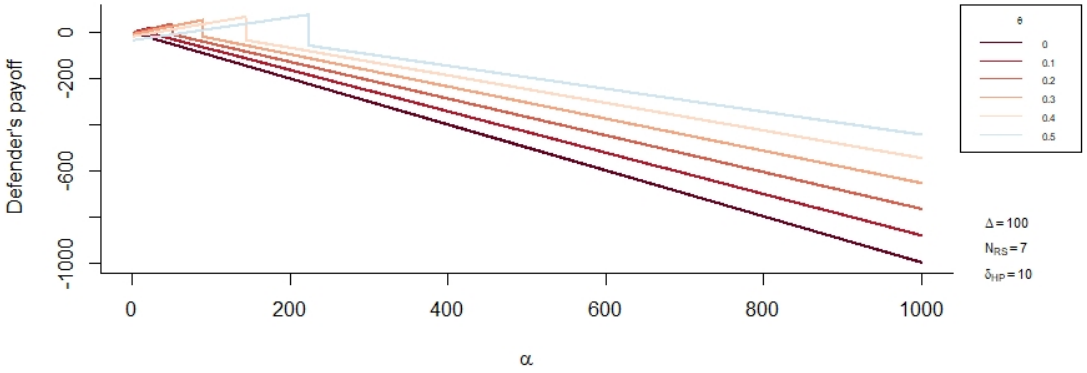

Figure 12: U_D vs. α for various θ with cost constraint and $\Delta = 100, N_{RS} = 7, \delta_{HP} = 10$

in the network N_{RS}, we observe the impact of the cost constraint on the available values of θ. θ is now bounded by 0.6, avoiding the "easy-win" situation. There is no available value of θ preventing all attacks. From the results, we notice that the overall best honeypot distribution is the highest possible value of θ ($\theta = 0.5$ in figure 12). This can be understood by the fact that the rewards of honeypots μ are still greater than the total cost of deployment and maintenance δ. Hence, if honeypots are cheap and they provide the defender with positive payoff when the attacker chooses strategy Λ_1 (the attacker launches the attack), then it is more interesting for the defender to deploy a maximum number of honeypots in the network. Hence, if no available θ allows to prevent all attacks, the preferred solution is to maximize θ.

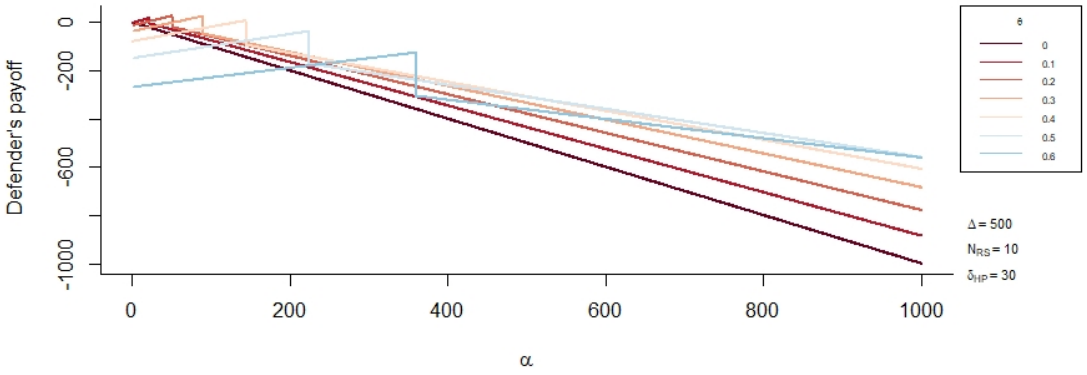

Figure 13: U_D vs. α for various θ with cost constraint and $\Delta = 500, N_{RS} = 10, \delta_{HP} = 30$

Figure 13 shows the defender's payoff when he allows a defense budget $\Delta = 500$, with a honeypot unit cost $\delta_{HP} = 30$, and 10 real systems in the network. By increasing the size of the network, the budget and the unit cost of honeypots, we observe an interesting test case where optimal values of θ are not maximized. θ is bounded by 0.7 and no value of θ allows to prevent all attacks. However, the deployment of honeypots is now more costly. In figure 13, we notice that a proportion $\theta = 0.6$ leads to a lesser payoff than a proportion $\theta = 0.5$ for a large range of α values. The number of real servers N_{RS} and the unit cost of honeypots δ_{HP} being greater, we understand that an increase in θ leads to subsequent defense costs that are no longer absorbed by the reward of honeypots μ. Hence, a too big number of costly honeypots in the network is not beneficial for the defender. The defender must find a balance between honeypots' reward and honeypots' cost. We further study the results in figure 14.

In figure 14, we perform a boxplot (mean, median, quartiles) of the payoffs obtained in figure 13, ordered by the different values of θ. Thanks to this graph, we can clearly see that a value $\theta = 0.5$ is more advantageous for the defender than a value $\theta = 0.6$. Even if there is no θ allowing to prevent all attacks, the optimal value of θ is not maximized as opposed to figure 12.

From the results observed in figures 11, 12 and 13, we understand that the introduction of a cost constraint in the game model leads to bounded values of θ and a better understanding of the defense costs. The optimal distribution of honeypots θ is not trivial to find and depends on the simulation's parameters. However, this doesn't change the game's strategies and only impacts the simulations. In the game model depicted in section 4.1 we assume that all honeypots are infallible. Again, this isn't realistic. Hence the need for a more advanced cost and efficiency assessment, developed in the next section.

5.2 Efficiency constraint

From previous results and in order to evaluate more efficiently the use of honeypots in small networks, we introduce the notion of honeypots' yield. This yield is represented by a probability

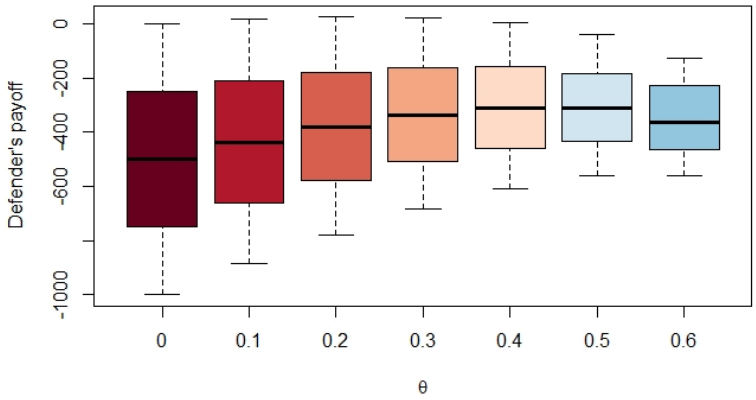

Figure 14: Boxplot of payoffs values for available θ from figure 13

function corresponding to the honeypot's capacity to lure and catch the attacker. A honeypot with an efficiency ratio close to one will have a greater chance to inflict damages to the attacker by deluding him and gathering information about the attacker. On the contrary, a honeypot with a low efficiency ratio will have a greater chance to be identified by the attacker and then, be used against the network to perform a successful attack. In other words, a poor efficiency ratio corresponds to a poorly disguised honeypot and an ineffective Intrusion Detection System.

We assume that this efficiency ratio decreases with the honeypots' proportion in the network θ. Experimented attackers will be able to identify similarities between the honeypots deployed and use this result to spy out the minority systems corresponding to real servers. If 90% of the network's constituents present similarities, attackers knowing that honeypots are deployed will tend to focus their attacks on the remaining 10%, hence decreasing the honeypots' efficiency ratio. This effect can be attenuated by an extra work on the honeypots' configuration. Network administrators can configure honeypots in order to make them different to each other and extremely realistic. This configuration increases the efficiency ratio of the honeypots but it induces a greater deployment and maintenance cost δ_{HP}. We consider that honeypots achieve maximum efficiency when their cost δ_{HP} reaches δ_{max}. We set δ_{max} to fit reality expectations since honeypots are not infinitely upgradeable. Hence the total honeypots' efficiency ratio is inversely proportional to the honeypots' distribution θ and proportional to the δ_{HP}/δ_{max} ratio.

We set the honeypots' efficiency function, plotted in figure 15 : $P(\theta, \delta_{HP}, \delta_{max}) = \frac{\delta_{HP}}{\delta_{max}(1+\theta)}$

Whereas the cost constraint Δ only affected simulations, the efficiency constraint P interferes with the players' payoffs. $P(\theta, \delta_{HP}, \delta_{max})$ being a probability of success (=efficiency), we consider that when an attacker attacks a honeypot HP, he has a P probability of being discovered, hence suffering damage μ from his attack, and a probability (1-P) to escape detection and gain information about the network's constituents (identify a honeypot) resulting in a positive payoff μ. In the previous game model, providing services for honeypots could only increase the defender's payoff since we considered deployment costs as unavoidable. Now, honeypots can be used against the defender by gathering information about the network's topology and then decrease the defender's payoff. Other

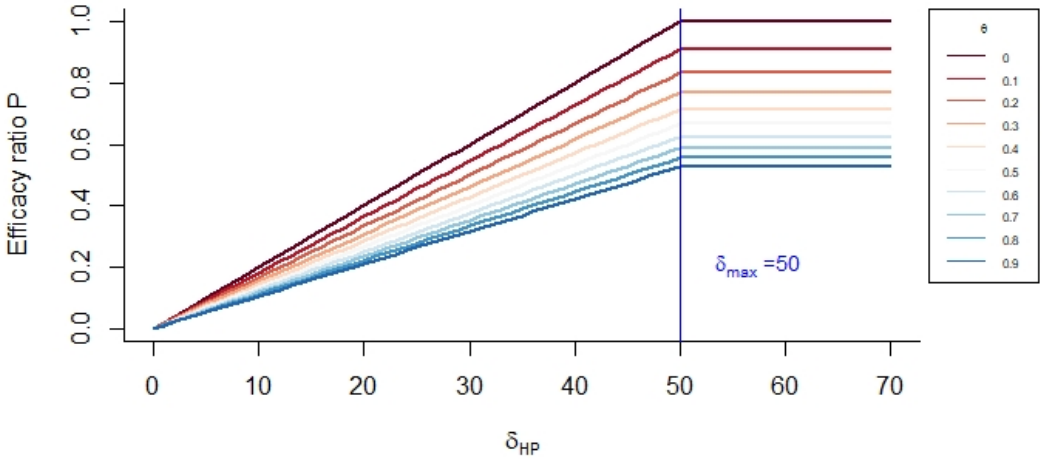

Figure 15: Efficacy P as a function of δ_{HP} for various θ, with $\delta_{max} = 50$

outcomes remain unchanged from the game model described in section 4.1. The new game tree is presented in figure 16, the updated leaves correspond to attacks against honeypots.

