Optimizing Security Strategies: A game theoretic approach to attack-defense trees

ARSALAAN KHAN, University of Twente, The Netherlands

Cybersecurity analysis requires effective methods to model various attack scenarios while managing associated costs. This research addresses this challenge by utilizing game theory solvers within the PRISM model checker to identify cost-effective security strategies in Attack Defense Trees (ADT). ADT provides a structured framework for modeling potential attack scenarios and defense strategies, while game theory offers a formal methodology for analyzing strategic interactions between attackers and defenders. This study develops methodologies for translating ADT into PRISM-compatible models and optimizing security strategies based on cost considerations. The research objectives include investigating the feasibility of using game theory solvers in the context of ADT, integrating them with PRISM, identifying cost-effective strategies using PRISM games, evaluating the cost-effectiveness of these strategies, and providing practical insights for practitioners. The findings demonstrate the viability of this approach and offer valuable guidance for enhancing cybersecurity defenses through cost-focused strategic modeling and analysis.

Additional Key Words and Phrases: Security, Game Theory, Attack Defense Trees (ADT) , Prism Model Checker

1 INTRODUCTION

In today's interconnected world, ensuring the security of systems and networks is becoming increasingly important. With the ever-increasing complexity of threats and vulnerabilities, the need for effective strategies to mitigate risks and protect critical assets has become crucial. There have been more than 350 million victims of cyberattacks or system failures in 2023 [7], which signifies the rising threat of these attacks. These threats have created various challenges for security professionals to ensure the security of computer systems.

However, these challenges are not new. Cybersecurity specialists have been fighting these attackers for decades and have been proposing effective solutions to ensure the security of a system. As better defensive solutions are proposed, more creative attack strategies are also used, leading to an arms race between attackers and defenders. This constant evolution results in systems having various vulnerabilities that attackers can exploit, while also having some defenses set up to prevent these attacks. A formal way to represent these possible attack scenarios and the defenses of the system was introduced in the form of Attack Defense Trees (ADTs) [9]. Following the terminology introduced in [9], the root node in an ADTree is called the proponent, and the other agent is the opponent. The aim of the proponent is to achieve the root goal, whereas the opponent tries to prevent the proponent from reaching that goal. An example ADT is demonstrated in Fig. 3

Each attack has a cost associated with it since it requires resources to execute. Similarly, deploying each defense also incurs costs. The goal of an attacker is to minimize its costs while achieving the goal stated in the root node when the attacker is the proponent. Conversely, the goal of a defender is to ensure that the attacker does not achieve its goals while minimizing its own costs. These competing goals of the attacker and defender indicate that this scenario can be described as a game where both parties compete against each other. In a competitive scenario like this, the concept of Pareto front becomes relevant. Instead of one optimal solution, there is a family of optimal solutions called the Pareto front. [12] The Pareto front is the solution in which one of the objectives cannot be improved without worsening another objective. [12]

Given that these scenarios can be described as games where both parties compete against each other, game theory principles could be effectively utilized to analyze an Attack Defense Tree [8]. Game theory provides a powerful framework for analyzing strategic interactions among multiple decision-makers with conflicting objectives. By modeling security scenarios as games between attackers and defenders, it is possible to systematically explore different strategies and their potential outcomes, leading to optimal solutions for both attackers and defenders.

However, there are many possible actions that can be taken by an attacker or a defender, especially in larger ADTs which are more applicable in real-life scenarios. Therefore, computing the optimal solutions for an Attack Defense Tree manually is not the most efficient way to approach the problem. Finding fast algorithms to analyse attack defense trees is crucial because of the increasing complexity and frequency of cyber threats. Faster algorithms would enable organizations to proactively strengthen their security systems. Consequently, PRISM games [5], an extension of the PRISM model checker that allows for the analysis of stochastic multiplayer games, has been utilized. Instead of calculating one optimal solution for attackers and defenders, this study aims to calculate all Pareto optimal solutions using the PRISM framework [10]. The goal of this research is to investigate how game theory solvers, specifically PRISM games, can be utilized within the PRISM framework to identify optimal security and countermeasure strategies in ADTs. Calculating the costs of executing an attack or deploying a defense falls outside the scope of this research. This study has assumed that all the costs involved are already known.

In the following sections of this paper, the specific objectives, related work, methodology, and outcomes of this research will be explored. The discussion will begin by detailing the specific objectives of the research, including the integration of game theory solvers with ADTs and PRISM, and the development of methodologies for translating ADTs into PRISM-compatible models. Related work in the fields of cybersecurity, game theory, and formal verification will then be reviewed to provide a comprehensive context for the study. The methodology section will describe the approach to

TScIT 41, July 8, 2024, Enschede, The Netherlands

modeling and analysis, including the use of PRISM games to identify Pareto optimal strategies. Finally, the outcomes of the research will be presented, demonstrating the effectiveness of the methodologies and offering practical insights for cybersecurity practitioners aiming to enhance the security of their systems through strategic modeling and analysis.

