

Game Theory and Intelligent Orchestration: The honeypots case study

BIG DATA AND INTELLIGENT APPLICATIONS

UNIVERSITY OF WESTERN MACEDONIA

CHRISTOS DALAMAGKAS – CDALAMAGKAS@UOWM.GR

PROF. PANAGIOTIS SARIGIANNIDIS – PSARIGIANNIDIS@UOWM.GR



Outline

- Introduction to Honeypots
 - Purpose
 - Classification
 - Frameworks
- Introduction to Game Theory
- The Honeypots Case Study – One-shoot game
- The maxmin approach
- Results and discussion



Honeypots

- Definition:
 - “An information system resource (e.g., a PC, a server, etc) whose values lies in unauthorized or illicit use of that resource” (Lance Spitzner - 2002)
- Usage:
 - Act as decoy to lure cyberattackers
- Goals:
 - Gather intelligence about current cyber threats → Feedback for improving the cyber security strategy of the organization
 - Consume attacker’s resources
 - Mislead them from targeting the real infrastructure



Honeypots Classification

- Based on Physicality:
 - **Virtual** → e.g., virtual machine
 - **Physical** → e.g., a real device that is used only as honeypot
- Based on Operation Field:
 - **Production** → Deployed in the operational environment to attract malicious insiders
 - **Research** → Deployed publicly on the Internet to attract cyberattackers or bots
- Based on Role:
 - **Server** → Waits for connections
 - **Client** → Actively searches for malicious servers that aim to lure clients
 - **Hybrid** → Combination of the above
- Based on Interaction:
 - **Low** → Implements basic functionalities and can easily be detected, but cheaper
 - **High** → Fully functional systems, realistic but expensive to implement and maintain

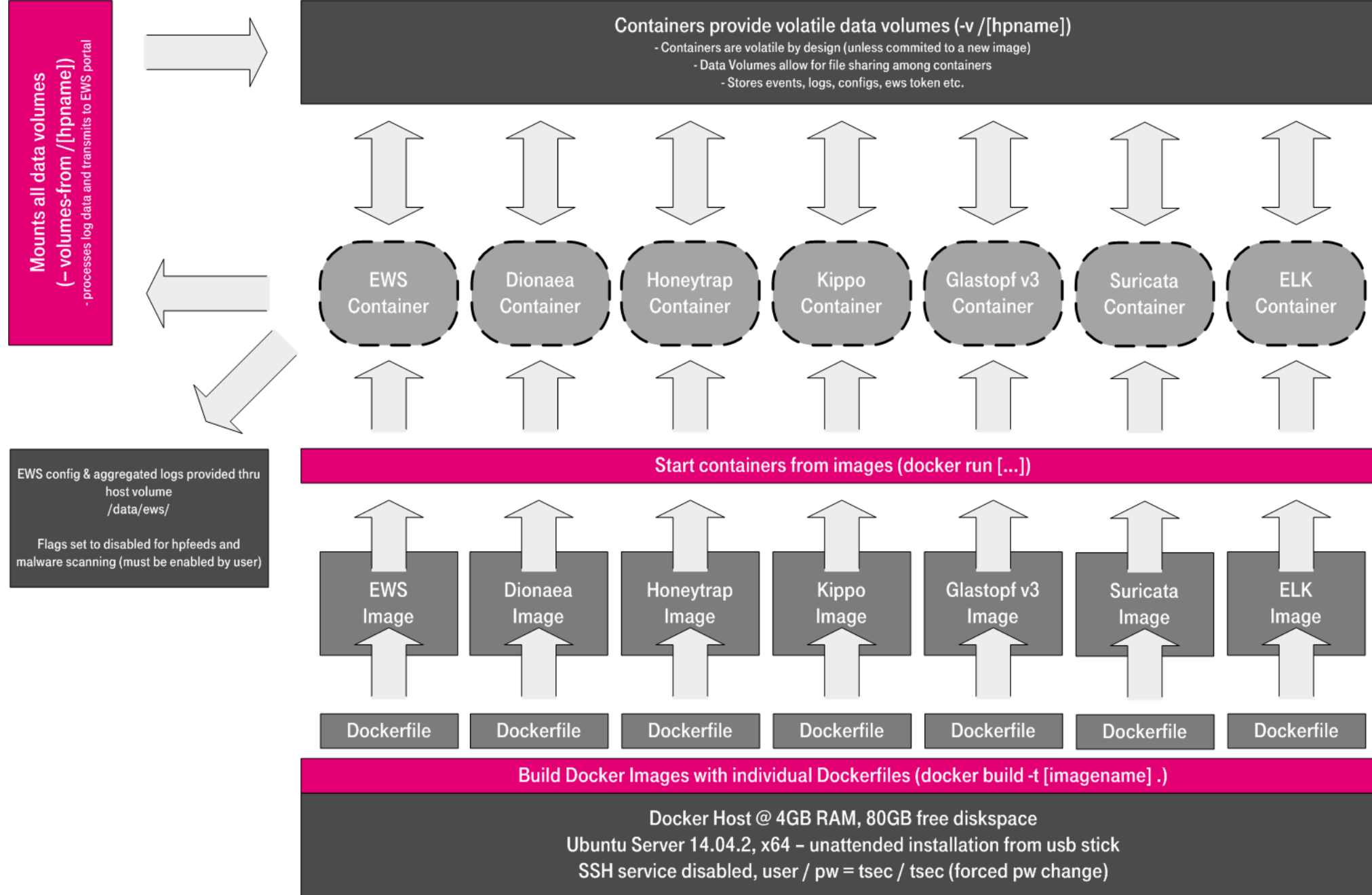


Honeypot Frameworks

- **Conpot**
 - Focuses on industrial protocols (Modbus TCP, S7Comm, BACnet, IPMI, EN/IP), also supports HTTP, SNMP, (T)FTP
- **Dionea**
 - SSH Honeypot server
- **T-Pot** by Deutsche Telekom
 - Honeypot suite, composed of multiple honeypot frameworks
 - A virtual machine that hosts multiple honeypots as docker containers
 - A dockerfile for each honeypot



The T-Pot Framework



Additional honeypot terminology

- **Honeynets:**
 - A network of honeypots
 - “A network of high interaction honeypots that simulates a production network and configured such that all activity is monitored, recorded and in a degree, discreetly regulated” (Lance Spitzner – Honeynet Project)
- **Honeyfarm**
 - A centralized collection of honeypots and analysis tools



Game Theory (GT) basics

- **Definition:**
 - An interdisciplinary field that studies methods for optimal decision making, considering the decisions of competitors
 - A methodology for modelling and formulating decision-making problems
 - Applications in financials, politics, psychology, sociology, evolutionally biology, computer science
- A game theory model is composed of:
 - Players
 - Strategies: Actions that are available for each player
 - Information about other player's strategies/actions
 - Payoff or Utilities: Rules that describe the benefit or damage of a player, based on the action that they choose. Objective is to maximise the payoff/utility
- GT objective: Find the variables (actions) that maximize the payoff function

Payoff: Function representing the benefit (positive or negative) of a strategy, considering the strategies chosen by the rest players
Utility: Function representing the benefit (positive or negative) of a strategy, regardless of the strategies chosen by the rest players



Game Categories (1/2)