The change in payoffs induces new equilibrium calculation, detailed in annex (section 8). However, the results are extremely similar to the previous game model (section 4.2), except that the value of μ is now replaced by $P\mu - (1-P)\mu$. We still consider that it is not conceivable for the defender to deliberately stop providing services, hence we promote strategy (Ω_1, Ω_1) for the defender.

Even if the defender's dominant strategy remains unchanged, this efficiency constraint will force the defender to find a balance between costs and efficiency to maximize his payoff. We will study this balance in the following.

		Attacker	
	(HP,RS)	Λ_1	Λ_2
Defender	(Ω_1, Ω_1)	$\theta(P\mu - (1-P)\mu - \delta) + (1-\theta)(-\alpha)$, $\theta(-P\mu + (1-P)\mu - \gamma) + (1-\theta)(\alpha - \gamma)$	$\theta(-\delta) + (1-\theta)\beta$, 0

Table 7: Payoff matrix with dominated strategies removed when including efficiency

Since we promote strategy (Ω_1, Ω_1) for the defender, we can easily study the attacker's equilibrium. In order to choose strategy $\{\Lambda_1\}$, we need to have the following equation fulfilled :

$\theta(-P\mu + (1-P)\mu - \gamma) + (1-\theta)(\alpha - \gamma) > 0 \qquad (P = \frac{\delta_{HP}}{\delta_{max}(1+\theta)})$

This equation simply means that the attacker's payoff must be positive in order to choose strat-

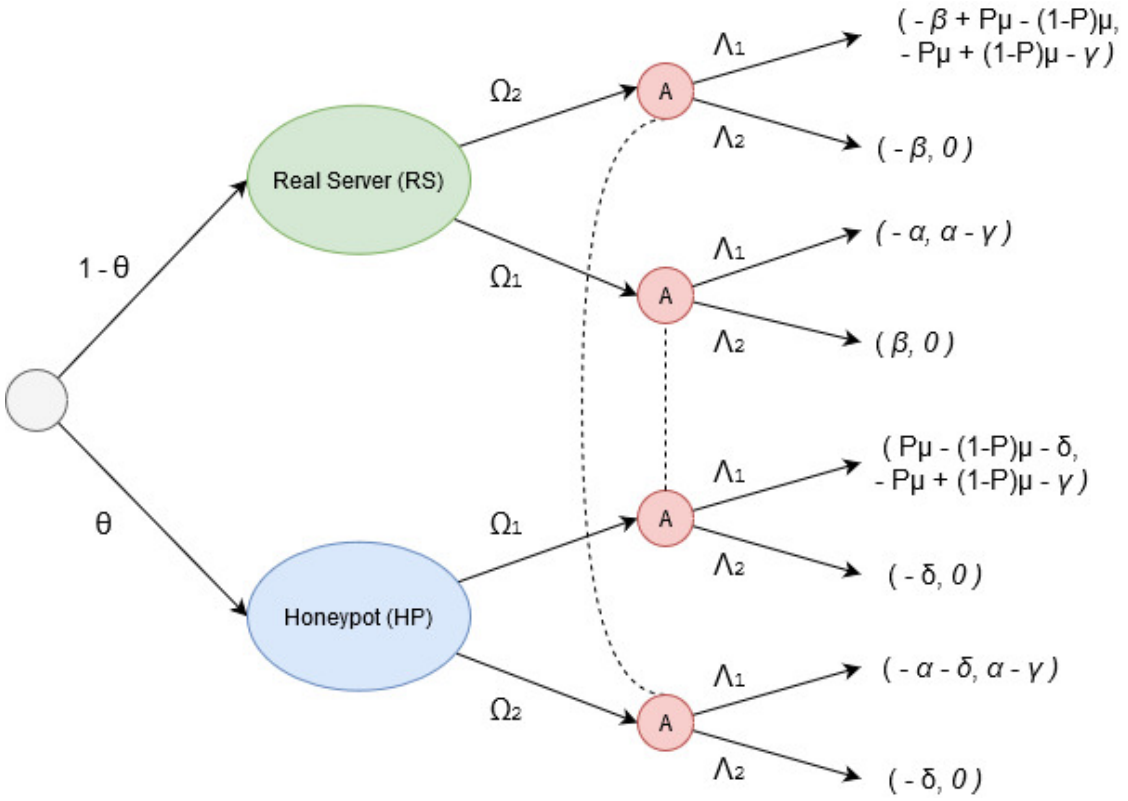

Figure 16: Game tree including efficiency constraint, from the attacker's perspective

egy $\{\Lambda_1\}$. P being a function of θ, we will solve this equation numerically and use it in the following.

5.3 Cost and efficiency constraints simulations

We have designed a game model assessing cost and efficiency constraints. We will now use these results to build a simulation that will allow us to have a better understanding of the optimal honeypot distribution θ for various parameters and under cost and efficiency constraints.

The simulation algorithm is built as follows :

Algorithm 1 enables us to easily evaluate the optimal honeypot deployments. Given a number of real servers N_{RS}, a defense budget Δ, a considered honeypot reward μ and a potential maximum attack damage α_{max}, the defender is able to use this algorithm in order to see which unit cost δ_{HP} he should attribute to his honeypots along with the optimal honeypot distribution θ. In other words, the number of honeypots to deploy in the network and which level of configuration they should have $(\delta_{HP}/\delta_{max})$. The evaluation is performed considering a range of attacks bounded by α_{max}. If the defender has prior information about the potential attackers and the damages they could deal, he can specify a shorter interval $(\alpha \in [\alpha_{min}, \alpha_{max}])$ in order to obtain adapted results.

The following is a numerical application of algorithm 1. To be consistent with previous simulations, we remain in the same order of magnitude for parameters values. The parameters values are

Algorithm 1 Simulation algorithm under cost and efficiency constraints

Input : $N_{RS}, \delta_{max}, \Delta, \mu, \alpha_{max}$

Output : Defender's best cumulative payoffs and the corresponding θ for various values of δ_{HP}

for $\delta_{HP} = 1, 2, ..., \delta_{max}$ **do**
  Calculate the maximum value of θ under cost constraint: $\theta_{max} \leftarrow \frac{\Delta}{\Delta + \delta_{HP} * N_{RS}}$
  for $\theta = 0, 0.05, ..., \theta_{max}$ **do**
    Calculate the honeypot efficiency P: $P \leftarrow \frac{\delta_{HP}}{\delta_{max}(1+\theta)}$
    Calculate the total defense cost δ: $\delta \leftarrow \frac{\delta_{HP} * N_{RS} * \theta}{1-\theta}$
    Initialize the cumulative sum of defender's payoffs $S_D(\theta)$: $S_D(\theta) \leftarrow 0$
    for $\alpha = 1, 2, ..., \alpha_{max}$ **do**
      Set $\beta \leftarrow \alpha + \mu + 1$
      Set $\gamma \leftarrow \alpha / 10$
      Calculate the attacker's payoff U_A:
      $U_A \leftarrow max(\theta(-P\mu + (1-P)\mu - \gamma) + (1-\theta)(\alpha - \gamma), 0)$
      if $U_A > 0$ **then**
        Attacker chooses strategy Λ_1
        Calculate the defender's payoff $U_D((\Omega_1, \Omega_1)|\Lambda_1)$:
        $U_D \leftarrow \theta(P\mu - (1-P)\mu - \delta) + (1-\theta)(-\alpha)$
      else
        Attacker chooses strategy Λ_2
        Calculate the defender's payoff $U_D((\Omega_1, \Omega_1)|\Lambda_2)$:
        $U_D \leftarrow \theta(-\delta) + (1-\theta)\beta$
      end if
      Add the defender's payoff to the cumulative sum: $S_D(\theta) \leftarrow S_D(\theta) + U_D$
    end for
    Store the value of $S_D(\theta)$ in S
  end for
  Return the maximum cumulative sum $S_D(\theta)$ in S and the corresponding θ:
  $S_{max}(\theta) \leftarrow max(S)$
end for
Plot $S_{max}(\theta)$ for each δ_{HP} with the corresponding θ

attributed in order to reflect a common attack scenario, however they are questionable. Each and everyone should apply the algorithm with own adapted values.

For this simulation we set : $N_{RS} = 2, \Delta = 200, \delta_{max} = 30, \mu = 250, \alpha_{max} = 1000$. With these values, we assume that the defense budget Δ is 20% of the potential maximum attack damage α_{max}, that there are few real servers in the network (AMI network) and that a honeypot discovering an attacker yields a benefit μ to the defender, corresponding to 25% of the potential maximum attack damage α_{max}. The results are presented in figure 17, where δ_{HP} varies from 1 to δ_{max}.

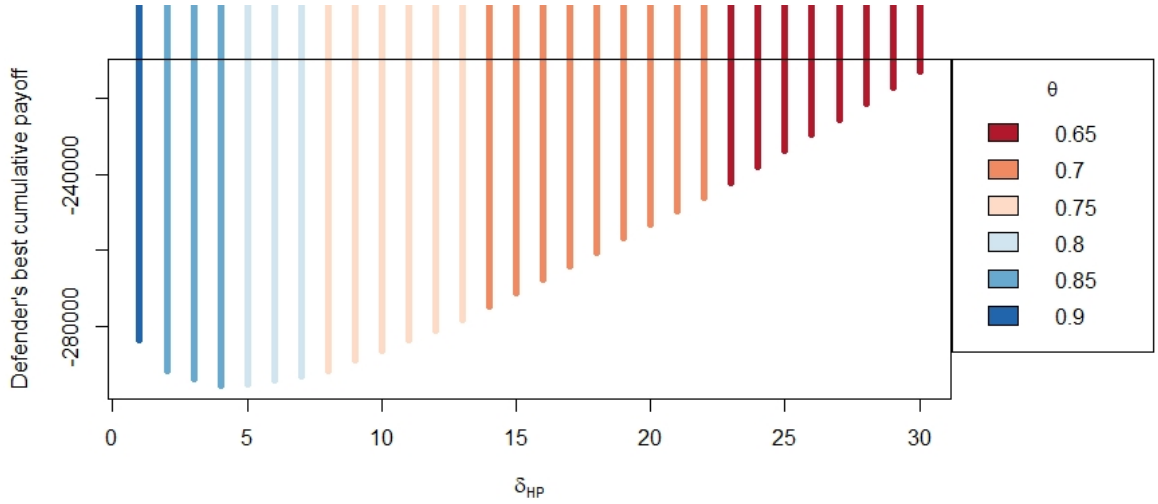

Figure 17: Cumulative defender's payoffs $S_{max}(\theta)$ with associated θ vs. honeypot unit cost δ_{HP}

From the defender's point of view, figure 17 is a summary of the best honeypot distribution θ for each δ_{HP}. Moreover, by displaying the cumulative payoffs, corresponding to $S_{max}(\theta)$ in algorithm 1 , this graph allows to enhance the best (δ_{HP}, θ) combination.