In summary, this research makes a significant contribution to the field of cybersecurity by combining the strengths of ADTs and game theory with the powerful capabilities of the PRISM model checker. The methodologies and strategies developed in this study have the potential to transform how cybersecurity defenses are designed and implemented, providing a more robust and strategic approach to countering sophisticated cyber threats. By advancing the understanding of optimal defense strategies and the application of formal verification methods in cybersecurity, this research aims to pave the way for more secure and resilient systems in an increasingly digital world.

2 RELATED WORK

Research in the fields of security analysis, game theory, and formal methods has laid the foundation for this research on optimizing security strategies with game theory and Attack Defense Trees (ADTs).

Game theory has played a significant role in security analysis, providing a formal framework for modeling strategic interactions between rational decision-makers [1]. Researchers have applied game theory concepts, such as Nash equilibrium and Bayesian games to analyze security games and identify optimal defense strategies against potential attacks [4]. By modeling the interactions between attackers and defenders as strategic games, researchers can systematically explore and quantify the effectiveness of various security strategies, offering valuable insights for enhancing cybersecurity measures.

Kordy et al. have made the connection between game theory and ADTs explicit [8]. They have argued in their research that attack-defense trees and binary zero-sum two-player games have equivalent expressive power when considering satisfiability. They also demonstrated that these models can be converted into each other while preserving their outcome and internal structure, establishing a formal equivalence between the two representations. This equivalence highlights the potential for leveraging game-theoretic approaches to analyze and optimize security strategies within the ADTs framework. Unlike this research paper, the research by Kordy et al. does not attempt to utilize their proof to model an attack defense tree as a game and use game theoretic principles to calculate optimal costs for the attacker and the defender.

Utilizing strategic games on ADTs is not a new concept. One research has shown an approach that can be used to evaluate the effectiveness and economic profitability of countermeasures (defenses) as well as their deterrent effect on attackers [4]. Their research provides decision-makers with a useful tool for performing better evaluations of IT security investments during the risk management process . This work outlines the importance of integrating economic considerations into security strategy analysis, ensuring that defenses are not only effective but also cost-efficient.

Recent research has also explored the use of PRISM games for security analysis. Eisentraut and Křetínský (2019) extended ADTs with costs and success probabilities, providing a framework for analyzing the probability of successful attacks/defenses and their expected costs. They proposed algorithms for reduction to PRISM-games and direct analysis of ADTs[6]. While this research focuses on identifying one optimal solution, this study seeks to calculate Pareto optimal solutions, offering a broader perspective on the trade-offs between different security strategies.

Pareto optimal solutions of ADTs have also been researched by Aslanyan and Nielson. In their research, they devised automated techniques that optimize all parameters at once. Moreover, in the case of conflicting parameters, their techniques compute the set of all optimal solutions, defined in terms of Pareto efficiency [3]. However, it is important to note that their techniques are limited to proper trees and do not extend to Directed Acyclic Graphs (DAGs). Lopuhaä - Zwakenberg et al. have argued that proper trees are more restrictive in terms of node connections, allowing only one parent for each node (except the root), while DAGs permit multiple parents for a node [11]. In contrast to the research of Aslanyan and Nielson, this research uniquely focuses on extending the concept of Pareto optimality to the more complex structure of DAGs within ADTs. By addressing this gap in the literature, this study aims to provide a more comprehensive understanding of Pareto optimal solutions in the context of ADTs with DAG structures.

Additionally, integrating PRISM with game theory solvers introduces new dimensions to security analysis. The PRISM model checker has been widely used for analyzing probabilistic systems, and its extension to PRISM-games allows for the modeling of stochastic multiplayer games. This capability is crucial for accurately representing the dynamic and probabilistic nature of cyberattacks and defenses, enabling more realistic and robust security strategy optimization.

This research seeks to build on these foundational works by combining game theory, ADTs, and PRISM-games to develop a comprehensive framework for optimizing security strategies. By translating ADTs into PRISM-compatible models and leveraging game-theoretic principles to identify Pareto optimal solutions, this study aims to enhance the strategic planning and decision-making processes in cybersecurity. The ultimate goal is to equip security professionals with advanced tools and methodologies that offer a balanced approach to minimizing risks and costs while maximizing defense effectiveness.

3 RESEARCH QUESTION

The primary research question (RQ) guiding this study is:

RQ: *How can game theory solvers be utilized to identify optimal security and countermeasure strategies in Attack Defense Trees?*

This question represents the central aim of this study, which is to explore the application of game theory principles in the analysis and optimization of security strategies within the context of Attack Defense Trees (ADTs). By addressing this question, the understanding of how strategic interactions between attackers and defenders can be systematically analyzed to improve cybersecurity defenses.