- Cooperative vs Non-cooperative
 - Cooperative → Players may form teams and exchange information in order to improve their payoff
 - Non-Cooperative → Each player works on their own
- Constant Sum vs Zero Sum
 - Constant → The sum of the payoff enjoyed by all players is constant
 - Zero Sum → The sum of the payoff enjoyed by all players is always zero. This means that the benefit of a player results to an equal damage for another player
- Finite or Infinite
 - Finite → The player can choose between a range of discrete strategies
 - Infinite → There is infinite number of strategies to be chosen
- Stackelberg games: One player (leader) moves first and all other players (followers) move after them



Depending on the game category, we know how to solve it

Game Categories (2/2)

- Simultaneous vs Sequential Moves:
 - Simultaneous → The players choose their strategy at the same time
 - Sequential → Each player waits for their round
- Mixed or Pure Strategy
 - Pure → Players chose their strategy in a deterministic manner
 - Mixed → Players choose their strategy based on probabilities (stochastic)
- Perfect vs Imperfect Information
 - Perfect → The player has perfect information about the strategies and payoffs of the other players
 - Imperfect → The player has incomplete information about the strategies and payoffs of the other players. Usually, the player tries to estimate the payoff function based on probabilities (**Bayesian game**)
- There are more categories



Solutions in GT models

- Ultimate goal is to determine the optimal action for each player
- How to accomplish that?
 - By maximizing the player's utility → Finding those values (actions) that maximize the player's utility
 - Optimization problem: Find the best solution of a function, out of all feasible solutions
- Categories of optimization problems to solve:
 - **Dominant strategy solution:**

A strategy s_i is dominant over the alternative $s'_i \in S$, if it satisfies:

$$P_i(s_i, s'_{-i}) \geq P_i(s'_i, s'_{-i})$$

In other words, the utility of a player is the maximum possible, regardless of the strategies chosen by other players.



Solutions in GT models

- Categories of optimization problems to solve:

- **Nash Equilibrium (NE)**

Its not always possible to have dominant solution. A Nash solution is a set of strategies that are obtained if for every player i and for each strategy $s'_i \in S_i$:

$$P_i(s_i, \mathbf{s}_{-i}) \geq P_i(s'_i, \mathbf{s}_{-i})$$

A solution is called Nash solution, if no player i has the incentive to change their strategy from s_i to s'_i , thus improving their benefit, assuming that all other players stick to the strategies they have chosen in \mathbf{s} .

Attributes of NE:

- A dominant solution is also Nash solution, but a Nash solution is not always dominant solution.
 - Each player is rational and aims to maximise their benefit, thus the solution can be predictable.
 - Each game with finite strategies and finite players has at least one NE solution.



Is NE always the best solution?

- NE is a rational outcome, considering a finite, non-cooperative game and the fact that each player is “selfish”
- But its always the best possible solution?
Let's take the prisoner's dilemma:
 - **Two members of a criminal gang** are arrested and imprisoned. Each prisoner is in solitary confinement with **no means of communicating with the other**. The prosecutors lack sufficient evidence to convict the pair on the principal charge, but they have enough to convict both on a lesser charge. Simultaneously, the prosecutors offer each prisoner a bargain. **Each prisoner is given the opportunity either to betray the other by testifying that the other committed the crime, or to cooperate with the other by remaining silent**



Prisoner's dilemma outcomes

	B betrays	B stays silent
A betrays	(-3, -3)	(0, -4)
A stays silent	(-4, 0)	(-1, -1)

- Since they are not cooperating, a rational outcome is for both to betray
- However, this is not the optimal solution!
 - If they chose to stay both silent, then the outcome would be more beneficial for both of them
- NE is insufficient in this case; This is measured by calculating the **social welfare**, i.e., the sum of all payoffs: $W(s) = \sum_{i \in N} U_i(s_i)$
- Price of Anarchy \rightarrow What is the cost for the players being selfish? $POA = \frac{\max_s W(s)}{\min_s W(s)}$
- Social cost is the opposite of social welfare

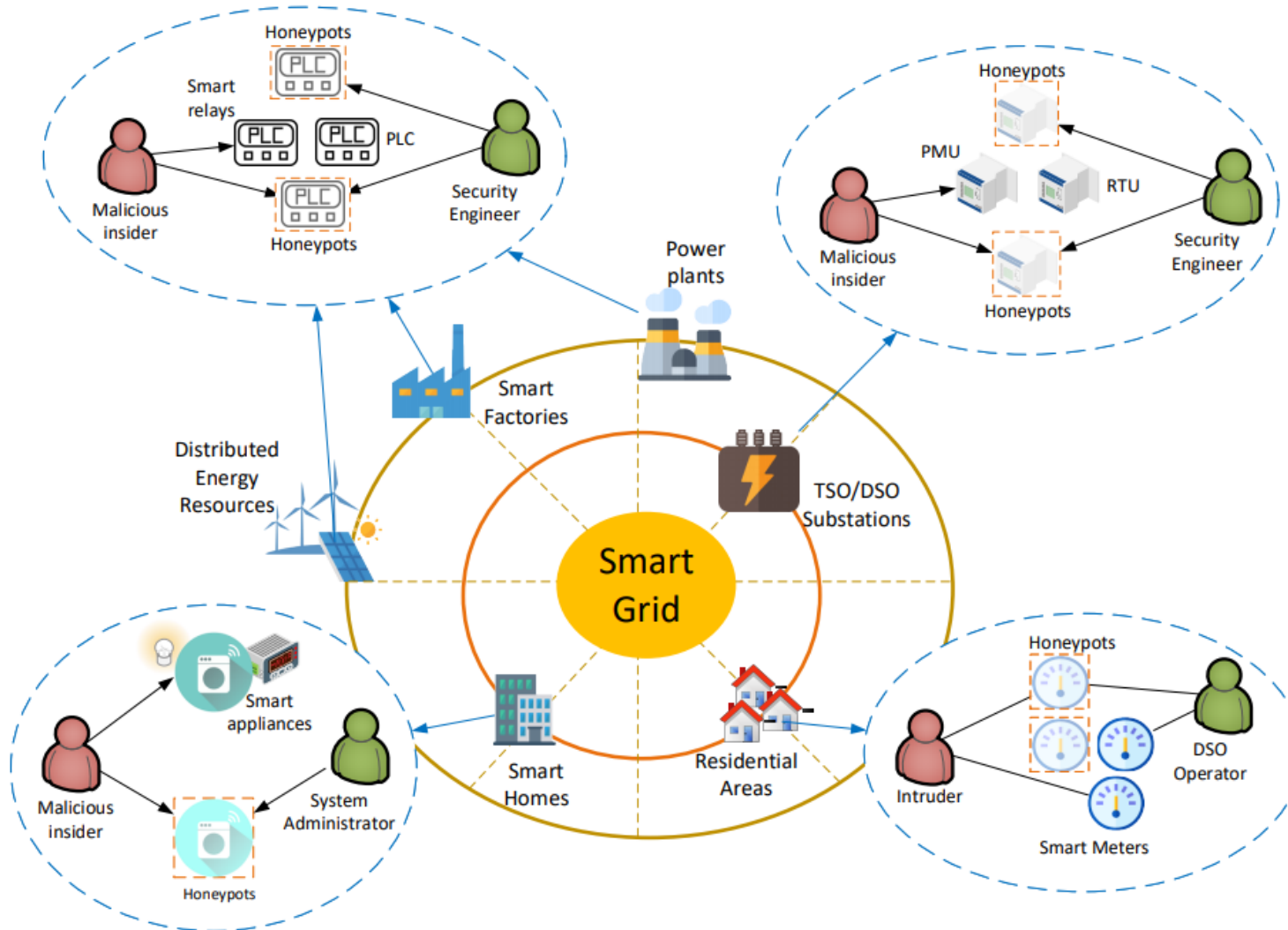


The Honeypots Case Study

- How many honeypots should the organization choose to deploy?
 - Large number of honeypots waste computing resources
 - Small number of honeypots is not adequate to capture attackers or to be detected, thus the risk is increased
- Game theory can help us take a decision:
 - We have contradicting interests; the defender wants to receive attackers at honeypots and the attacker wants to target as many real devices as possible
 - The number of honeypots should be minimized so that the interest of the defender is maximized, while the attacker takes always the right decisions
- Game players:
 - Defender → The system administrator of the organization under attack
 - Attacker → Malicious insider or cyberattacker over the Internet that aims at the assets of the defender