As an example, when a defender attributes a cost $\delta_{HP} = 15$ to the honeypots' deployment, knowing that the maximum value is $\delta_{max} = 30$, the best honeypot distribution θ, meaning the one leading to a maximized payoff, corresponds to $\theta = 0.7$. Now, from all these optimized (δ_{HP}, θ) combinations, the worst choice for the defender would be to attribute a cost $\delta_{HP} = 4$ with $\theta = 0.85$. On the contrary, the best choice for the defender would be to attribute a honeypot cost $\delta_{HP} = \delta_{max} = 30$ with a honeypot distribution $\theta = 0.65$. This combination $(\delta_{HP} = 30, \theta = 0.65)$ yields the best cumulative payoff for attacks ranging from 1 to $\alpha_{max} = 1000$.

From figure 17, we observe that for low values of δ_{HP}, meaning for poorly configured honeypots, the optimal honeypot distribution gets closer to its maximal possible value. The intuition coming from this observation would be to increase the number of honeypots in the network if the defender decides to configure them minimally. However, the deployment of many poorly configured honeypots is not optimal when we consider other possible combinations. From figure 17, we notice that the optimal combination for the defender is to maximize the configuration of his honeypots, even if this means to decrease the number of honeypots in the network. This observation would tend to prefer quality over quantity. The following simulations will try to confirm or disprove this intuition.

Figure 18 also presents the maximum cumulative defender's payoff with associated θ as a function of the attributed honeypot unit cost δ_{HP}. However, for this simulation we increase the cost

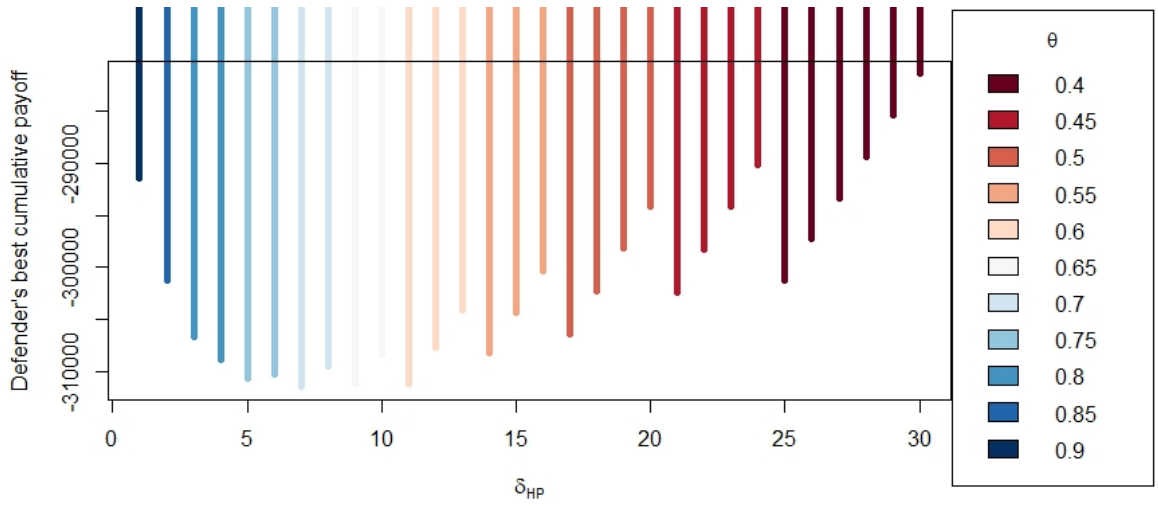

Figure 18: Cumulative defender's payoffs $S_{max}(\theta)$ with associated θ vs. honeypot unit cost δ_{HP}

constraints by setting the number of real servers to $N_{RS} = 3$ and substantially decreasing the defense budget to $\Delta = 60$. From comparing figure 17 and figure 18, we observe that decreasing the defense budget leads to worse defender's payoffs. The mean value of cumulative payoffs in figure 17 is -264012.4 whereas the mean value is -301297.4 in figure 18, hence a 15% decrease. Moreover, we notice that similar values of θ are grouped by δ_{HP} intervals. For a same value of θ, the optimized combination consists of increasing the value of δ_{HP}, in most cases. However, we observe that the strategy consisting of deploying a huge number of poorly configured honeypots is no longer the worst possible choice for the defender. We might want to develop this trend in further simulations.

In order to see if the best solution is always to maximize the honeypot configuration by setting $\delta_{HP} = \delta_{max}$, we perform numerous simulations and we study the evolution of the best (δ_{HP}, θ) combination under parameters variation. To build the results presented in figure 19, we propose algorithm 2 which returns the optimal (δ_{HP}, θ) for each value of the varying parameter. The best (δ_{HP}, θ) corresponds to the maximum cumulative payoff, e.g. $(\delta_{HP} = 30, \theta = 0.65)$ in figure 18.

When applying algorithm 2, fixed values for parameters which don't vary are the same as the previous simulations : $N_{RS} = 2, \Delta = 200, \delta_{max} = 30, \mu = 250, \alpha_{max} = 1000$.

Algorithm 2 Simulation algorithm for the study of δ_{HP} under parameter variation

Input : Varying parameter X, $X \in \{N_{RS}, \delta_{max}, \Delta, \mu\}$ and its maximum value X_{max}
Output : Defender's best (δ_{HP}, θ) choice for each $X \in [1, .., X_{max}]$

1: **for** $X = 1, ..., X_{max}$ **do**
2: Apply algorithm 1 with inputs : $X \cup \{N_{RS}, \delta_{max}, \Delta, \mu\} \setminus X \cup \alpha_{max}$
3: Return the value of (δ_{HP}, θ) corresponding to the best cumulative payoff
4: **end for**
5: Plot the values of δ_{HP} and θ for each X value

From figure 19, we observe that for all simulations, δ_{HP} only takes two values: 1 and δ_{max}. Meaning that for the cases studied, there exists two optimal combinations: maximized honeypots

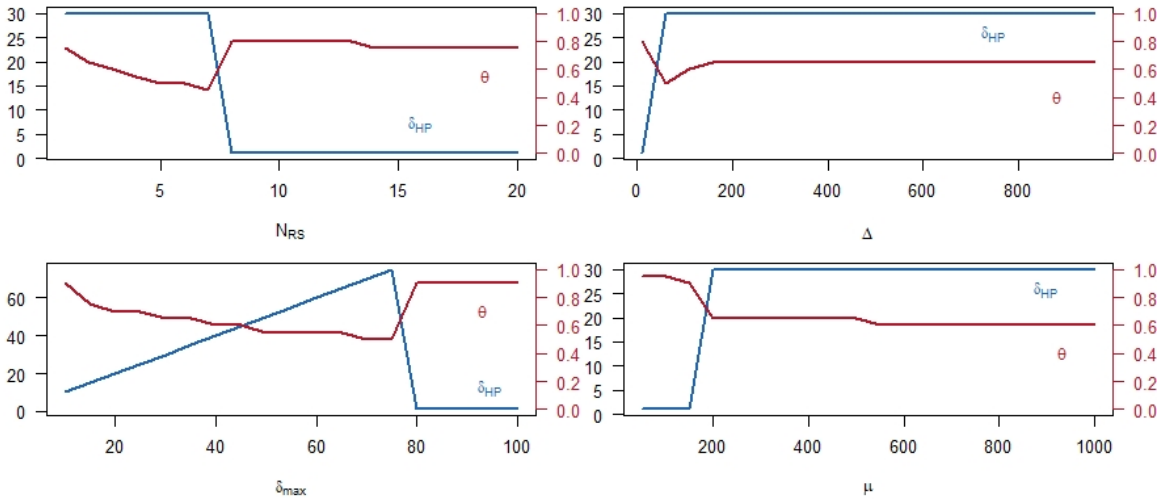

Figure 19: Optimized values of (δ_{HP}, θ) under parameters variation

configuration ($\delta_{HP} = \delta_{max}$) and minimized honeypot configuration ($\delta_{HP} = 1$). The choice between the two depends essentially on constraints applied by parameters. When constraints are high (N_{RS} increases, Δ decreases, δ_{max} increases, μ decreases), the model would prefer to minimize honeypots' configuration and maximize θ. On the other hand, when we reduce the constraints, the model prefers to maximize honeypots' configuration with a stabilized value of θ between 0.6 and 0.8.

From these observations, we can interpret the optimal (δ_{HP}, θ) combinations as response-strategies to constraints. When the defender's resources are insufficient, when there are too many real servers in the network compared to the available defense budget, the preferred strategy is to deploy a huge number of poorly configured honeypots in order to limit the damages caused by attacks. This "mass" strategy yields a better payoff than a very low proportion (due to constraints) of highly configured honeypots. Now, when resources are sufficient, the strategy tends to a stabilized state where the honeypot configuration is maximized ($\delta_{HP} = \delta_{max}$) and the honeypot distribution θ depends on the simulation parameters but is comprised between 0.6 and 0.8 in most cases. These results are interesting since they show that infinite defense resources lead to a stabilized state with optimal (δ_{max}, θ_{stab}) combination. Hence, from our model, we would recommend defenders to reduce constraints by increasing their defense budget in order to be able to approach the (δ_{max}, θ_{stab}) combination corresponding to the stability state. This stability state corresponds to the values of δ_{HP} and θ when Δ tends to infinity. The underlying practical message would be: "give yourself the resources to be able to maximize honeypots' configuration and reach optimal and stabilized honeypot distribution θ_{stab}."

However, these observations also indicate the limits of our game model. The "mass" strategy promoted by our model when resources are insufficient corresponds to a damage limitation strategy that doesn't necessarily fit reality. When we are in the case where the model promotes $\delta_{HP} = 1$, we are, in fact, in a case where the resources of the defender do not allow him to prevent any attack. Even if the defender tries to increase honeypots' efficiency, he is too constrained by budget limitations to be able to be "efficient enough". The attacker always has a positive payoff, hence,

the attacker will always launch attacks. The defender can only maximize his payoff by decreasing his defense costs. Since Bayesian players are rational, the defender will then reduce defense costs since they are useless to keep the attacker from launching attacks. The "mass" strategy is, hence, an abandonment strategy that is not practically applicable.