Researching this primary question naturally leads to several sub-research questions (Sub RQs), which delve into specific aspects of the broader inquiry:

Sub RQ1*: How feasible is it to utilize game theory solvers to find an optimal security strategy in Attack Defense Trees?*

This sub-question focuses on the practicality and effectiveness of employing game theory solvers in the context of ADTs. It seeks to determine the extent to which these solvers can handle the complexities and nuances of ADTs, including the scalability of the approach and the computational resources required. Investigating this feasibility aims to identify potential limitations and strengths of integrating game theory solvers with ADTs, thereby providing a foundational understanding of their applicability.

Sub RQ2*: How can Attack Defense Trees be translated into a PRISM-compatible model?*

The second sub-question addresses the methodological aspect of this research. It involves developing and refining algorithms and techniques to convert ADTs, which represent complex security scenarios, into models that are compatible with the PRISM model checker's input requirements. This translation process is critical for enabling the use of PRISM games in the analysis, and it involves ensuring that the semantics and structure of the original ADTs are preserved in the PRISM-compatible models. This sub-question explores the challenges and solutions associated with this translation process, aiming to establish a robust framework for accurate model conversion.

Sub RQ3*: How does the output of the PRISM model checker relate to Attack Defense Trees?*

The third sub-question focuses on the interpretation and application of the results obtained from the PRISM model checker. It seeks to understand how the outputs generated by PRISM games, such as optimal strategies and Pareto optimal solutions, can be mapped back to the original ADTs. This involves analyzing how the strategic insights provided by PRISM games can inform decision-making processes in cybersecurity, and how these results can be used to enhance the effectiveness of security strategies. By addressing this sub-question, this research aims to bridge the gap between theoretical analysis and practical application, ensuring that the findings of this research can be effectively utilized by security professionals.

4 BACKGROUND

4.1 Attack Defense Tree

Attack Defense Trees are hierarchical structures where nodes represent attack or defense actions, and the tree's structure illustrates the dependencies and relationships between these actions. An attacker aims to compromise a system by achieving certain goals, while a defender aims to prevent the attacker from reaching these goals by implementing countermeasures[9].

In this paper, the representation of the hierarchial structure is similar to the one used in the paper by Kordy et al [9]. Attacks are represented by a circular red node and defenses are by a circular green node which is demonstrated in Fig. 1

These nodes are connected by a line, indicating the logical relationships and dependencies between them. An attack node connected to another attack node signifies sequential attack step required to achieve a goal. Conversely, a defense node connected to an attack node represents a countermeasure to that attack.

Attack
Defense
Attack node
Defense node

Fig. 1. An attack and a defense node

To further illustrate the logical dependencies between actions, AND gates and OR gates are used within the tree structure. AND gates, as shown in Fig 2 are represented by an arc connecting multiple nodes to a single node which indicates that all preceding actions must be successfully executed for the subsequent action to be viable. On the other hand, OR gates are not represented by an arc. When nodes do not have an arc connecting them, they form an OR gate. This indicates that if any of the preceding actions are executed successfully, it is sufficient to proceed to the next step. This means that an attacker has multiple options to achieve the same goal. Similarly, defenders can use OR gates to represent alternative defenses , where executing any of them can prevent an attack.

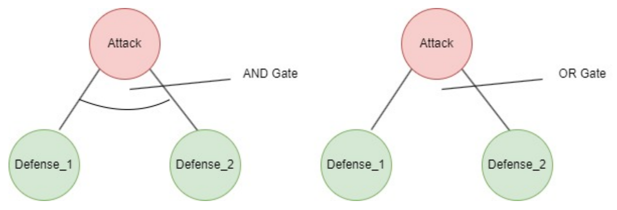


Fig. 2. Illustration of the AND gate and the OR gate

With this information, we can now analyze an entire Attack Defense Tree , evaluating the various attack strategies and the corresponding defense mechanisms. An example tree is shown in Fig. 3. Using PRISM games, this paper has determined the optimal costs for the attacker and the defender within this tree.

An analysis of the tree given in Fig. 3 can be performed to understand the cost dynamics between the attacker and the defender in that tree. The costs associated with each node are provided in curly braces to the right of the node.

The tree begins with a "Root Node," which is the node that the attacker wants to reach. The defender wants to prevent the attacker from reaching it. The root node branches into two main attack nodes: A1 and A2, with costs of 20 and 15, respectively.

For A1, the defender can counter the attack with the defense D1, costing 10 or defense D1_2, costing 2. If the attacker chooses A2, the defender has to defending with D2 and D2_2, costing 5 and 4, respectively.