Smart Grid Honeypots



- Example of honeypots location in a smart grid environment
- A honeypot could be located in:
 - Smart home
 - Smart meters of residential areas
 - In digitalized substations
 - In smart factories
 - In Renewables plants



Game Rules

- Defender has limited number of IP addresses (N_{max}), which can be assigned either to operational devices or honeypots
- N is the sum of **connected** real devices and honeypots, N_r are the real devices
- Available strategies:
 - $s_{A,i} \in \{1,0\}$. The strategy of the attacker for the i-th host: to attack the i-th host (1) or not to attack (0)
 - $s_{D,i} \in \{-1, 1\}$. The strategy of the defender for the i-th host – use the i-th host as real device (1) or as honeypot (-1)
- a_i, d_i represent constant preferences (weights) from the attacker and defender perspective
- θ is the portion of N that are honeypots
- ϕ is the portion of N that is being attacked
- ϕ_m is the maximum portion of N that can be attacked (usually 1 / 100%)
- All hosts have the same probability of being honeypots or being attacked



Initial Utility Functions

$s_{A,i} \in \{1,0\}$. The strategy of the attacker for the i-th host: to attack the i-th host (1) or not to attack (0)

$s_{D,i} \in \{-1, 1\}$. The strategy of the defender for the i-th host
– use the i-th host as real device (1) or as honeypot (-1)

$$U_A[t] = \overset{\text{benefit for attacking real devices}}{a_1 \sum_{i=1}^N \frac{(1 + s_{D,i})}{2} s_{A,i}} - \overset{\text{damage for attacking honeypots}}{a_2 \sum_{i=1}^N \frac{1 - s_{D,i}}{2} s_{A,i}} - \overset{\text{damage for total number of attacks}}{a_3 \sum_{i=1}^N s_{A,i}}$$

$$U_D[t] = \overset{\text{benefit for honeypots being attacked}}{d_1 \sum_{i=1}^N \frac{(1 - s_{D,i})}{2} s_{A,i}} - \overset{\text{damage for real devices being attacked}}{d_2 \sum_{i=1}^N \frac{(1 + s_{D,i})}{2} s_{A,i}} - \overset{\text{damage for real devices not served}}{d_3 \left(\sum_{i=1}^N \frac{(1 + s_{D,i})}{2} - N_r \right)^2} - \overset{\text{damage for the total number of hosts}}{d_4 N}$$



θ is the portion of N that are honeypots
 ϕ is the portion of N that is being attacked

One-Shot Game Definition

- By converting the combinatorial problem to a continuous one:

$$U_A = a_1(1 - \theta)N\phi - a_2\theta\phi N - a_3\phi N$$

$$U_D = d_1\theta\phi N - d_2(1 - \theta)N\phi - d_3((1 - \theta)N - N_r)^2 - d_4N$$

- Set of actions for defender: $A_D = \{\theta \in [0,1], N \in [0, N_{max}]\}$
- Set of actions for attacker: $A_A = \phi \in [0, \phi_m]$
- Our goal is to determine the proper action of the defender (i.e., the number of honeypots θN) so that their utility is maximized
- We choose NE as the solution to this game
 - Note that if we choose NE, then we assume that the attacker weights (a_1, a_2, a_3) are known



One-Shot Game – NE Definition

- The action profile (θ^*, ϕ^*, N^*) is a NE if by deviating from it none of the players can gain anything, i.e.,

$$\begin{aligned}U_D(\theta^*, \phi^*, N^*) &\geq U_D(\theta, N, \phi^*) \\U_A(\theta^*, \phi^*, N^*) &\geq U_A(\theta^*, N^*, \phi)\end{aligned}$$

- Thus, a strategy of each player belongs to the NE if this is a best reply to the strategy of the other player
- To decrease solution candidates, we are going to proof that If the NE exists, then $\phi^* \in \{0, \phi_m\}$

Proof: Let's assume that the set $(\theta^*, N^* \neq 0, \phi')$ is a NE and that $\phi' \in \{0, \phi_m\}$. Then, it holds that:

$$(a_1(1 - \theta)N - a_2\theta N - a_3N)\phi' \geq (a_1(1 - \theta) - a_2\theta - a_3)\phi_m$$

This results to $\phi' \geq \phi_m$, which contradicts the assumption.



NE of the One-Shoot Game

$$(\theta^*, N^*, \phi^*) = \begin{cases} (0, \frac{2d_3 N_r - d_4}{2d_3}, 0), & \text{if} \\ & 0 \leq \frac{2d_3 N_r - d_4}{2d_3} \leq N_{\max} \text{ and } a_1 \leq a_3, \\ (0, N_{\max}, 0), & \text{if} \\ & \frac{2d_3 N_r - d_4}{2d_3} > N_{\max} \text{ and } a_1 \leq a_3, \\ (0, 0, 0), & \text{if} \\ & \frac{2d_3 N_r - d_4}{2d_3} < 0 \text{ and } a_1 \leq a_3, \\ (\frac{(d_1 + d_2)\phi_m + 2d_3 N_{\max} - 2d_3 N_r}{2d_3 N_{\max}}, N_{\max}, \phi_m), & \text{if} \\ & 0 \leq \frac{(d_1 + d_2)\phi_m + 2d_3 N_{\max} - 2d_3 N_r}{2d_3} \leq N_{\max} \text{ and } d_1 \phi_m \geq d_4 \\ & \text{and } (a_1 + a_2)N_r \geq (a_2 + a_3)N_{\max} + \frac{(a_1 + a_2)(d_1 + d_2)}{2d_3}, \\ (0, N_r - \frac{d_2 \phi_m + d_4}{2d_3}, \phi_m), & \text{if} \\ & d_1 \phi_m < d_4 \text{ and } a_1 > a_3 \text{ and } 0 < N_r - \frac{d_2 \phi_m + d_4}{2d_3} \leq N_{\max}, \\ (0, N_{\max}, \phi_m), & \text{if} \\ & (d_1 + d_2)\phi_m + 2d_3 N_{\max} - 2d_3 N_r < 0 \text{ and} \\ & a_1 > a_3 \text{ and } N_r - \frac{d_2 \phi_m + d_4}{2d_3} > N_{\max}, \\ (0, 0, \phi_m), & \text{if} \\ & (d_1 + d_2)\phi_m + 2d_3 N_{\max} - 2d_3 N_r < 0 \text{ and} \\ & a_1 > a_3 \text{ and } N_r - \frac{d_2 \phi_m + d_4}{2d_3} < 0, \\ \nexists, & \text{elsewhere.} \end{cases}$$

- The possible NEs are derived by setting optimal values for ϕ or N and solving the derivatives $U'_D = 0$ and $U'_A = 0$ respectively.
- For specific problem parameters, the NE (is exists):
 - Is unique
 - Is located at the global maximum
- The fourth branch is the most appropriate solution, since:
 - $N = N_{\max}$
 - $\phi = \phi_m$
 - $0 < \theta < 1$



What if NE does not exist?