Our results show that there is a minimum defense budget to allocate in order to counter attacks by having sufficiently effective honeypots, otherwise, honeypots only become a burden to the defender by creating additional defense costs. Past this minimum defense cost, the optimal honeypot deployment corresponds to highly-configured honeypots distributed with a proportion θ that is found thanks to a stability state of infinite defense resources. Our game model doesn't promote the combination of highly-configured honeypots (high-interaction honeypots) and lowly-configured honeypots (low-interaction honeypots) as done in [16] and [17]. However, our assessment of the honeypots' efficiency can be arguable and it could be interesting to study a game model that includes three devices, namely real servers, low-interaction honeypots and high-interaction honeypots and compare the results.

In summary, when a defender wants to deploy honeypots in an AMI network, he can use the proposed simulation algorithms (1, 2) to find the optimal (δ_{HP}, θ) combination in his case in order to maximize his chances of countering an attack. He needs to evaluate the values of the parameters according to his network (set numerical values for his defense budget Δ, the number of servers to protect in his network N_{RS}, his approximation of the potential attack damage α_{max}, etc.). Then, he can perform simulations thanks to the algorithms and choose the (δ_{HP}, θ) that yields the best payoff for him. Results don't pretend to perfectly match reality, however they can give general advice for the deployment of honeypots in an AMI network.

6 Game model under evolutionary simulations

Non-cooperative game theory, as studied in the previous sections of this document, relies on two extremely strong assumptions, called *heroic assumptions* in [19]. The first one is Rationality (*maximization*), meaning that every player of the game is a rational decision maker with a clear understanding of the world. The second assumption is Correctness (*consistency*), meaning that the player's expectations of other players' behavior, is correct. In other words, the players of non-cooperative games under Nash equilibria are omniscient and rational. Considering these assumptions, we can easily wonder how strategies developed in game theory studies can be applied to real-life cases. We studied a non-cooperative game involving an attacker and a defender in the previous sections. We analyzed the strategies available for the defender in order to find the best strategy and we optimized the parameters of this strategy in order to maximize the defender's payoff. All these results were based on the fact that the attacker was rational. The attacker knew the defender's strategy, the proportion of honeypots in the network, the cost of an attack against a honeypot, the payoff of a successful attack, the number of real servers in the network and the cost of an unsuccessful attack. This knowledge is definitely not available in a real-life attack. Hence, we would be tempted to question the usefulness of our results in real-life. Would the optimized parameters (proportion of honeypots and configuration costs), found thanks to the algorithm developed in section 5, be efficient against an irrational attacker? Is the theoretical strategy the best defense strategy in a simulation where the attackers are not as predictable as supposed?

In order to give initial answers, we want to confront our optimization algorithm (cf. algorithm 1) to test cases where the game theory assumptions are no longer fulfilled. To do so, we will develop basic simulation algorithms using evolutionary game theory. In these evolutionary simulations, populations will randomly encounter each other and confront if they are in competition. The fittest population will dominate the confrontation and eliminate the adversary population. By doing so, the number of individuals in each population will vary with time and might lead to the extinction of some populations, to equilibria, or to cycles in the populations. Populations that survive whereas the others are extinct, or populations that have the biggest number of individuals when an equilibrium is reached, will be considered as the most fitted to the study case.

The underlying goal of this section is to probe our theoretical optimizations against simulations that do not respect rationality. We want to see if our results could be useful in real life or, on the contrary, if they are inadequate since the assumptions made are too far away from the real-life cases. This section is organized as follows: first, we will explain the simulation's principles. Then, we will develop the simulation's technical details (structure and algorithms). The simulation's behaviors will be rapidly assessed before the introduction of the test cases used to draw conclusions on our theoretical results. The results found will be discussed in the last subsection. Note that in this section, we will not develop evolutionary game theory (replicator equation, search for Evolutionary Stable Strategies), but we will rather use some simulation techniques used in evolutionary game theory to build a practical simulation model.

6.1 Simulation principles

In this subsection, we will explain the different principles that were used to construct the simulation. As said previously, this simulation is a tool used to observe a theory under varying contexts. We do not extract proofs or theoretical conclusions from the results obtained and we do not base our simulation on existing models (Hawk/Dove, War of attrition, Kin selection) but rather create a new game with self-made rules that try to best translate an irrational real-life confrontation in a local power grid.

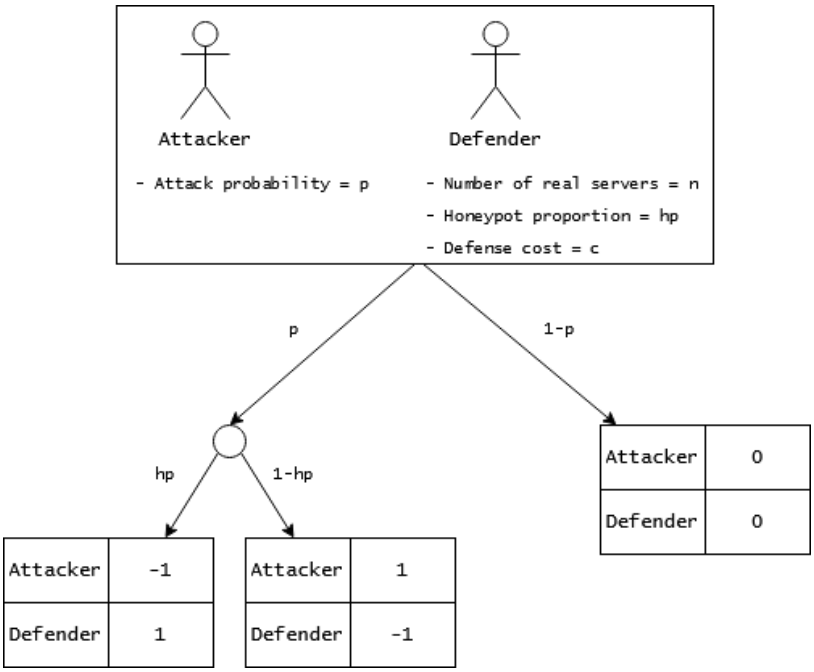

Figure 20: Diagram representing the possible outcomes of a confrontation between an attacker and a defender

Our game model is a finite population evolutionary game with two types of players and competition between them. For a given type of player, variations can be introduced by setting initial parameters (attributes of a variation). These parameters remain unchanged throughout the game for a same variation. A population is a set of players that have the same attributes, in other words, players that represent the same variation. There can be several populations for the same type of player. Our game model is built from the following components:

- **Population** The game is composed of a finite set of players. A player has a type. There are two types of players: Defender and Attacker. Each player has attributes specific to its type. The attributes of each player are set at the beginning of the game, during the initialization. A population is a set composed of players that have the same type and the same attributes.
  - Defenders' attributes are : their number of real servers, their honeypot proportion and their defense cost.
  - Attackers' attributes are: their attack probability.
- **Game rules** At each turn of the game, every player encounters exactly one other player. Populations of defenders are competing with every populations of attackers and with other populations of defenders. Populations of attackers are competing with every populations of defenders but are passive with other populations of attackers. An attacker A, when encountering a defender D (see figure 20), will attack with a probability equal to its own attack probability attribute. We introduce an attack probability in order to simulate the irrationality of the attacker. The attacker's choice of attacking is no longer related to the defender's setup but to a set probability. When attacking, A randomly (and blindly) picks a device in the

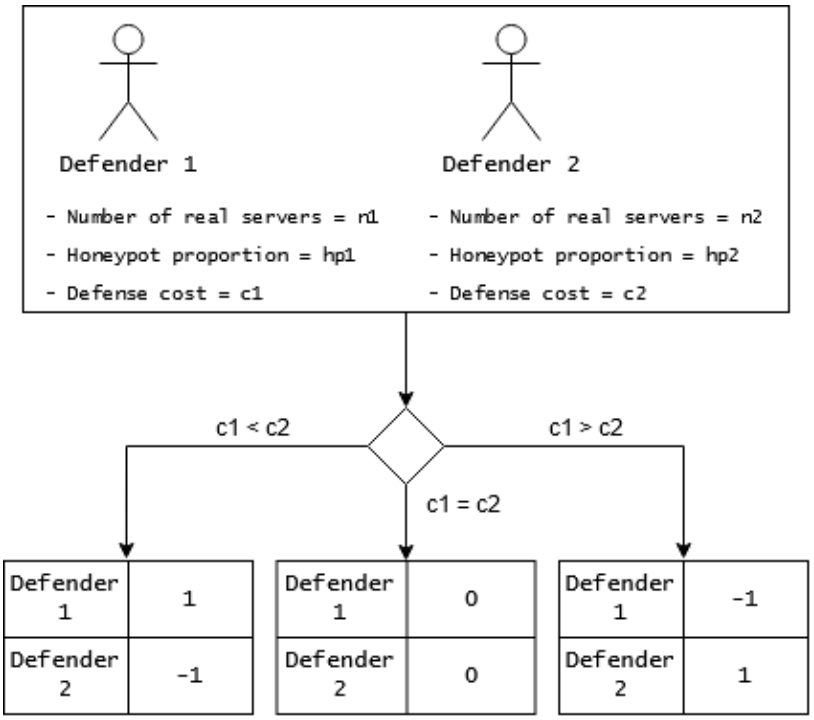

Figure 21: Diagram representing the possible outcomes of a confrontation between two defenders

defender's set of devices. The probability of picking a honeypot is equal to the defender's honeypot proportion attribute. If the picked device is a real server, A is considered to be winning, he will obtain a payoff equal to 1 and D will obtain a payoff equal to -1. On the other hand, if the picked device is a honeypot, D wins and gets a payoff equal to 1 and A yields a payoff equal to -1. When A doesn't attack, both players have a payoff equal to 0. Now, when a defender of population X confronts a defender of population Y (see figure 21), the winner is the defender with the lowest defense cost attribute. The winner yields a payoff equal to 1 and the looser yields a payoff equal to -1.

- **Replicator rule** Let A a player that belongs to population X and B a player that belongs to population Y. Both encounter at generation (turn) n. If the payoff of A is strictly greater than the payoff of B, then A remains in population X and B will join population X for the generation $n+1$. Meaning that player B will be a player of the same type and with the same attributes as A for the next turn.