There is an OR gate between A1 and A2 and the root node. Therefore, the attacker can choose either of the attacks to reach the root

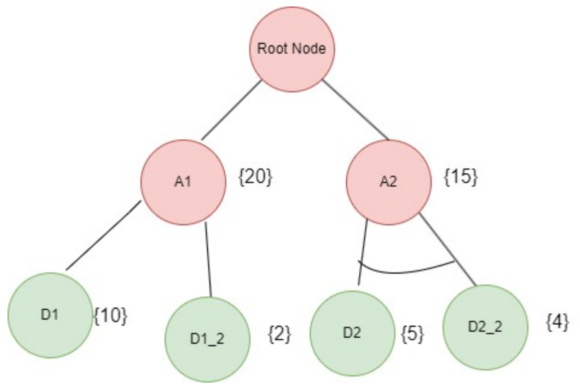


Fig. 3. Example Attack Defense Tree

node. The minimum cost for the attacker to succeed is 15 since that is the lowest costing node (A2) that will allow the attacker to reach the root node. There is an OR gate between the defenses D1 and D1_2. Therefore, to defend A1 , the defender can choose between D1 and D1_2. Since D1_2 only costs 2 and the D1 costs 10, D1_2 is the optimal defense for attack node A1. Since the attacker can reach the root node with either A1 or A2, both the attacks must be defended. There is an AND gate between D2 and D2_2. Therefore, both of them must be active to defend against attack A2. The cost to defend against attack A2 is the sum of the cost of D2 and D2_2 which is 9. Considering these costs for the defender, the minimum cost for the defender to guarantee that the attacker does not reach the root node is the minimum cost to defend A1 and the minimum cost to defend A2 whose sum is equal to 11.

By modeling this ADT in PRISM games and using PRISM's property verifying tool, it is possible to confirm these strategic cost assessments. This will validate this analysis and ensure that the theoretical cost optimization aligns with practical simulation outcomes.

4.2 PRISM games

PRISM Games is an extension of the PRISM model checker, specifically designed to support the analysis and verification of stochastic multi-player games (SMGs). It combines probabilistic verification techniques with game-theoretic approaches to model and analyze systems where multiple agents (or players) interact, often with competing objectives[5]. PRISM Games extends the capabilities of the PRISM model checker to handle scenarios involving strategic decision-making under uncertainty [5] , [10]. It allows the modeling of various types of games, including turn-based games where players take turns making decisions and also concurrent games where players make decisions simultaneously [5].The main advantage of PRISM Games is its ability to handle complex interactions between multiple players, each with their own strategies and objectives.

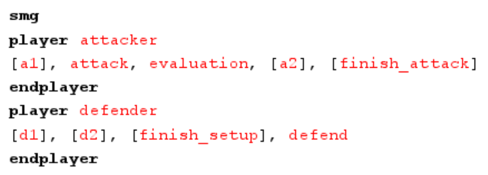
The choice of PRISM Games for modeling and analyzing Attack-Defense Trees (ADTs) was made due to its unique capabilities and suitability for handling the complexity and strategic interactions present in security scenarios. PRISM Games allows for the explicit modeling of different players (e.g., attackers and defenders) and their respective strategies, making it an ideal tool for analyzing ADTs [5]. ADTs involve strategic decision-making where both attackers and defenders have competing objectives and must choose their actions based on their goals. PRISM Games supports this through its formalism for specifying player actions and reward structures. This ensures that all possible outcomes of the interactions between attackers and defenders are considered, providing a comprehensive analysis of the security measures.

5 METHODOLOGY

5.1 Development of Translation Method

A systematic methodology was developed to translate ADTrees into PRISM-compatible models. This involved conceptualizing the translation process and establishing clear mappings between ADT components and PRISM model structures. The resulting translation methods provided a robust framework for leveraging PRISM's analytical capabilities in security strategy optimization.

(1) **Defining players: [Fig. 4]**
In PRISM games, the models modelled as Stochastic Multi-Player Games (SMG), defining the players is the foundational step. The SMG framework is suitable for scenarios involving multiple players with possibly competing objectives. In the ADT model, the attacker and defender have opposing goals where the attacker aims to reach the root node while the defender aims to prevent this. While this research does not involve randomness, the SMG framework allows for future extensions where actions could have probabilistic outcomes. In this ADT model, there are two players: the attacker and the defender. Each player has a set of actions they can perform during their turn.
Player Attacker: The attacker has actions corresponding to attacking the nodes (A1 and A2) and finishing their attack phase.
Player Defender: The defender has actions corresponding to defending the nodes (D1, D1_2, D2, D2_2) and finishing their setup phase.



```
smg
player attacker
[a1], attack, evaluation, [a2], [finish_attack]
endplayer
player defender
[d1], [d2], [finish_setup], defend
endplayer
```

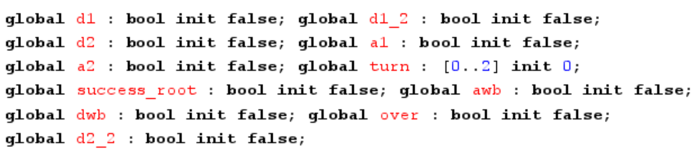
Fig. 4. Player Definition in PRISM

.