- The user should give predefined values for d (defender's weights) and a (attacker's weights)
 - If a, d satisfy the conditions of branch 4, then the Nash solution can be obtained by calculating the value of θ^* in branch 4
 - If a, d cannot satisfy branch 4, then the solution would be not to deploy any honeypots, or there is no solution at all
- If the NE does not exist, a maxmin analysis can be applied
 - Instead of relying on predictions, the concern is to maximize the lowest value the other player can for the player to receive when they know the player's action.

$$\max_{\substack{0 \leq \theta \leq 1, \\ 0 \leq N \leq N_{max}}} \min_{0 \leq \phi \leq \phi_m} U_D$$

- We observe that U_D is proportional to ϕ , thus the attacker can force U_D to the lowest values by choosing either ϕ_m or 0.



convex = only one optimal solution which is globally optimized

The maxmin optimization problem


- When $\phi = \phi_m$

$$U_D = d_1\theta\phi_m N - d_2(1 - \theta)\phi_m N - d_3((1 - \theta)N - N_r)^2 - d_4N$$

- When $\phi = 0$

$$U_D = -d_3((1 - \theta)N - N_r)^2 - d_4N$$

- Thus, the optimization problem is formulated as follows:

$\begin{aligned} \max_{\theta, N} \quad & y \\ \text{s.t.} \quad & C_1 : 0 \leq \theta \leq 1, \\ & C_2 : 0 \leq N \leq N_{\max}, \\ & C_3 : d_1\theta\phi_m N - d_2(1 - \theta)\phi_m N - d_3((1 - \theta)N - N_r)^2 - d_4N \geq y, \\ & C_4 : -d_3((1 - \theta)N - N_r)^2 - d_4N \geq y. \end{aligned}$	$\begin{aligned} \theta N &= N_1 \\ (1 - \theta)N &= N_2 \end{aligned}$		$\begin{aligned} \max_{N_1, N_2} \quad & y \\ \text{s.t.} \quad & C_1 : N_1 + N_2 \leq N_{\max}, \\ & C_2 : d_1\phi_m N_1 - d_2\phi_m N_2 - d_3(N_2 - N_r)^2 - d_4(N_1 + N_2) \geq y, \\ & C_3 : -d_3(N_2 - N_r)^2 - d_4(N_1 + N_2) \geq y, \\ & C_4 : N_1, N_2 \geq 0. \end{aligned}$	<p>non-convex</p> <p>convex</p>
---	---	---	--	---



Solving convex problems in Python

- If we calculate N_1, N_2 , then we can retrieve N, θ
- Variables are N_1, N_2, y
 - y is the optimization variable
- The constraints are given as a Python array
- `cvxpy.Maximize()` returns the objective function
- `cvxpy.Problem()` constructs the problem in respect to the constraints
- Finally, `solve()` does the magic, and returns N_1 and N_2

```
import cvxpy as cp

# Max-min algorithm to determine optimal n, theta if Nash equilibrium does not exist
def maxmin():
    n1 = cp.Variable()
    n2 = cp.Variable()
    y = cp.Variable()

    constraints = [n1 + n2 <= n_max,
                  d1 * n1 - d2 * n2 - d3 * (n2 - n_r) ** 2 - d4 * (n1 + n2) >= y,
                  -d3 * (n2 - n_r) ** 2 - d4 * (n1 + n2) >= y,
                  n1 >= 0,
                  n2 >= 0]

    obj = cp.Maximize(y)

    prob = cp.Problem(obj, constraints)
    prob.solve()

    # 1.3 By solving the system of n1, n2, we get theta and n
    _n = n1.value + n2.value
    _theta = n1.value / (n1.value + n2.value)
    return [_n, _theta]
```



Multi-stage Game with Observed Actions and Incomplete Information

- A more realistic case:
 - The attacker and defender play the same game more than once (multi-stage)
 - The defender is uncertain about the attacker's type, aka weights a_i (incomplete information)
- Properties of repeated games:
 - Players observe the outcome of the first round before starting the second round
 - Payoff of the entire game is the sum of the payoff of the previous stages
 - Player's strategic choices are influenced by the outcome of the choices made in earlier stage
- Assumptions
 - Two different types of attackers $a, b \rightarrow$ Different weights a_i, d_i for each attacker type
 - In each timeslot, all attacks are coming from a specific type of attacker
 - The attacker does not have perfect information of the last value of θ , but knows all previous choices
 - Mixed strategy \rightarrow The attacker plays according to a probability distribution over the available strategies



$d_{1,\{a,b\}}$ damage for real devices being attacked, $d_{2,\{a,b\}}$ benefit for honeypots being attacked, $d_{3,\{a,b\}}$ damage for real devices not served, $d_{4,\{a,b\}}$ damage for the total number of hosts

$a_{1,\{a,b\}}$ benefit for attacking real devices, $a_{2,\{a,b\}}$ damage for attacking honeypots, $a_{3,\{a,b\}}$ damage for total number of attacks

Multi-stage Game Rules

- Defender has limited number of IP addresses (N_{max}), which can be assigned either to operational devices or honeypots
- N is the sum of **connected** real devices and honeypots, N_r is the connected real devices
- Available strategies:
 - $s_{A,i} \in \{1,0\}$. The strategy of the attacker for the i-th host: to attack the i-th host (1) or not to attack
 - $s_{D,i} \in \{-1, 1\}$. The strategy of the defender for the i-th host – use the i-th host as real device (1) or as honeypot (-1)
- $a_{i,\{a,b\}}, d_{i,\{a,b\}}$ represent constant preferences (weights) from the attacker and defender perspective, for attacker type a, or b.
- θ is the portion of N that are honeypots , ϕ is the portion of N that is being attacked
- ϕ_m is the maximum portion of N that are attacked (usually 1 / 100%)
- Two types of attackers, **a** and **b**
- μ is the belief of the defender that the attacker is of type **a**
- All hosts have the same probability of being honeypots or being attacked



Payoffs

$$\mathbb{E}[U_{A_i}] = a_{1,i}, (1 - \theta)\phi_i N - a_{2,i}\theta\phi_i N - a_{3,i}\phi_i N$$

$$\begin{aligned} \mathbb{E}[U_D] = & \mu \times (d_{1,a}\theta\phi_a N - d_{2,a}(1 - \theta)\phi_a N - d_3((1 - \theta)N - N_r)^2 - d_4 N) \\ & + (1 - \mu) \times (d_{1,b}\theta\phi_b N - d_{2,b}(1 - \theta)\phi_b N - d_3((1 - \theta)N - N_r)^2 - d_4 N). \end{aligned}$$