The encounter between the attacker and the defender is basic. The attacker has a probability $P = (1 - hp)p$ of winning the confrontation, with hp being the honeypot proportion of the defender and p being its own attack probability. The irrationality of the attacker is translated by the attack probability p. The choice between attacking and withdrawing for an attacker is not depending on the defender's setup as opposed to our previous study cases. The only possibility for a defender to increase its chances of surviving against an attacker is then to increase the honeypot proportion hp. However, by increasing its honeypot proportion, the defender will increase its defense cost. When encountering other defenders, minimizing the defense cost allows a better fitness. Defenders need

to find a balance between their honeypot proportion to counter attackers and their defense cost to survive against other defenders. These notions were present in the theoretical assumptions of the previous sections but their consequences are greater in this simulation since they are responsible for the survival of the population.

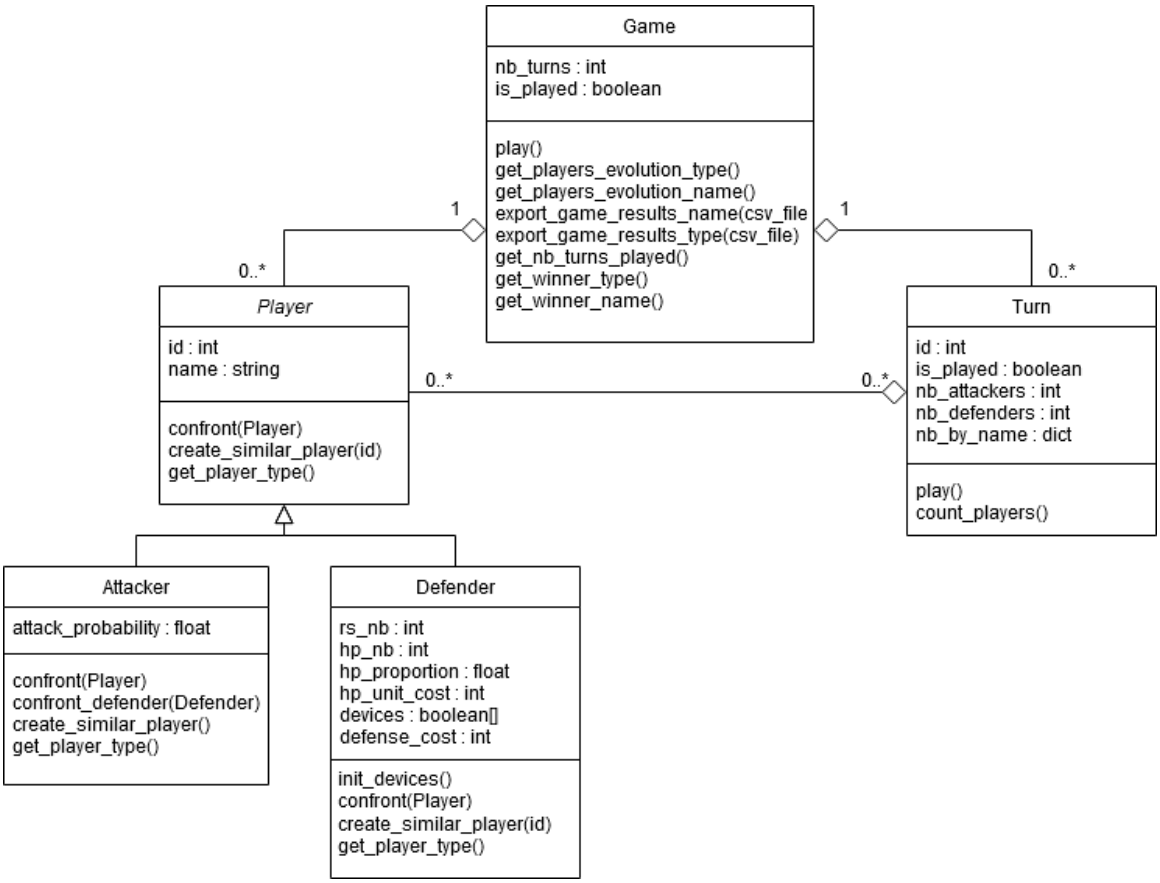

Figure 22: UML class diagram of the implemented simulation

6.2 Simulation structure and algorithms

This simulation has been implemented using Object-Oriented Python programming language. Class objects have been developed, corresponding to the game, the turns of the game and the players of the game respectively (see figure 22). The source code of the simulation is available on Github [1].

To provide additional information to the code's documentation, we represent the UML class diagram of our simulation in figure 22 and we detail the two main algorithms of the game in 3 and 4. As explained in the previous subsection, a Game is composed of several Turns. At each Turn, a new Turn object is created with the current Players set. This newly created Turn object is then played and we update the Game's Players with the resulting Players set after confrontation (Turn play). At

[1] https://github.com/woccelli/game-simulator

each Turn, the Game checks if there remains Players of both types. Otherwise, the game is stopped since no more attackers or defenders are present in the game. Before each Turn is played, the Players set is shuffled in order to randomize encounters between Players. Each player will confront its direct following neighbor in the Players set and the loosing Player of the confrontation will be replaced by a Player similar to the winning Player (same type and attributes).

In order to evaluate our theoretical results, we need to assess the results of numerous games with varying parameters. To do so, we implemented simulation scripts able to record the results of all the games played with their corresponding parameters. In the following subsections, each time a simulation is performed, we will provide the used parameters corresponding to the Players populations, the number of games played, the maximum number of turns for each game, and other information. The scripts are also available in the source code of the project.

Algorithm 3 Game play algorithm

Input : $Players$ (set of objects of type Player), n
Output : List of the n turns played during the game along with their respective Players set

Create a Turn t with input : $Players$
Create an empty list of Turns $turns$
$turns \leftarrow turns \cup t$
for $i = 1, 2, ..., n$ **do**
    Create a new Turn t with input: Players
    $t.play()$
    $turns \leftarrow turns \cup t$
    $Players \leftarrow t.Players$
    if $count(Players.Defenders) == 0$ or $count(Players.Attackers) == 0$ **then**
        Return $turns$
    end if
end for
Return $turns$

Algorithm 4 Turn play algorithm

Input : $Players$ (set of objects of type Player)
Output : Updated set of Players after confrontation

$Players \leftarrow shuffle(Players)$
for i=0..$Players.length - 2$, step=2 **do**
    $win \leftarrow Players[i].confront(Players[i + 1])$
    if win **then**
        $Players[i + 1] \leftarrow Players[i].create\_similar\_player()$
    else
        $Players[i] \leftarrow Players[i + 1].create\_similar\_player()$
    end if
end for
Return $Players$

6.3 Model behavior assessment

In order to better understand the basic behavior of our simulation, we perform simple test cases that are developed in this subsection.

First, we evaluate the importance of the defender's ability to replicate against attackers, meaning, the defender's parameter responsible for its victory in a confrontation against an attacker: the honeypot proportion. The first test case makes defenders with no honeypot ($hp\_proportion = 0$), confront attackers that have an attack probability of 1. In this first simulation (table 8), each time an attacker encounters a defender, he's sure to attack and he's sure to attack a real server, hence, he's sure to win the confrontation. We expect the population of defenders to rapidly go extinct during the simulation.

Game		
	Number of Turns	100
	Number of Games	10000
Defender		
	Def1:	1000
	nb_rs	10
	$hp\_proportion$	0
	hp_unit_cost	30
Attacker		
	Atk1:	1000
	$attack\_proba$	1

Table 8: Input parameters of simulation 1

As expected, table 9 shows that out of 10000 games, defenders go extinct after less than 5 turns in average and they are unable to win against attackers since during a single confrontation against an attacker, their probability of winning is equal to $hp\_proportion * attack\_proba$. Our game model considers that defenders with no honeypots are unable to survive against attackers only.

Number of games played	10000
Average number of turns per game	4.7042
Number of games won by Atk1	10000
Number of games won by Def1	0

Table 9: Results of simulation 1

In order to see if the chances of winning for a defender are correctly modeled in our game model, we implement a new test case where the defender's $hp\_proportion$ is equal to 0.5 and the $attack\_probability$ is equal to 0.25 (see table 10). From the results displayed in table 11, we observe that no games were won by extinction of one population before the limit of 200 turns per game.

Game		
	Number of Turns	200
	Number of Games	1000
Defender		
	Def1:	1000
	nb_rs	10
	hp_proportion	0.5
	hp_unit_cost	30
Attacker		
	Atk1:	1000
	attack_proba	0.25

Table 10: Input parameters of simulation 2

Further tests show that the results are similar for a limit of 500 turns per game. However, wins by extinction were observed when the maximum number of turns was up to 50000. We want to limit the maximum number of turns to 200, hence, in order to assess the populations behavior, we count the number of games where a population has more individuals after the final turn. In our case, out of 1000 games, 511 possess a greater number of attackers after the *200th* turn. A stable state is present at turn *#200* with populations alternating around 50% of the total number of individuals. It is important to note that the number of games won by defenders is not equal to $hp\_proportion * attack\_probability$. Indeed, when the attacker doesn't attack, no population is able to replicate, this outcome simply postpones the confrontation leading to a replication of one population (unless the attack probability is null, but this case is of few interest). By its design, our game model defines the chances of survival (replication) of defenders against attackers as a function of the honeypot proportion. And, the attack probability influences the number of turns needed to reach extinction of one population but has no direct impact on the number of games won by extinction when a single population of defenders is confronted to a single population of attackers. By itself, this model does not propose useful information. However, when combining several populations of defenders, the results become harder to predict and the model helps us having a better understanding of the relationships between the populations.

Number of games played	1000
Average number of turns per game	200
Number of games won by extinction by Atk1	0
Number of games won by extinction by Def1	0
Number of games dominated by Atk1 at turn *#200*	511
Number of games dominated by Def1 at turn *#200*	489

Table 11: Results of simulation 2

The third simulation includes 3 populations of defenders for one population of attackers. We will first study a scenario where each population has the same initial number of individuals. The populations of defenders differ by their *hp_proportion*, hence by their defense cost. In a single confrontation, defender 1 (see table 12) will have a defense cost of $4 * 30 = 120$, whereas defender 2

would have a defense cost of $16 * 30 = 480$. Defender 1 would then, win the confrontation against defender 2 but would be weaker against attackers.