(2) **Define Global Variables: [Fig. 5]**

Global variables are used to maintain the state of the game, such as whether a particular node is active, the current turn, and the overall outcome of the game.
Node Activation: d1, d1_2, d2, a1, a2, d2_2 are boolean variables indicating whether the respective nodes are active. All

node state variables are initialized to false to indicate that none of the nodes are active at the start of the game.
Turn Management: turn is an integer variable that tracks whose turn it is (0 for defender setup, 1 for attacker actions, 2 for evaluation).
Outcome Tracking: success_root, awb, dwb, and over are boolean variables used to determine the outcome of the game. success_root indicates if the attacker has reached the root node. The variables awb and dwb are used to check if the attacker or the defender has won the game respectively. Boolean over indicates whether the game is over.

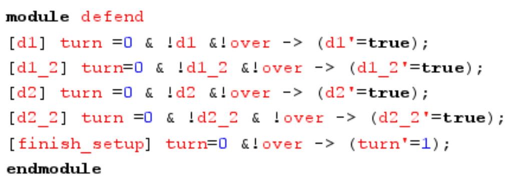


```
global d1 : bool init false; global d1_2 : bool init false;
global d2 : bool init false; global a1 : bool init false;
global a2 : bool init false; global turn : [0..2] init 0;
global success_root : bool init false; global awb : bool init false;
global dwb : bool init false; global over : bool init false;
global d2_2 : bool init false;
```

Fig. 5. Global Variables defined in PRISM

.

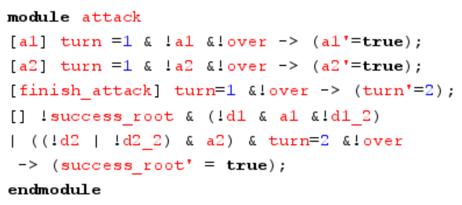
(3) **Defender Module: [Fig. 6]**
Modules in PRISM represent the different actions that can be taken by the players which in this case are the defender and the attacker. Each module specifies the possible actions that can be taken by a player and the conditions under which these actions can occur. The defender's module includes actions to activate defense nodes (D1, D1_2, D2, and D2_2). The finish_setup action indicates the end of the defender's turn.
Action Definitions: Each [dX] action checks if the current turn is the defender's setup phase (turn = 0) and if the node is not yet activated (!dX), then activates the node (dX' = true).
Turn Transition: [finish_setup] transitions the turn from the defender to the attacker (turn' = 1).



```
module defend
[d1] turn =0 & !d1 &!over -> (d1'=true);
[d1_2] turn=0 & !d1_2 &!over -> (d1_2'=true);
[d2] turn =0 & !d2 &!over -> (d2'=true);
[d2_2] turn =0 & !d2_2 & !over -> (d2_2'=true);
[finish_setup] turn=0 &!over -> (turn'=1);
endmodule
```

Fig. 6. Defend module in PRISM

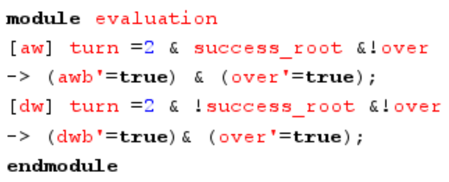
(4) **Attacker Module: [Fig. 7]**
The attacker's module includes actions to activate attack nodes (A1 and A2). The finish_attack action indicates the end of the attacker's turn.
Action Definitions: Each [aX] action checks if the current turn is the attacker's action phase (turn = 1) and if the node is not yet activated (!aX), then activates the node (aX' = true).
Turn Transition: [finish_attack] transitions the turn from the attacker to the evaluation phase (turn' = 2).
Success Condition: The final condition checks if the attack succeeds (success_root = true) based on the activation states of defense and attack nodes.



```
module attack
[a1] turn =1 & !a1 &!over -> (a1'=true);
[a2] turn =1 & !a2 &!over -> (a2'=true);
[finish_attack] turn=1 &!over -> (turn'=2);
[] !success_root & (!d1 & a1 &!d1_2)
| ((!d2 | !d2_2) & a2) & turn=2 &!over
 -> (success_root' = true);
endmodule
```

Fig. 7. Attack module in PRISM

(5) **Evaluation Module: [Fig. 8]**
The evaluation module determines the outcome of the game based on whether the attacker successfully reaches the root node.
Attacker Wins: [aw] action sets the attacker win boolean (awb' = true) and marks the game as over (over' = true) if the attacker succeeded (success_root = true).
Defender Wins: [dw] action sets the defender win boolean (dwb' = true) and marks the game as over (over' = true) if the attacker failed (!success_root).