$$(\theta^*, N^*, \phi_a^*, \phi_b^*) =$$

$$\begin{cases} (\tilde{\theta}_1, N_{\max}, \phi_{a,m}, \phi_{b,m}), \text{ if} \\ \quad 0 \leq \tilde{\theta}_1 \leq 1 \text{ and } \mu d_{1,a} \phi_{a,m} + d_{1,b} \phi_{b,m} (1 - \mu) \geq d_4 \\ \quad \text{and } -\mu (d_{1,a} + d_{2,a}) \phi_{a,m} + (\mu - 1) (d_{1,b} + d_{2,b}) \phi_{b,m} + 2d_3 N_r \geq \frac{2d_3 (a_{2,a} + a_{3,a}) N_{\max}}{a_{1,a} + a_{2,a}} \\ \quad \text{and } -\mu (d_{1,a} + d_{2,a}) \phi_{a,m} + (\mu - 1) (d_{1,b} + d_{2,b}) \phi_{b,m} + 2d_3 N_r \geq \frac{2d_3 (a_{2,b} + a_{3,b}) N_{\max}}{a_{1,b} + a_{2,b}}, \\ (0, N_r - \frac{d_4 + d_{2,b} \phi_{b,m} + d_{2,a} \mu \phi_{a,m} - d_{2,b} \mu \phi_{b,m}}{2d_3}, \phi_{a,m}, \phi_{b,m}), \text{ if} \\ \quad \tilde{\theta}_1 < 0 \text{ and } 0 \leq N_r - \frac{d_4 + d_{2,b} \phi_{b,m} + d_{2,a} \mu \phi_{a,m} - d_{2,b} \mu \phi_{b,m}}{2d_3} \leq N_{\max}, \\ (0, N_{\max}, \phi_{a,m}, \phi_{b,m}), \text{ if} \\ \quad \tilde{\theta}_1 < 0 \text{ and } N_r - \frac{d_4 + d_{2,b} \phi_{b,m} + d_{2,a} \mu \phi_{a,m} - d_{2,b} \mu \phi_{b,m}}{2d_3} > N_{\max}, \\ (0, 0, \phi_{a,m}, \phi_{b,m}), \text{ if} \\ \quad \tilde{\theta}_1 < 0 \text{ and } N_r - \frac{d_4 + d_{2,b} \phi_{b,m} + d_{2,a} \mu \phi_{a,m} - d_{2,b} \mu \phi_{b,m}}{2d_3} < 0, \\ (\tilde{\theta}_2, N_{\max}, \phi_{a,m}, 0), \text{ if} \\ \quad 0 \leq \tilde{\theta}_2 \leq 1 \text{ and } \mu \phi_{a,m} d_{1,a} \geq d_4 \text{ and } \frac{(a_{1,a} + a_{2,a})(2d_3 N_r - \mu \phi_{a,m} (d_{1,a} + d_{2,a}))}{d_3 (a_{2,a} + a_{3,a})} \geq N_{\max} \\ \quad \text{and } \frac{(a_{1,b} + a_{2,b})(2d_3 N_r - \mu \phi_{a,m} (d_{1,a} + d_{2,a}))}{d_3 (a_{2,a} + a_{3,a})} \leq N_{\max}, \\ (0, 0, \phi_{a,m}, 0), \text{ if} \\ \quad \tilde{\theta}_2 < 0 \text{ and } N_r - \frac{d_4 + \mu d_{2,a} \phi_{a,m}}{2d_3} < 0, \\ (\tilde{\theta}_3, N_{\max}, 0, \phi_{b,m}), \text{ if} \\ \quad 0 \leq \tilde{\theta}_3 \leq 1 \text{ and } (\mu - 1) \phi_{b,m} d_{1,b} \geq d_4 \text{ and } \frac{(a_{1,a} + a_{2,a})(2d_3 N_r - \mu \phi_{a,m} (d_{1,a} + d_{2,a}))}{d_3 (a_{2,a} + a_{3,a})} \leq N_{\max} \\ \quad \text{and } \frac{(a_{1,b} + a_{2,b})(2d_3 N_r - \mu \phi_{a,m} (d_{1,a} + d_{2,a}))}{d_3 (a_{2,a} + a_{3,a})} \geq N_{\max}, \\ (0, 0, 0, \phi_{b,m}), \text{ if} \\ \quad \tilde{\theta}_3 < 0 \text{ and } N_r - \frac{(1-\mu) \phi_{b,m} d_{2,b} + d_4}{2d_3} < 0, \\ \nexists, \text{ elsewhere,} \end{cases}$$

Bayesian NE (BNE)

- Similarly with NE, $\phi_a^* \in \{0, \phi_{a,m}\}$ and $\phi_b^* \in \{0, \phi_{b,m}\}$
- We assume that μ is a common prior, i.e. the attacker knows the defender's belief of μ

where

$$\tilde{\theta}_1 = \frac{(d_{1,a} + d_{2,a}) \mu \phi_{a,m}}{2N^* d_3} + \frac{(d_{1,b} + d_{2,b} - \mu d_{1,b} - \mu d_{2,b}) \phi_{b,m} + 2d_3 (N^* - N_r)}{2N^* d_3},$$

$$\tilde{\theta}_2 = \frac{\mu \phi_{a,m} d_{1,a} + \mu \phi_{a,m} d_{2,a} + 2d_3 N^* - 2d_3 N_r}{2N^* d_3},$$

$$\tilde{\theta}_3 = \frac{(1 - \mu) \phi_{b,m} d_{1,b} + (1 - \mu) \phi_{b,m} d_{2,b} + 2d_3 N^* - 2d_3 N_r}{2N^* d_3},$$

Update of belief

$$\mathbb{P}(\phi^t | A_a, h^t) = \binom{N}{\phi^t N} (\phi_a^{t*})^{\phi^t N} (1 - \phi_a^{t*})^{N(1-\phi^t)}$$

$$\mathbb{P}(\phi^t | A_b, h^t) = \binom{N}{\phi^t N} (\phi_b^{t*})^{\phi^t N} (1 - \phi_b^{t*})^{(1-\phi^t)N}$$

- Each round, the defender's belief about the attacker's type should be updated. In general: $\mu^t = \mathbb{P}(A_a | h^t)$ and $1 - \mu^t = \mathbb{P}(A_b | h^t)$
- h^t is the history profile of the attacker
- Using Bayes' rule, the belief can be defined as:

$$\mu^{t+1} = \frac{\mathbb{P}(\phi^t | A_a, h^t) \mathbb{P}(A_a)}{\mathbb{P}(\phi^t | h^t)}$$

- By making the following operations:
 - $\mathbb{P}(A_a) = \mu^t$
 - $\mathbb{P}(\phi^t | A_a, h^t)$ and $\mathbb{P}(\phi^t | A_b, h^t)$ are calculated according to binomial distribution, since each player believes that their opponent plays according to BNE.

• Finally:

$$\mu^{t+1} = \frac{\mu^t (\phi_a^{t*})^{\phi^t N} (1 - \phi_a^{t*})^{N(1-\phi^t)}}{\mu^t (\phi_a^{t*})^{\phi^t N} (1 - \phi_a^{t*})^{N(1-\phi^t)} + (1 - \mu^t) (\phi_b^{t*})^{\phi^t N} (1 - \phi_b^{t*})^{(1-\phi^t)N}}$$