Game		
	Number of Turns	100
	Number of Games	1000
Defender		
	Def1:	1000
	nb_rs	4
	hp_proportion	0.5
	hp_unit_cost	30
	Def2:	1000
	nb_rs	4
	hp_proportion	0.8
	hp_unit_cost	30
	Def3:	1000
	nb_rs	4
	hp_proportion	0.2
	hp_unit_cost	30
Attacker		
	Atk1:	1000
	attack_proba	1

Table 12: Input parameters of simulation 3

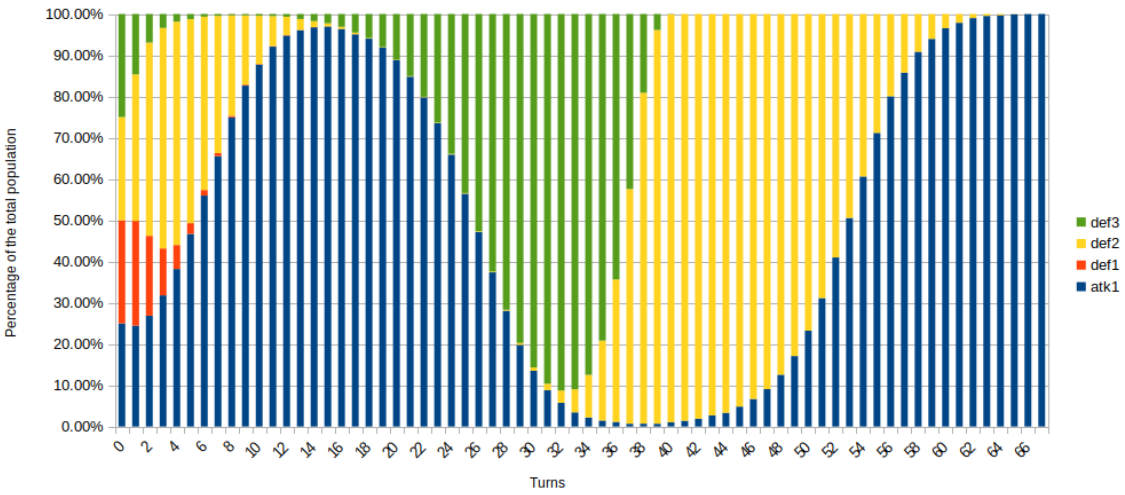

Figure 23: Evolution of the populations in simulation 3

As we observe in figure 23, populations vary drastically in the first turns. When attackers dominate, defenders with the highest honeypot proportion (*def3*, in green) tend to increase and

overcome the attackers population since they have a better chance of replication against them. However, when the population of *def3* increases, defenders with a lower honeypot proportion increase their population as a response since their overall defense cost is lower, hence, their chances of replication against *def3* are higher. In those cases, if the population of attackers has not gone extinct, their number of individuals can, again, increase against poorly equipped defenders. We observe from these results that populations that almost went extinct with less that 1% of the total number of individuals can significantly and rapidly increase their number if the main population of the game is a population against which they have high chances of replication. Attackers can take advantage of the confrontations between defenders and wait until there remains only one poorly equipped population of defenders which is easy to overcome. This is the situation we observe in figure 23. From these observations, we can wonder how the initial number of attackers can affect the final outcome of the game. We want to give equal chances of survival between the defenders, meaning the same initial number of individuals, however, the number of attackers can vary to simulate various contexts where we consider attacks as omnipresent or infrequent. The next test case keeps the same initial populations of defenders but makes vary the number of attackers in order to evaluate the model's behavior when, for a given defender, the initial most represented adversary is an attacker or another population of defenders.

Game		
	Number of Turns	100
	Number of Games	50*50
Defender		
	Def1:	1000
	nb_rs	4
	hp_proportion	0.5
	hp_unit_cost	30
	Def2:	1000
	nb_rs	4
	hp_proportion	0.8
	hp_unit_cost	30
	Def3:	1000
	nb_rs	4
	hp_proportion	0.2
	hp_unit_cost	30
Attacker		
	Atk1:	100-5000 by step 100
	attack_proba	0.75

Table 13: Input parameters of simulation 4

In order to evaluate the impact of the initial number of attackers, we make it vary between 100 and 5000 whereas the number of defenders remains constant and equal to 3000. For each step of 100 attackers, we perform 50 game simulations and we count the number of games won by attackers (or dominated by them at the final turn). Figure 24 displays the number of games won by attackers out of the 50 games played as a function of the initial number of attackers. We observe that the number of games won is not a linear function of the initial number of attackers. Variations occur

with 80% of the games lost when the initial number of attackers is equal to 800, corresponding to 21% of the total number of individuals. And on the contrary, almost all games are won when the initial attackers population is equal to 1400 individuals, corresponding to 32% of the total number of individuals in the game. Thanks to these results, we notice that conclusions cannot be drawn from a single test case where the initial populations are set. During further simulations, we will perform our evaluations with a varying initial number of attackers in order to have a more comprehensive vision of the possible outcomes of the game.

Figure 24: Number of games won by attackers vs. the initial number of attackers in simulation 4

The assessment of the basic behaviors of our practical game model allows us to identify the important features to consider in order to have a comprehensive evaluation of the results obtained. For example, by setting a maximum number of turns per game which is low, we might miss information about the actual winner of the game. The variation in games' outcomes when the initial population of attackers increases is also an important feature to take into account. It would be reductive to only consider a fixed number of attackers for every simulation. We need to observe results on a large set of input parameters to be able to provide a complete and undistorted interpretation.

6.4 Simulation results

In this subsection, we will introduce the theoretical results found thanks to the optimization algorithms detailed in section 5. The goal is to compute, prior to the simulation, a theoretically optimized honeypot proportion and honeypot unit cost, and to create a defender population that has these values as parameters. Then, simulations will be performed, confronting this theoretically optimized defender population against other populations of defenders and other populations of at-

tackers. Results will give us hints on the resilience of the theoretical optimal honeypot proportion applied to an evolutionary context where the players of the game interact differently from the game model which helped to develop the theoretical values. As a reminder, we do not want to prove the efficiency of our theoretical results but rather to assess its compliance to a different context where rationality is not considered.

Game			
		Number of Turns	200
		Number of Games	100*13
Defender			
	Def_theory:		1000
		nb_rs	5
		hp_proportion	0.5
		hp_unit_cost	50
	Def2:		1000
		nb_rs	5
		hp_proportion	0.2
		hp_unit_cost	50
	Def3:		1000
		nb_rs	5
		hp_proportion	0.7
		hp_unit_cost	50
	Def4:		1000
		nb_rs	5
		hp_proportion	0.9
		hp_unit_cost	50
Attacker			
	Atk1:		300-3900 by step 300
		attack_proba	0.8

Table 14: Input parameters of simulation 5

In order to introduce the theoretical honeypot distribution into our simulation, we first need to set the inputs of algorithm 1, meaning the maximum defense budget Δ, the attack payoff α, the attack cost γ, the honeypot reward μ and the maximum honeypot unit cost δ_{max}. To do so, we place ourselves in the role of a network administrator using our optimization algorithm: depending on our perception of costs and rewards, we give values that represent orders of magnitude. For example, we set the attack reward $\alpha = 1500$. Since in our case, other populations of defenders are equally dangerous as attackers, we decide to set the maximum defense budget to $\Delta = 1000$. We set a large value for Δ since we want to reach the equilibrium state shown in figure 19. The value of μ is equal to $\frac{1}{2}$ of the attack reward α. We also set a large value of μ compared to previous simulations in order to reach the equilibrium state shown in figure 19. The attack cost γ is still equal to $\frac{\alpha}{10}$. By applying these input parameters to algorithm 1, the output corresponding to the theoretical optimal honeypot proportion θ and the optimal honeypot unit cost δ_{HP} are equal to 50% and 50 respectively. This means that our optimized defender will possess 5 real servers, 5 honeypots, and a defense cost equal to $\delta = 250$.

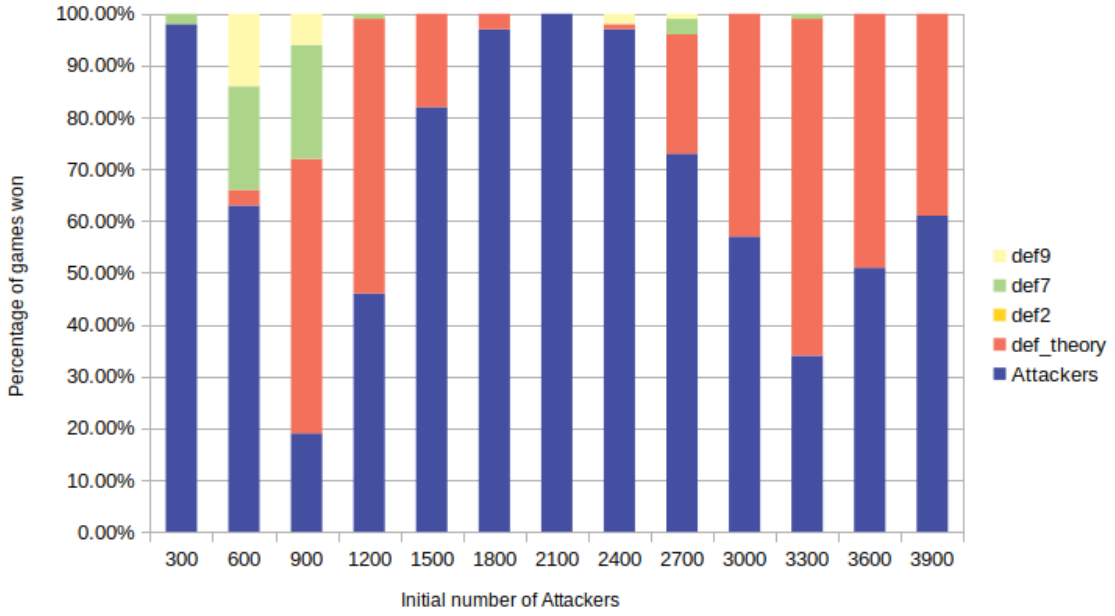

Figure 25: Percentage of games won per population vs. the initial number of attackers in simulation 5

Now that we have our theoretical defender's configuration, we need to test it against other populations of defenders and attackers in our evolutionary game model. With the results of the previous subsection, we observed that a fixed initial number of attackers leads to reductive outcomes, hence, we will perform our simulations with a varying initial number of attackers. We introduce three other populations of defenders (see table 14) corresponding to a very low honeypot proportion $hp\_proportion = 0.2$, a medium-high honeypot proportion $hp\_proportion = 0.7$ and a very high honeypot proportion $hp\_proportion = 0.9$. Every defender population has a honeypot unit cost equal to 50. The simulation goes as follows: we perform 100 games with the four populations of defenders and the one population of attackers by steps of 300 attackers. Hence, by steps of 300 attackers, all populations confront in 100 games and we count the number of games won out of the 100 games for each population. By doing so, we evaluate the ability of each population to win games with varying initial numbers of attackers.