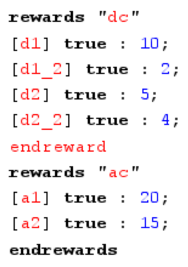
```
module evaluation
[aw] turn =2 & success_root &!over
-> (awb'=true) & (over'=true);
[dw] turn =2 & !success_root &!over
-> (dwb'=true)& (over'=true);
endmodule
```

Fig. 8. Evaluation module in PRISM

(6) **Create the reward structure: [Fig. 9]**
Rewards are defined to represent the costs associated with each action for both the attacker and the defender.By defining rewards in PRISM, specific costs to the actions are assigned. This makes it possible to evaluate the effectiveness of different strategies for both the attacker and the defender. The costs for each action are demonstrated in Fig. 3
Defender Costs: "dc" reward structure defines the cost of each defense action.
Attacker Costs: "ac" reward structure defines the cost of each attack action.

5.2 Identification of Optimal Strategies

Utilizing the PRISM model checker, optimal security strategies within ADTs were identified and optimized. Optimization criteria focused on minimizing attacker costs while optimizing resource allocation for defenders. This made it possible to create the optimal strategy for the attacker and the defender. PRISM Games' support for multiplayer game analysis enabled the identification of optimal



```
rewards "dc"
[d1] true : 10;
[d1_2] true : 2;
[d2] true : 5;
[d2_2] true : 4;
endreward
rewards "ac"
[a1] true : 20;
[a2] true : 15;
endrewards
```

Fig. 9. Reward structure in PRISM

strategy spaces within diverse ADT scenarios which made it possible to define a strategy including the optimal move for the attacker and the defender at any stage of the tree. Several properties within PRISM were used to analyze the ADT. Important properties such as the minimum attacker cost, maximum attacker cost, minimum defender cost and maximum defender cost were analyzed.

5.2.1 Simulation. PRISM offers a simulate feature that allows users to simulate the execution of a model under various scenarios. [2] This feature enables the visualization of state transitions and the examination of how different strategies unfold over time. By using the simulate function, PRISM can generate sample paths through the state space, observing the outcomes and costs associated with each sequence of actions. By observing the state transitions, it is possible to see how defenses are activated and how attacks are executed, leading to a deeper understanding of the strategic dynamics and also makes it possible to easily track the accumulation of costs for both the defender and the attacker. This enables us to compare the total costs incurred under different strategies and scenarios, facilitating the identification of cost-effective approaches.

5.2.2 Properties. Properties in PRISM are formal specifications used to describe and verify the behavior of models. [2] They are written in temporal logic and allow users to query various aspects of the system's performance, reliability, and costs. By defining properties, PRISM can formally analyze the expected outcomes, costs, and success probabilities of both the attacker and defender under various scenarios. This helps in identifying optimal strategies, understanding trade-offs, and ensuring that the modeled security mechanisms meet desired objectives. The properties analyzing the costs of the defender have the «defender» keyword. It specifies that only the player defender's actions should have an effect on this property. This is because the defender plays first and no matter what the attacker chooses the defender has to choose its best option. Conversely, the properties analyzing the costs of the attacker do not have any keyword like the defender does. This is because the attacker must take into account the actions of the defender and make a decision accordingly. The following properties have been used to analyze the ADT :

- **Minimum Defender cost** : The property defined as :

$$<< defender >> R"dc"min =?[F"dw"]$$

calculates the minimum total cost for the defender to achieve a winning state for the defender. A winning state for the defender is when after the attacks have been executed, the variable successRoot remains false. This property is critical for understanding the least amount of resources the defender needs to allocate to successfully defend the attacker's attacks. Returns 0 if it the attacker always has a path to the root node irrespective of the defenses in place suggesting the defender to do nothing and save resources.

- **Minimum Attacker cost** : The property defined as :

$$R"ac"min =?[F"aw"]$$

calculates the minimum total cost for the attacker to achieve a winning state. This property helps in understanding the least cost path the attacker can take to successfully execute an attack. A winning state for the attacker is when after the attacks have been executed, the variable successRoot changes to true.

- **Maximum Defender cost** : The property :

$$<< defender >> R"dc"max =?[F"dw"]$$

calculates the maximum total cost for the defender to achieve a winning state. This property is important for understanding the upper limit of resources the defender might need to allocate to ensure a successful defense.

- **Maximum Attacker cost** : The property :

$$R"ac"max =?[F"aw"]$$

calculates the maximum total cost for the attacker to achieve a winning state. This property is important for evaluating the worst-case scenario for the attacker in terms of cost. It returns infinity if it is possible for the defender to make it impossible for the attacker to reach the root node.

5.3 Pareto Efficient Solutions

Pareto efficient solutions are useful for visualizing the trade-offs between different strategies. In the case of an attack-defense tree (ADT), it helps in understanding the optimal balance between the costs incurred by the defender and the attacker. To obtain the points of the graph, the ADT was simulated multiple times as mentioned in section 5.2.1 Each time a different attacker strategy was used. The minimum cost for the defender to defend the tree with the attacker strategy was noted. Since there are only 2 possible attacker strategies in the example tree used, the cost to attack A1 and A2 were plotted against the minimum cost to defend those nodes respectively. The results are in section 6.2

6 RESULTS AND DISCUSSION

6.1 Property Verification

In this section, the outcomes of this study are presented . This research aimed to identify cost-effective security strategies by leveraging game theory solvers within the PRISM model checker. The properties in Section 5.2.2 have been verified. The results are in the table below.