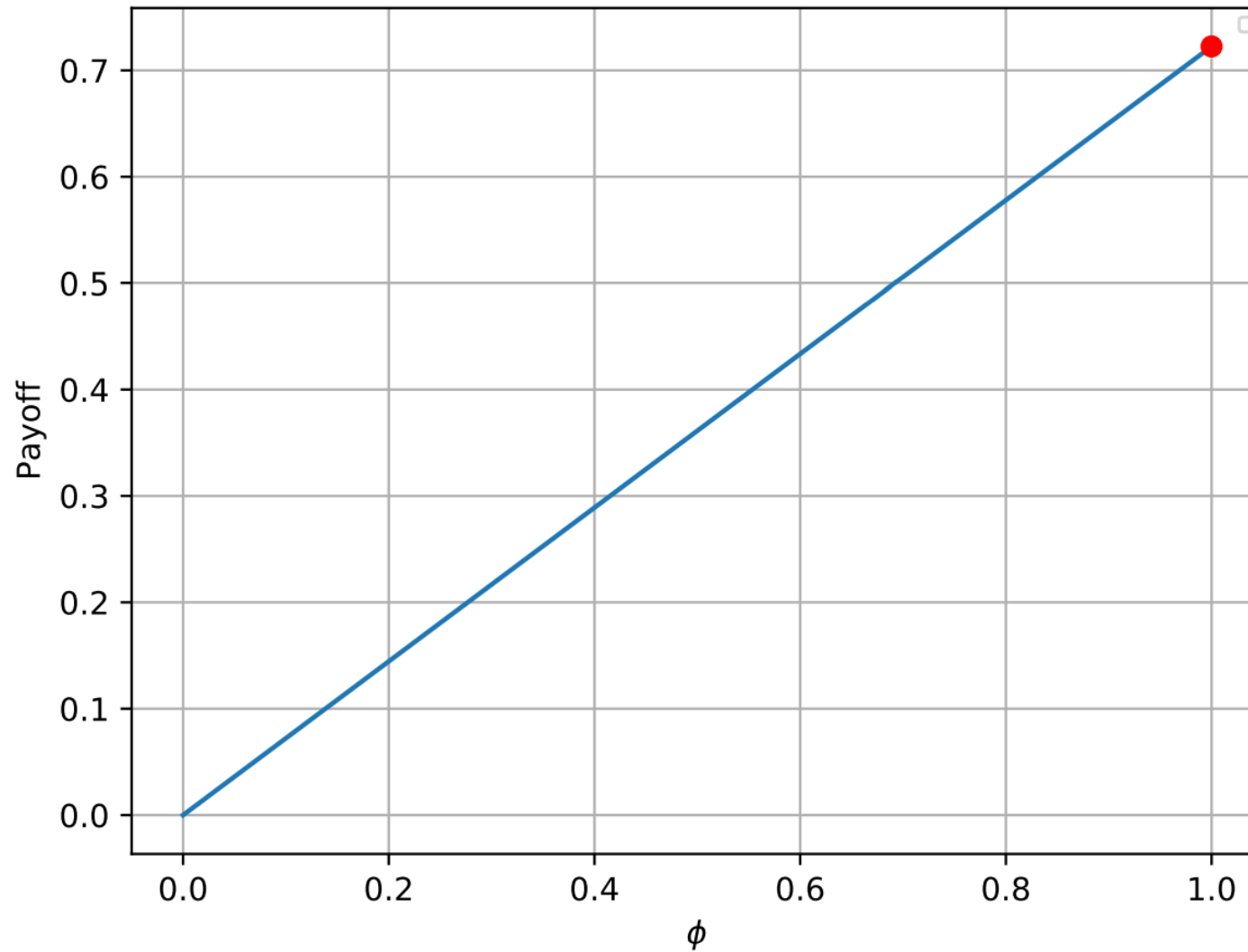


Experiment Setup

- Comparing the NE for the attacker and the defender with 2000 random solutions to verify that the NE yields the maximum payoff, whilst the opponent always chooses the best strategy

Parameter	Value
N_r	3
N_{max}	10
ϕ_{max}	1
$a_{\{1,2,3\}}$	[0.76, 0.01, 0.1]
$d_{\{1,2,3\}}$	[0.03, 0.4, 0.45, 0.01]
Random solutions for θ	2000
Random solutions for ϕ	2000

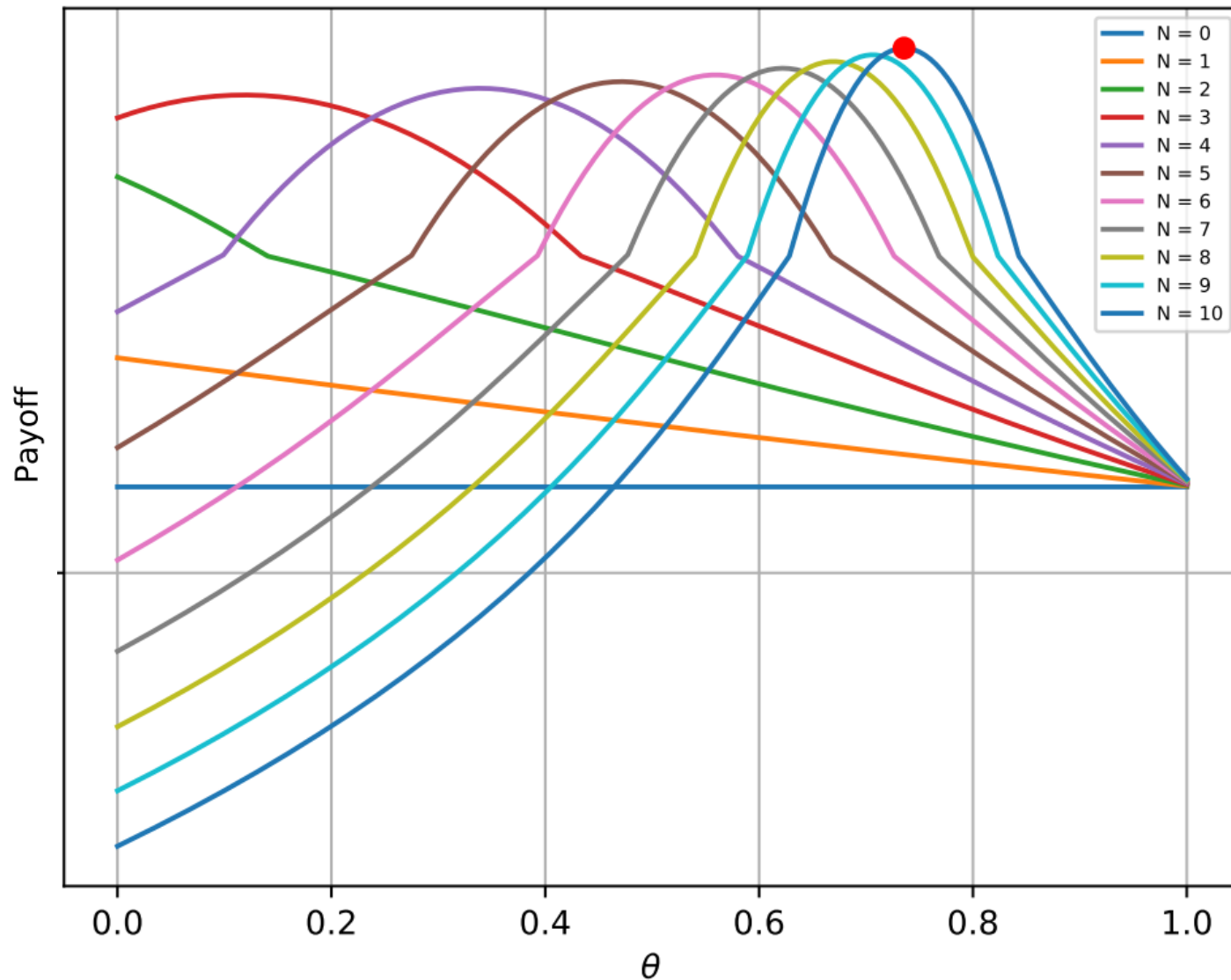




Attacker's Payoff

- The attacker's payoff increases as ϕ increases
- The Red bullet denotes the NE payoff
- Verifies that NE yields maximum payoff, **assuming that** the defender chooses the best strategy