Figure 25 shows the percentage of games won by each population as a function of the initial number of attackers. With input parameters detailed in table 14, the theoretically optimized configuration is a honeypot proportion equal to 50% and a honeypot unit cost of 50. This population is represented in figure 25 by *def_theory*. In figure 25, we observe that 65.8% of the total games were won by attackers. 26.9% of the total games were won by our theoretically optimized defenders and 3.8% of the total games were won by the defenders having a honeypot proportion equal to 0.7. Attackers population largely dominates the simulations, however the theoretically optimized defender population is shown to win the most games out of the 4 defender populations. The defender population with a honeypot proportion equal to 0.2 has been unable to win a single game in our simulation. Nevertheless, we cannot be fully satisfied with these results to assess the resilience of our theoretical results. Indeed, the choice of the input parameters Δ, α, μ and γ has an

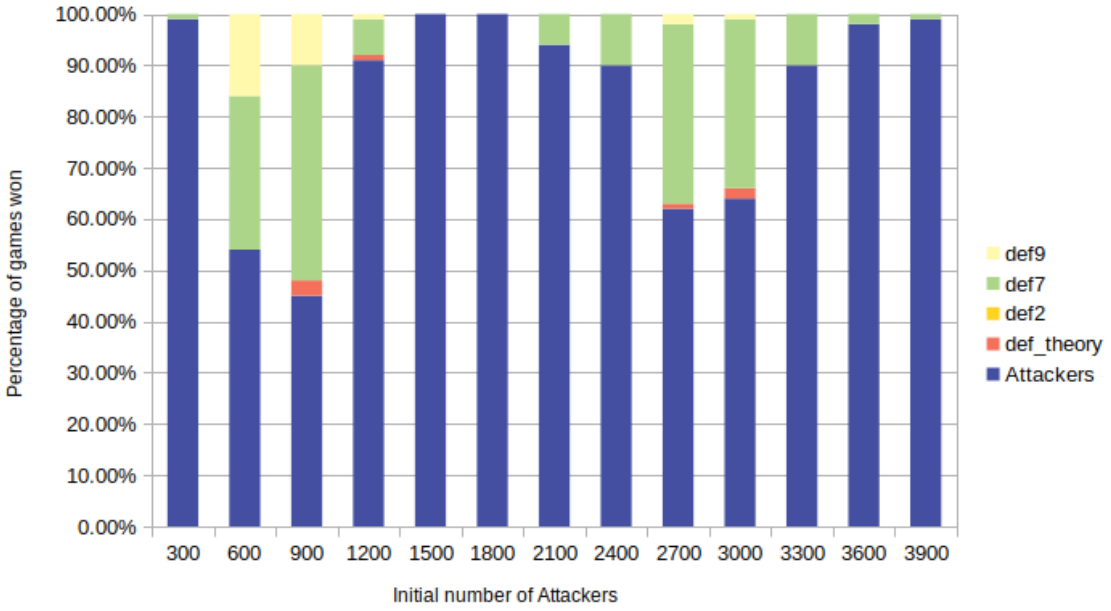

Figure 26: Percentage of games won per population vs. the initial number of attackers in simulation 6

important impact on the resulting theoretical honeypot proportion and is set by the user with no specific rules. We need introduce slight variations in these parameters to see if the overall outcome, meaning the number of games won, is strongly affected. To do so we set the input parameters $\alpha = 1000, \Delta = 1000, \mu = 500, \gamma = 100$. With these input parameters, the theoretically optimized honeypot proportion is equal to 0.45 and the honeypot unit cost is equal to 50. We perform the same simulations as previously with varying initial numbers of attackers and 100 games played per step between these attackers and 4 populations of defenders. We display the results in figure 26. From figure 26 we observe that attackers won 83.5% of the total number of games, defenders with a honeypot proportion equal to 0.7 won 13.6% of the games and our theoretically optimized defenders won only 0.5% of the games. As previously, attackers greatly dominate the simulations but our theoretically optimized defender population is no longer the most fitted defender population to survive. A slight variation in the input parameters triggers a great difference in the number of games won during our simulations. The impact of this irregularity will be discussed in the next subsection.

6.5 Discussion and conclusions

In this section, we have built an evolutionary-based game model in order to perform simulations representing encounters in a network between defenders, the network administrators and the attackers, the hackers. When building this game model, we used rules and principles that did not appear in the first non-cooperative game model developed. Indeed, in the evolutionary game model, strategies are represented by populations and there is no mixed outcome, when populations confront, they can either replicate, be destroyed or stay unchanged. In addition, we make several populations of defender compete, hence, there is no longer only one adversary for a defender. This game model

allows us to observe the relationships between the populations of the game. The algorithm developed in section 5 was not developed in order to fit this new model and obtain best results in a range of simulations. The initial motive of implementing the evolutionary game model was indeed to introduce the results obtained in our static game model into a new model where interactions between players and the basic rules of the game have changed. We wanted to see if the optimized honeypot proportion and the optimized honeypot unit cost obtained from our previous simulations were able to perform well in a simulation where players are no longer rational and where outcomes are definitive for a single player (replication or extinction).

From the results obtained in the previous subsection, we first observed very interesting outcomes where the theoretically optimized defender population was, out of the 4 defenders populations, the most fitted population to win games in our simulations. However, a slight variation in the input parameters of the algorithm leading to a honeypot proportion of 0.45 instead of 0.5 triggered completely different results where the theoretically optimized defender population was almost entirely dominated by other populations. This irregularity helps us to point out two limits of our work. First, the setting of parameters α, Δ, γ and μ is too subjective to ensure stable results. Indeed, these values are entirely set by the user, the network administrator, with no specific rule except a rule of thumb. They completely determine the outcome of the algorithm, meaning the optimal honeypot proportion and the honeypot unit cost. Even if we try to obtain a stable state by increasing the values of Δ and μ to simulate a defense with great resources (see section 4.3), it is difficult to set accurate values that reflect reality. The second limit of our work is related to the evaluation model. The evolutionary game model we developed to assess our algorithm results introduces many exterior variations such as the initial number of individuals in each population and the maximum number of turns played in a game. These variations greatly impact the outcomes of the games modeled, making it harder to assess the actual efficiency of our theoretical configurations. Indeed, the probability of winning for any population of defenders under some parameters (see figure 26) is almost null. Even if we observe the results under varying initial numbers of attackers, the impact of these exterior factors is too superior to draw useful and accurate conclusions on our theoretical results. Hence, as a conclusion to these evolutionary simulations, we cannot presume the efficiency or the inefficiency of the theoretical results found in the previous sections. The limitations encountered with the setting of the algorithm's parameters and the difficulty to separate the impact of the defender's configuration from exterior factors such as the initial number of individuals, prevented us from drawing scientifically valid conclusions.

7 Conclusion and future work

In this document, we used existing work on honeypot game theory in AMI networks to build a simplified version of a non-cooperative game model with incomplete information. We used this game model to extract equilibrium values for the attacker's strategies and we made assumptions to set the defender's dominant strategy. From these, we focused on the threshold value of θ, the proportion of honeypots in the network, in order to determine the situations where attackers will prefer to call off their attacks. To illustrate our theoretical results, we plotted simulations and we studied the impact of parameters' variation on the threshold value of θ. Our work allowed us to have a rough idea of sufficient proportions of honeypots to prevent most attacks. Details are provided with the introduction of cost and efficiency constraints. We set a limit in the defender's resources with a maximum defense budget Δ and a honeypot unit deployment cost δ. We also added an efficiency constraint on the honeypots by decreasing their ability to lure attackers with increasing number of honeypots in the network. These constraints prevented aberrant results where defenders deployed thousands of identical honeypots and allowed to build a simulation providing a comprehensive comparison of all the defender's configurations (honeypot proportion and honeypot unit cost). These results represent useful information for potential network administrators however, the assumptions made concerning the rationality of players and their knowledge about other players payoffs constitute an important limitation for real-life applications. We were unable to claim that these configurations would show their effectiveness in real systems. In order to provide additional information and assess the resilience of our theoretically optimized configurations, we developed an evolutionary based game model where attackers and various defender configurations were represented by populations of players confronting each other during several generations. We proposed an implementation of this game model in Python and we performed several simulations in order to see if the theoretically optimized defender configuration performed better in survival with rules different from the initial game model. Our goal was to overcome the limitation due to the rationality of players with attackers randomly attacking devices. Nevertheless, new limitations were observed. First, the difficulty to choose accurate initial parameters for the algorithm returning the optimal configuration. Second, the difference in functioning between the static game model and the evolutionary game model that introduced major variations preventing us from faultlessly evaluating the role of the honeypot proportion in the survival of defender populations.

With our work, we realized that the gap between theory and practice in the deployment of honeypots in an AMI network is consequent. Game theory can provide hints about the optimal strategies and configurations to adopt, but in practice, criteria such as the human factor, weakened by social engineering, and the knowledge and resources of potential hackers, are extremely difficult to model. We believe that any theory extracted from game models would be less efficient than practical penetration tests or red team vulnerability assessments on large-size testbeds. We reckon that more scientific publications on game theory applied to the deployment of honeypots in Smart Grid networks won't necessarily bring useful contributions to the research field. However, there is an important opening for the implementation of honeypots such as Conpot [20], or research on the optimal configuration of honeypots for energy systems actually used in Smart Grid networks [14].

References

[1] John Nash. "Non-cooperative games". In: *Annals of mathematics* (1951), pp. 286–295.

[2] Ramyar Rashed Mohassel et al. "A survey on Advanced Metering Infrastructure". In: *International Journal of Electrical Power & Energy Systems* 63 (Dec. 2014), pp. 473–484. DOI: 10.1016/j.ijepes.2014.06.025.

[3] Aaron Hansen, Jason Staggs, and Sujeet Shenoi. "Security analysis of an advanced metering infrastructure". In: *International Journal of Critical Infrastructure Protection* 18 (2017), pp. 3–19. ISSN: 1874-5482. DOI: https://doi.org/10.1016/j.ijcip.2017.03.004. URL: http://www.sciencedirect.com/science/article/pii/S1874548217300495.

[4] G. Dileep. "A survey on smart grid technologies and applications". In: *Renewable Energy* 146 (2020), pp. 2589–2625. ISSN: 0960-1481. DOI: https://doi.org/10.1016/j.renene.2019.08.092. URL: http://www.sciencedirect.com/science/article/pii/S0960148119312790.