Property	Value
Minimum Defender Cost	11.0
Maximum Defender Cost	21.0
Minimum Attacker Cost	15.0
Maximum Attacker Cost	∞

Minimum Defender Cost : The minimum cost for the defender represents the optimal scenario where the defender spends the least amount of resources to successfully defend the root node. This cost is incurred when the defender uses the least costly effective defenses. In this case, activating only D1_2 (2) and both D2 (5) and D2_2 (4) results in a total cost of 11.

Minimum Attacker Cost: The minimum cost for the attacker represents the least amount of resources needed to reach the root node when the defender does not activate any defenses. In this case, the attacker can choose the cheaper attack path, which is A2 with a cost of 15.

Maximum Attacker Cost: The maximum cost for the attacker is considered to be infinity because, in a scenario where the defender perfectly defends all paths, the attacker would need an infinite amount of resources to succeed. This implies that under perfect defense, the attacker cannot reach the root node regardless of the resources spent.

Maximum Defender Cost: The maximum cost for the defender represents the worst-case scenario in terms of resource expenditure to defend the root node. This cost is incurred when the defender has to activate all possible defenses to ensure that the attacker does not succeed. In this case, activating D1 (10), D1_2 (2), D2 (5), and D2_2 (4) results in a total cost of 21.

The results provide a comprehensive understanding of the costs associated with the strategic interactions between the attacker and the defender in the modeled attack-defense tree. The range of costs for both players indicates the efficiency and effectiveness of their strategies. For the defender, the wide range between the maximum (21.0) and minimum (11.0) costs highlights the variability in defensive strategies. It emphasizes the importance of choosing cost-effective defenses to minimize resource expenditure while ensuring security. For the attacker, the significant difference between the minimum (15.0) and maximum (infinity) costs highlights the challenge of breaching a well-defended system. The attacker must carefully evaluate the potential defenses and choose the least costly path to maximize their chances of success. These results validate the theoretical model and demonstrate the utility of PRISM games in analyzing complex security scenarios. By identifying the optimal and worst-case costs for both players, we gain valuable insights into the dynamics of attack and defense strategies, enabling better decision-making in real-world applications.

6.2 Pareto Solutions

Through simulation the following results were obtained for all the possible attack strategies such that the defender wins :

Strategy	Attacker cost	Defender Cost
1	0.0	0.0
2	15.0	9.0
3	20.0	2.0

Strategy 1 : This strategy is when the attacker and the defender both do nothing. In a situation like this , the optimal move for the defender is to also not take any action. This incurs a cost of 0.0 for both the attacker and the defender.

Strategy 2 : This strategy is when the attacker activates A2. The defender must have activated the defenses D2_2 and D2. This incurs a cost of 9 for the defender which is the sum of the costs of activating D2 and D2_2. It incurs a cost of 15.0 for the attacker which is the cost of attacking the node A2.

Strategy 3 : This strategy is when the attacker activates A1. The defender must have activated either the defense D1_2 or D1. The optimal choice for the defender here is activating D1_2 since it costs only 2 compared to D1 costing 10. This incurs an attacker cost of 20.0 and a defender cost of 2.0. The graph with the points in the table can be seen in Fig. 10

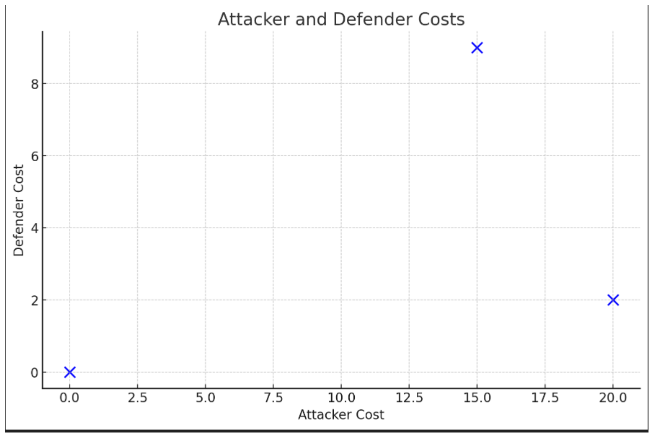


Fig. 10. Attacker and Defender costs

7 CONCLUSION

This research has demonstrated the potential of combining game theory with Attack Defense Trees (ADTs) and the PRISM model checker to optimize security strategies. By translating ADTs into PRISM-compatible models and utilizing game theory solvers, this study has identified cost-effective defense strategies, providing valuable insights for cybersecurity practitioners. The study established that game theory solvers are practical and effective tools for analyzing ADTs. These solvers can handle the complexities and nuances of ADTs, providing a comprehensive view of the strategic interactions between attackers and defenders.By translating ADTs into PRISM-compatible models, the research identified cost-effective security strategies that balance the trade-offs between minimizing defense costs and maximizing the effectiveness of security measures.