Defender's Payoff

- For each N , the payoff follows specific pattern
- Big number of N yields maximum possible payoff
- $\theta \cong 0.77$

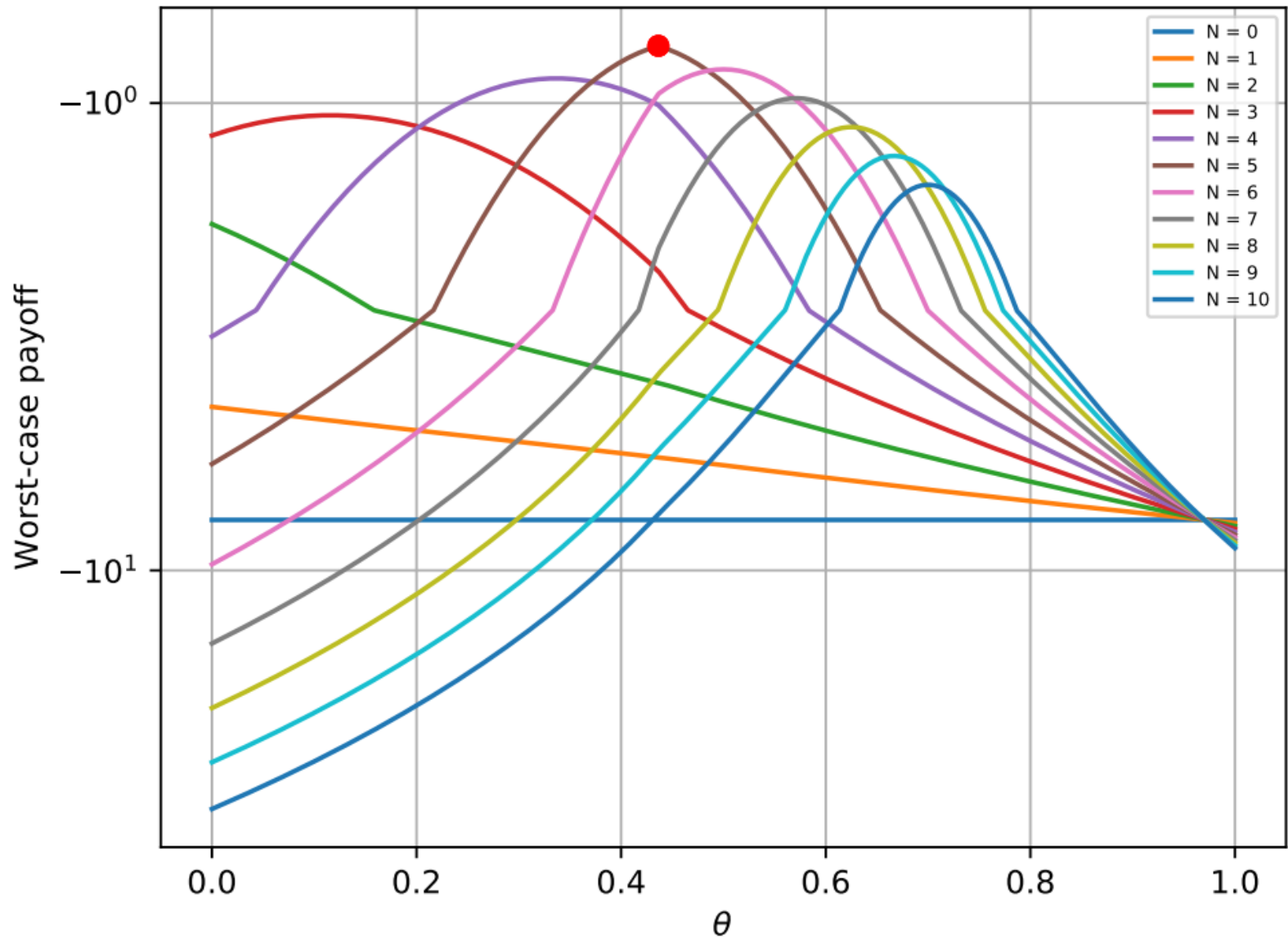


Experiment Setup - When NE does not exist

- Comparing the NE for the attacker and the defender with 2000 random solutions to verify that the NE yields the maximum payoff, whilst the opponent always chooses the best strategy

Parameter	Value
N_r	3
N_{max}	10
ϕ_{max}	1
$a_{\{1,2,3\}}$	[0.81, 0.01, 0.06]
$d_{\{1,2,3\}}$	[0.31, 0.24, 0.81, 0.14]
Random solutions for θ	2000
Random solutions for ϕ	2000