[5] Durgadevi Karthikeyan and Ganeshkumar Pugalendhi. "An effective trust based defense mechanism to thwart malicious attack in smart grid communication network". In: Mar. 2017, pp. 1–9. DOI: 10.1109/ITCOSP.2017.8303102.

[6] Igor C.G. Ribeiro et al. "THOR: A framework to build an advanced metering infrastructure resilient to DAP failures in smart grids". In: *Future Generation Computer Systems* 99 (2019), pp. 11–26. ISSN: 0167-739X. DOI: https://doi.org/10.1016/j.future.2019.03.021. URL: http://www.sciencedirect.com/science/article/pii/S0167739X18326268.

[7] The Smart Grid Interoperability Panel - Smart Grid Cybersecurity Committee. "Guidelines for Smart Grid Cybersecurity". In: *CSRC — NIST* (Sept. 2014). DOI: 10.6028/NIST.IR.7628r1.

[8] *IEC 61850 - Wikipedia*. [Online; accessed 27. Dec. 2019]. Dec. 2019. URL: https://en.wikipedia.org/wiki/IEC_61850.

[9] CEN-CENELEC-ETSI Smart Grid Coordination Group. "Smart Grid Reference Architecture". In: (2012).

[10] Ye Yan et al. "A Survey on Cyber Security for Smart Grid Communications". In: *Communications Surveys & Tutorials, IEEE* 14 (Jan. 2012), pp. 998–1010. DOI: 10.1109/SURV.2012.010912.00035.

[11] Zakaria El Mrabet et al. "Cyber-security in smart grid: Survey and challenges". In: *Computers & Electrical Engineering* 67 (2018), pp. 469–482.

[12] Henry Mwiki et al. "Analysis and triage of advanced hacking groups targeting western countries critical national infrastructure: APT28, RED October, and Regin". In: *Critical Infrastructure Security and Resilience*. Springer, 2019, pp. 221–244.

[13] Jacob Sakhnini et al. "Security aspects of Internet of Things aided smart grids: A bibliometric survey". In: *Internet of Things* (2019), p. 100111. ISSN: 2542-6605. DOI: https://doi.org/10.1016/j.iot.2019.100111. URL: http://www.sciencedirect.com/science/article/pii/S2542660519302148.

[14] Christos Dalamagkas et al. "A Survey On Honeypots, Honeynets And Their Applications On Smart Grid". In: June 2019. DOI: 10.1109/NETSOFT.2019.8806693.

[15] Kun Wang et al. "Strategic Honeypot Game Model for Distributed Denial of Service Attacks in the Smart Grid". In: *IEEE Transactions on Smart Grid* PP (Feb. 2017), pp. 1–1. DOI: 10.1109/TSG.2017.2670144.

[16] Wen Tian et al. “Defense Strategies Against Network Attacks in Cyber-Physical Systems with Analysis Cost Constraint Based on Honeypot Game Model”. In: *Computers, Materials & Continua* 58 (Jan. 2019), pp. 193–211. DOI: 10.32604/cmc.2019.05290.

[17] Nadia Boumkheld et al. “Honeypot Type Selection Games for Smart Grid Networks”. In: Sept. 2019.

[18] David H. Reiley, Michael B. Urbancic, and Mark Walker. “Stripped-Down Poker: A Classroom Game with Signaling and Bluffing”. In: *The Journal of Economic Education* 39.4 (2008), pp. 323–341. DOI: 10.3200/JECE.39.4.323-341. eprint: https://doi.org/10.3200/JECE.39.4.323-341. URL: https://doi.org/10.3200/JECE.39.4.323-341.

[19] George J Mailath. “Do people play Nash equilibrium? Lessons from evolutionary game theory”. In: *Journal of Economic Literature* 36.3 (1998), pp. 1347–1374.

[20] Arthur Jicha and Mark Patton. “SCADA honeypots: An in-depth analysis of Conpot”. In: Sept. 2016, pp. 196–198. DOI: 10.1109/ISI.2016.7745468.

8 Annex 1 - Equilibrium analysis of section 5.2

		Attacker	
		Λ_1	Λ_2
Defender	(Ω_1, Ω_1)	$\theta(P\mu - (1-P)\mu - \delta) + (1-\theta)(-\alpha)$, $\theta(-P\mu + (1-P)\mu - \gamma) + (1-\theta)(\alpha - \gamma)$	$\theta(-\delta) + (1-\theta)\beta$, 0
	(Ω_1, Ω_2)	$P\mu - (1-P)\mu - \theta\delta + (1-\theta)(-\beta)$, $-P\mu + (1-P)\mu - \gamma$	$\theta(-\delta) + (1-\theta)(-\beta)$, 0
	(Ω_2, Ω_1)	$-\alpha - \theta\delta$, $\alpha - \gamma$	$\theta(-\delta) + (1-\theta)\beta$, 0
	(Ω_2, Ω_2)	$\theta(-\alpha - \delta) + (1-\theta)(-\beta + P\mu - (1-P)\mu)$, $\theta(\alpha - \gamma) + (1-\theta)(-P\mu + (1-P)\mu - \gamma)$	$\theta(-\delta) + (1-\theta)(-\beta)$, 0

Table 15: Game model including efficiency constraint payoff matrix

When the attacker launches the attack $\{\Lambda_1\}$:

$$U_D((\Omega_1, \Omega_1)|\Lambda_1) - U_D((\Omega_1, \Omega_2)|\Lambda_1) = (1-\theta)(\beta - \alpha - P\mu + (1-P)\mu) \tag{17}$$

$$U_D((\Omega_1, \Omega_1)|\Lambda_1) - U_D((\Omega_2, \Omega_1)|\Lambda_1) = \theta(\alpha + P\mu - (1-P)\mu) > 0 \qquad (\alpha > \mu) \tag{18}$$

$$U_D((\Omega_1, \Omega_1)|\Lambda_1) - U_D((\Omega_2, \Omega_2)|\Lambda_1) = \theta(\alpha + P\mu - (1-P)\mu) + (1-\theta)(\beta - \alpha - P\mu + (1-P)\mu) \tag{19}$$

$$U_D((\Omega_1, \Omega_2)|\Lambda_1) - U_D((\Omega_2, \Omega_1)|\Lambda_1) = \alpha + P\mu - (1-P)\mu + (1-\theta)(-\beta) \tag{20}$$

$$U_D((\Omega_1, \Omega_2)|\Lambda_1) - U_D((\Omega_2, \Omega_2)|\Lambda_1) = \theta(\alpha + P\mu - (1-P)\mu) > 0 \quad (\alpha > \mu) \tag{21}$$

$$U_D((\Omega_2, \Omega_1)|\Lambda_1) - U_D((\Omega_2, \Omega_2)|\Lambda_1) = (1-\theta)(\beta - \alpha - P\mu + (1-P)\mu) \tag{22}$$

When the attacker does not launch the attack $\{\Lambda_2\}$:

$$U_D((\Omega_1, \Omega_1)|\Lambda_2) - U_D((\Omega_1, \Omega_2)|\Lambda_2) = 2\beta(1-\theta) > 0 \tag{23}$$

$$U_D((\Omega_1, \Omega_1)|\Lambda_2) - U_D((\Omega_2, \Omega_1)|\Lambda_2) = 0 \tag{24}$$

$$U_D((\Omega_1,\Omega_1)|\Lambda_2) - U_D((\Omega_2,\Omega_2)|\Lambda_2) = 2\beta(1-\theta) > 0 \tag{25}$$

$$U_D((\Omega_1,\Omega_2)|\Lambda_2) - U_D((\Omega_2,\Omega_1)|\Lambda_2) = -2\beta(1-\theta) < 0 \tag{26}$$

$$U_D((\Omega_1,\Omega_2)|\Lambda_2) - U_D((\Omega_2,\Omega_2)|\Lambda_2) = 0 \tag{27}$$

$$U_D((\Omega_2,\Omega_1)|\Lambda_2) - U_D((\Omega_2,\Omega_2)|\Lambda_2) = 2\beta(1-\theta) > 0 \tag{28}$$

From equations 18 and 24 we observe that (Ω_2,Ω_1) is dominated by (Ω_1,Ω_1). Similarly, from equations 21 and 27 we observe that (Ω_2,Ω_2) is dominated by (Ω_1,Ω_2).

Now if we study mixed strategies : Let r the probability that D chooses strategy (Ω_1,Ω_1) (the defender provides services for both entities). $(1-r)$ is the probability of the defender choosing strategy (Ω_1,Ω_2)
Let q the probability that A chooses strategy Λ_1 (the attacker launches the attack).

$$\begin{aligned}
&r[\theta(-P\mu+(1-P)\mu-\gamma)+(1-\theta)(\alpha-\gamma)]+(1-r)[-P\mu+(1-P)\mu-\gamma]\\
=&0r+0(1-r)\\
\Rightarrow&r(-P\mu+(1-P)\mu-\gamma)(\theta-1)+r(1-\theta)(\alpha-\gamma)-P\mu+(1-P)\mu-\gamma=0\\
\Rightarrow&r=\frac{P\mu-(1-P)\mu+\gamma}{(1-\theta)(P\mu-(1-P)\mu+\alpha)}
\end{aligned} \tag{29}$$

$$\begin{aligned}
&q[\theta(P\mu-(1-P)\mu-\delta)+(1-\theta)(-\alpha)]+(1-q)[\theta(-\delta)+(1-\theta)\beta]\\
=&q[P\mu-(1-P)\mu-\theta\delta+(1-\theta)(-\beta)]+(1-q)[\theta(-\delta)+(1-\theta)(-\beta)]=0\\
\Rightarrow&q(1-\theta)(-\beta-\alpha-P\mu+(1-P)\mu)+2\beta=0\\
\Rightarrow&q=\frac{2\beta}{\beta+\alpha+P\mu-(1-P)\mu}
\end{aligned} \tag{30}$$

We observe from equation 30 that to have an equilibrium, we need : $\beta < \alpha + P\mu - (1-P)\mu$, otherwise, there is no BNE. This condition means that it is preferable for the defender D to stop providing services in order to avoid the damages caused by an attack. This rational behaviour does not fit reality since the goal of the defender (network administrator) is to provide services to legitimate users. The defender cannot simply shut services down in order to avoid attacks. We will assume that $\beta > \alpha + \mu$ in order to prevent the defender to stop providing services for real servers. Strategy $\{(\Omega_1,\Omega_1)\}$ becomes the dominant strategy for the defender.