7.1 Limitations

Despite the promising results, there are several limitations to this study:

- **Cost Assumptions:** The research assumes that the costs of attacks and defenses are known and accurate. In real-world scenarios, these costs can be dynamic and difficult to estimate.
- **Scalability:** While the approach is effective for smaller ADTs like the one analyzed in this paper, the computational complexity increases significantly with the size of the tree. This may limit its applicability to larger, real-world systems.
- **Static Analysis:** The study performs a static analysis of ADTs without considering the evolving nature of cyber threats. Real-time or dynamic analysis could provide security strategies more applicable to real life security scenarios.
- **Multi-objective Optimization:** While the study considered cost-effectiveness, other factors such as probability of successful attack were not included. Multi-objective optimization approaches could provide strategies more applicable to real life scenarios.

7.2 Future Work

To address the limitations and build on the findings of this research, a few ideas for future work are proposed :

- **Scalability Enhancements:** Enhancing the scalability of the proposed methodologies is crucial. This could be achieved by optimizing the algorithms used for translating ADTs into PRISM-compatible models
- **Implement Multi-objective optimization:** Properties of actions other than costs such as probability of success could be analyzed.
- **Automate the translation process:** A tool could be developed which takes simple text as input and outputs the correct PRISM code.

8 AI STATEMENT

During the preparation of this work the author used AI tools including ChatGPT and Grammarly in order to enhance the readability of this research paper. After using this tool/service, the author reviewed and edited the content as needed and takes full responsibility for the content of the work.

REFERENCES

[1] Game theory for security and risk management. In Static & Dynamic Game Theory: Foundations & Applications. Springer, 2018.
[2] Prism manual. https://www.prismmodelchecker.org/manual/Main/Welcome, n.d. Accessed: 2024-06-17.
[3] Z. Aslanyan and F. Nielson. Pareto efficient solutions of attack-defence trees. In R. Focardi and A. Myers, editors, Principles of Security and Trust, pages 95–114, Berlin, Heidelberg, 2015. Springer Berlin Heidelberg.
[4] S. Bistarelli, M. Dall'Aglio, and P. Peretti. Strategic games on defense trees. pages 1–15, 08 2006.
[5] T. Chen, V. Forejt, M. Kwiatkowska, D. Parker, and A. Simaitis. Prism-games: A model checker for stochastic multi-player games. In N. Piterman and S. A. Smolka, editors, Tools and Algorithms for the Construction and Analysis of Systems. TACAS 2013, volume 7795 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2013.
[6] J. Eisentraut and J. Křetínský. Expected cost analysis of attack-defense trees. In D. Parker and V. Wolf, editors, Quantitative Evaluation of Systems. QEST 2019, volume 11785 of Lecture Notes in Computer Science. Springer, Cham, 2019.
[7] Identity Theft Resource Center. 2023 annual data breach report. Retrieved from https://www.idtheftcenter.org/wp-content/uploads/2024/01/ITRC_2023-Annual-Data-Breach-Report.pdf, 2023.
[8] B. Kordy, S. Mauw, M. Melissen, and P. Schweitzer. Attack–defense trees and two-player binary zero-sum extensive form games are equivalent. In T. Alpcan, L. Buttyán, and J. S. Baras, editors, Decision and Game Theory for Security. GameSec 2010, volume 6442 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2010.
[9] B. Kordy, S. Mauw, S. Radomirović, and P. Schweitzer. Attack-defense trees. Journal of Logic and Computation, 24(1):55–87, 2012.
[10] M. Kwiatkowska, G. Norman, and D. Parker. Prism: Probabilistic symbolic model checker. In T. Field, P. G. Harrison, J. Bradley, and U. Harder, editors, Computer Performance Evaluation: Modelling Techniques and Tools. TOOLS 2002, volume 2324 of Lecture Notes in Computer Science. Springer, Berlin, Heidelberg, 2002.
[11] M. Lopuhaä-Zwakenberg, C. E. Budde, and M. Stoelinga. Efficient and generic algorithms for quantitative attack tree analysis. IEEE Transactions on Dependable and Secure Computing, 20(5):4169–4187, 2023.
[12] H. Yao, Z. Xu, Y. Hou, Q. Dong, P. Liu, Z. Ye, X. Pei, M. Oeser, L. Wang, and D. Wang. Advanced industrial informatics towards smart, safe and sustainable roads: A state of the art. Journal of Traffic and Transportation Engineering (English Edition), 10, 03 2023.