Maxmin Payoff

- Great N does not guarantee maximum payoff
- $N^* = 5$
- $\theta \cong 0.42$

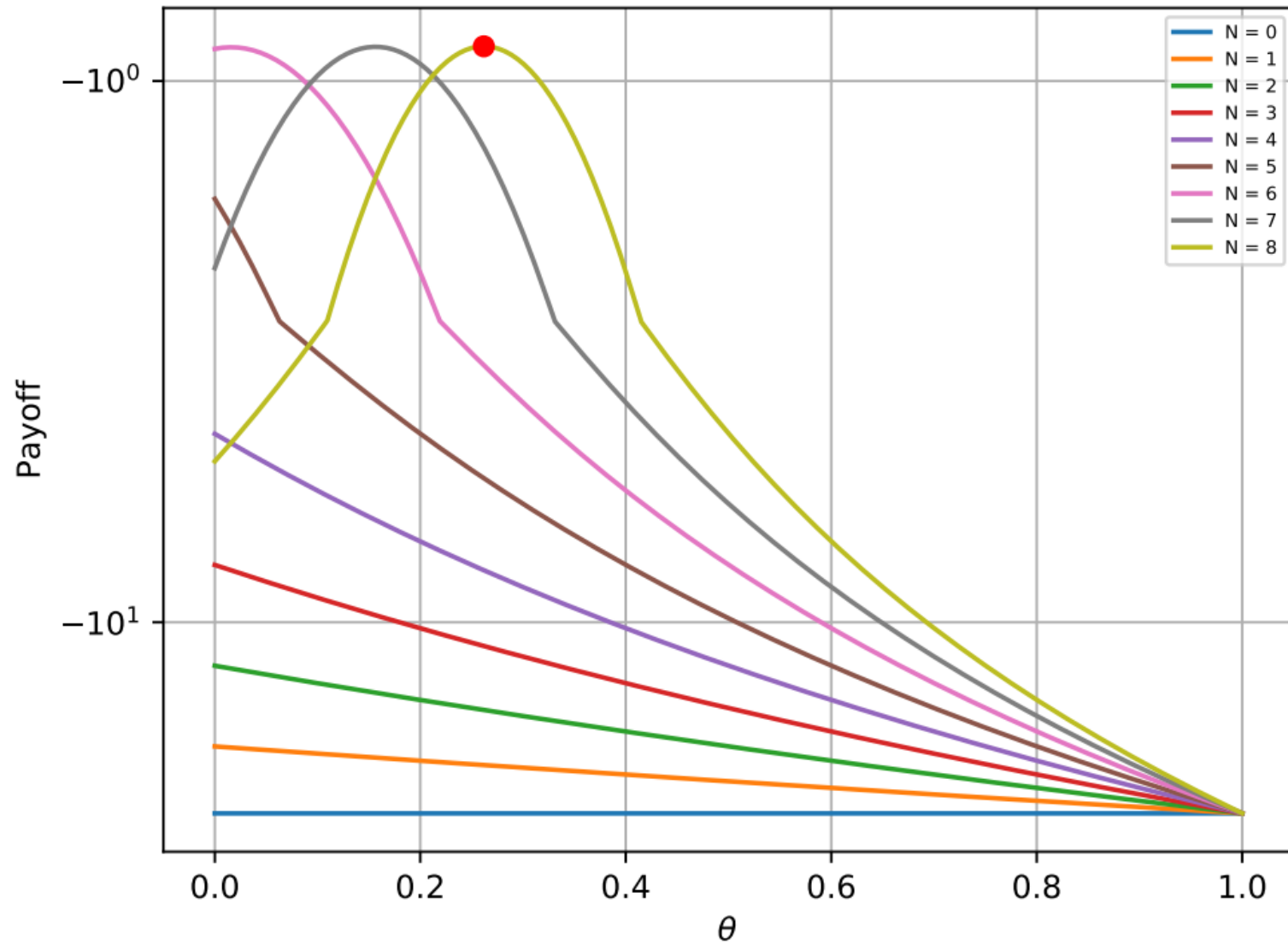


Experiment Setup – Repeated Game

- Comparing the NE for the attacker and the defender with 2000 random solutions to verify that the NE yields the maximum payoff, whilst the opponent always chooses the best strategy

Parameter	Value
Number of rounds	50
N_r	6
N_{max}	8
$\phi_{a,max}$	0.6
$\phi_{b,max}$	0.2
$a_{a\{1,2,3\}}$	[0.48, 0.46, 0.10]
$a_{b\{1,2,3\}}$	[0.39, 0.48, 0.02]
$d_{a\{1,2\}}$	[0.70, 0.04]
$d_{b\{1,2\}}$	[0.04, 0.68]
d_3, d_4	0.77, 0.006
Random solutions for θ, ϕ	2000 each

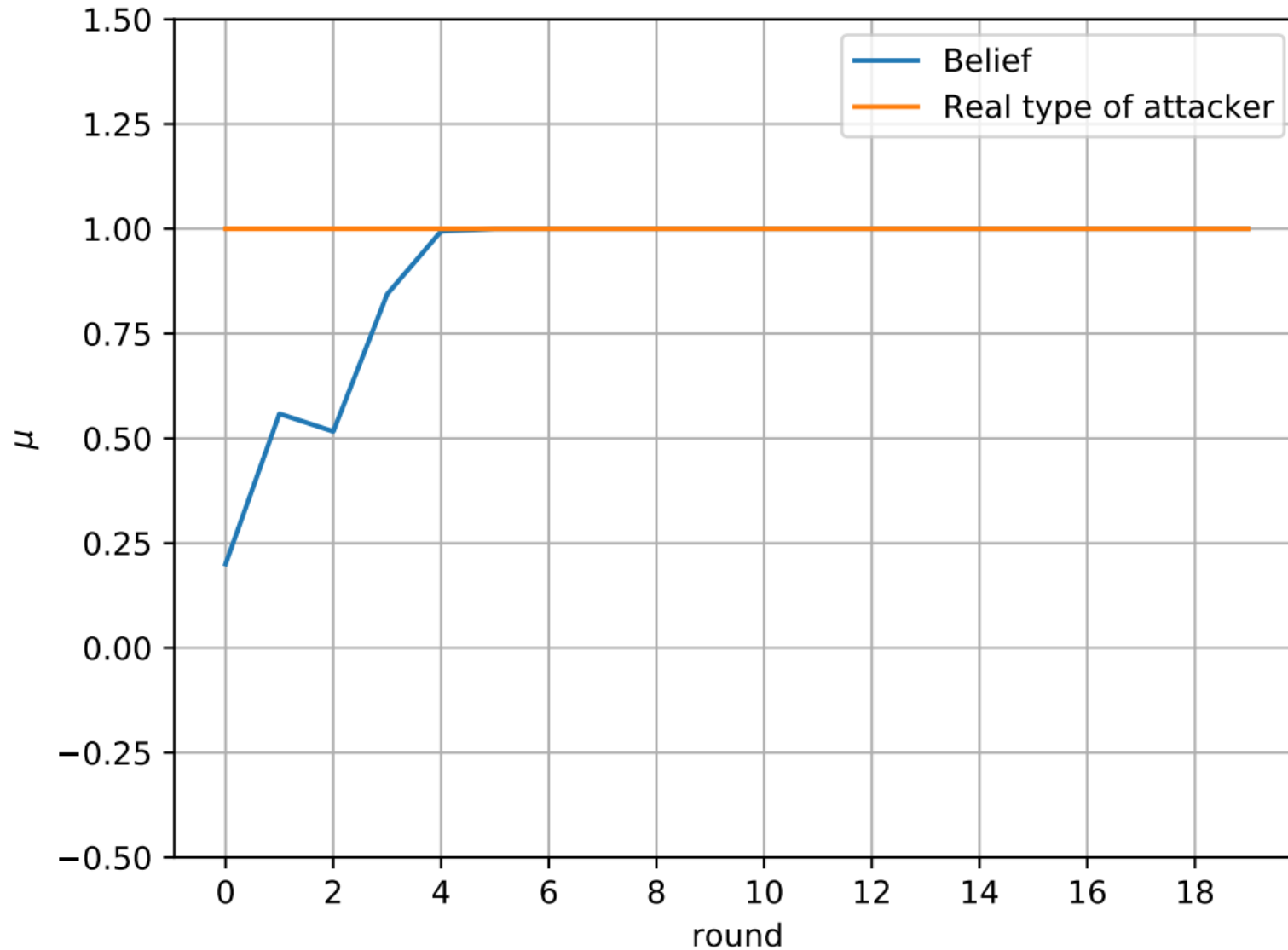




Defender's Payoff - BNE

- Greater N yields better payoff
- $N = 8$
- $\theta \cong 0.38$





Convergence of μ

- Compares the belief of the defender VS the actual type of the attacker
- Starting point of μ is ~ 0.24
- After 4 rounds, the defender realizes the actual type of attacker



References

- P. Diamantoulakis, C. Dalamagkas, P. Radoglou-Grammatikis, P. Sarigiannidis, and G. Karagiannidis, "Game Theoretic Honeypot Deployment in Smart Grid," Sensors, vol. 20, no. 15, p. 4199, Jul. 2020: <https://zenodo.org/record/3967314>
- N. Nisan, T. Roughgarden, E. Tardos, and V. V. Vazirani, Eds., Algorithmic Game Theory. Cambridge University Press, 2007. (<https://www.cs.cmu.edu/~sandholm/cs15-892F13/algorithmic-game-theory.pdf>)
- C. Dalamagkas, P. Sarigiannidis, S. Kapetanakis and I. Moscholios, "Dynamic scheduling in TWDM-PONs using game theory", Optical Switching and Networking, Dec. 2017, DOI: 10.1016/j.osn.2017.12.004
- C. Dalamagkas et al., A Survey On Honeypots, Honeynets And Their Applications On Smart Grid," in 2019 IEEE Conference on Network Softwarization (NetSoft), 2019. DOI: 10.1109/NETSOFT.2019.8806693.
- Design of Market Mechanism for Bandwidth Allocation in XG-PON, <https://dspace.uowm.gr/xmlui/handle/123456789/835>

