Formalizing Real-world Threat Scenarios

Paul Tavolato[1], Robert Luh[1,2] and Sebastian Eresheim[1,2]
[1]*Research Group Security and Privacy, University of Vienna, Kolingasse 14-16, A-1090 Vienna, Austria*
[2]*Department of Computer Science, UAS St. Pölten, A-3100 St. Pölten, Austria*

Keywords: Threat Analysis, Formal Methods, Stochastic Game Theory, Model Checking.

Abstract: Using formal methods in threat analysis would be of great benefit to securing modern IT systems. To this end a strictly formal description of attacker-defender scenarios is vital. This paper demonstrates how attacker and defender behavior and its interrelationship can be defined using Markov decision processes and stochastic game theory. Based on these definitions, model checking methods can be applied to find quantitative answers to important questions relevant in threat analysis. A main focus lies on the applicability of the method to real-world situations. This is accomplished by incorporating information from several proven tactical and technical knowledge bases. Practicability of the method is shown by using the model checking tool PRISM-games.

1 INTRODUCTION

Fighting cyber security threats has become an inevitable must for every organization, may it be small-scale businesses, public service providers, or multinational corporations. Since 100% security cannot be achieved in reality, a detailed look at the effectiveness of counteractive measures is vital in providing the best possible defense within a given budget. Alternatively, it might be necessary to determine the exact budget needed to guarantee a certain level of security. One of the fundamental operations in this context is threat analysis.

In the realm of cyber security, threat analysis is understood as the process of assessing the activities and capabilities of unknown intelligence entities or criminals. Cybersecurity threats can be defined as a malicious act that seeks to disrupt proper operation of IT systems by violating one or more of the central cybersecurity properties: Confidentiality (by stealing data), Integrity (by manipulating data), and Accessibility (by disrupting or rendering impossible the normal flow of data processing). These principals of protection are known as the CIA triad (Keyser, 2018). In a broader understanding threat analysis also comprises the process used to determine which components of the system need to be protected, as well as the types of threats (security risks) they should be protected from. This information is necessary to determine strategic locations in the IT architecture and to design reasonable and effective security measures to be implemented (McCabe, 2007).

Several methods for threat analysis have been proposed in the literature, including STRIDE and Pasta (see for example (Shostack, 2014), (Tarandach & Coles, 2020), or (Swiderski & Snyder, 2004)). Though all these methods are very useful, they lack an essential property: they are mainly of an informal or semi-formal nature. If they contain some formal parts, these are not strictly formal in a mathematical sense. Therefore, they cannot be used in connection with formal methods. By formal methods we understand the use of mathematically rigorous techniques for modeling a system and the formal definition of properties in some sort of logic, e.g., temporal logic. The validity of these properties can then be proved automatically by the use of software tools. Such proofs are often carried out by a method called model checking. For cyber security formal methods have been used so far almost exclusively in the realm of software development, especially with regard to the development of safety-critical systems. Using formal methods with threat analysis on a higher system level, however, is still a marginal topic in the scientific literature. Our main focus lies on the analysis of existing IT systems at the users' site. The user in this scenario has no possibility to change the software itself. S/he is only able to set and enforce security measures that will detect, mitigate or – at best – prevent attacks on her/his system.

Tavolato, P., Luh, R. and Eresheim, S.
Formalizing Real-world Threat Scenarios.
DOI: 10.5220/0010781300003120
In *Proceedings of the 8th International Conference on Information Systems Security and Privacy (ICISSP 2022)*, pages 281-289
ISBN: 978-989-758-553-1; ISSN: 2184-4356

This paper revolves around applying formal methods in threat analysis: The system, consisting of both the attacker and the defender side is modeled in a mathematically rigorous way. Due to the evident antagonism between defender and attacker, concepts from non-cooperative game theory are used, assuming the attack on a system is a game between the system owner, the defender, and a hacker, the attacker. Such a game contains non-determinism: the attacker as well as the defender can choose among a set of available actions (restricted by insight, skill and budget). They are also of a stochastic nature: the success or failure of an action is determined by a success probability. Given such a rigorous formal definition of the attacker-defender system (the game), one can formulate properties that represent essential questions such as:

- What is the maximum probability that an attacker with a specific goal and profile will ultimately succeed?
- What is the maximum defender success probability with a given defender budget in the given situation?
- How does a significant lower defender budget influence the defender's minimum success probability?

To answer these and similar questions we will use formal methods of stochastic model checking.

Semantically the model is based on an adversarial cyber security game for threat assessment called PenQuest (Luh, Temper, Tjoa, Schrittwieser, & Janicke, 2019). In this role-playing game (RPG) two players, the attacker and the defender, fight against each other in order to achieve their respective goal: The attacker has a predefined goal (violating one part of the CIA triangle) and the defender has a given infrastructure he wants to defend against attacks. The game is characterized by its high degree of practical relevance, mimicking real-life situations in cyber security as close as possible. This is accomplished by taking information from several proven data sources such as STIX (Structured Threat Information eXpression language) (MITRE Corporation, D), the APT kill chain by Hutchinson (Hutchins, Cloppert, & Amin, 2011), the CAPEC (Common Attack Pattern Enumeration and Classification) attack patterns (MITRE Corporation, A), the MITRE ATT&CK (Adversarial Tactics, Techniques & Common Knowledge) attack and mitigation patterns (MITRE Corporation, B), the NIST SP 800-53 Countermeasures (Joint Task Force Transformation Initiative, 2015), and MITRE D3FEND (MITRE Corporation, C).

The main tasks in formalizing realistic threat scenarios comprise the modeling of attacker behavior along a kill chain, of defender behavior, and of the interplay between the two.

2 RELATED WORK

As mentioned above, most of the work on formal methods and security focuses on the construction (specification, development, implementation) of secure and safe software, such as the implementation of network protocols, the implementation of secure cryptographic algorithms, or generally the implementation of software without vulnerabilities. For an overview see for example (Chong, et al., 2016) or (Nanda & Jeppu, 2018). This work, however, goes in a different direction.

Some formal aspects of attack modeling are described in (Guelzim & Obaidat, 2015). For our purposes, the concept of attack-defense trees (ADT) used in threat analysis is of particular importance. Initially of only semi-formal nature, more formal treatments of ADTs were developed in recent years. Wideł et al. (Wideł, Audinot, Fila, & Pinchinat, 2019) provide an overview of the use of formal models in security; in (Aslanyan, Nielson, & Parker, 2016) the authors describe a formal system in order to analyze quantified properties. Attack trees alone (without defense actions) are discussed in (Gadyatskaya, et al., 2016), where Priced Timed Automata are used to formalize attack trees and model checking is performed using the tool Uppaal CORA; however, no stochastic aspects were used in this approach. Another formal treatment of attack trees is discussed in (Pekergin, Tan, & Fourneau, 2016), where a stochastic analysis is described, but no model checking approaches are used. In (Buldas, Gadyatskaya, Lenin, Mauw, & Trujillo-Rasua, 2020) constraint programming is used to evaluate attack trees with incomplete information.

The idea of using stochastic game theory (Shapley, 1953) in connection with cyber security dates back to at least 2002 (Hamilton, Miller, Ott, & Saydjari, 2002). The equivalence of attack-defense trees and game theory was formally proved by Kordy et al. (Kordy, Mauw, Melissen, & Schweitzer, 2010). There are several papers about the application of game theory to various aspects of security: a good overview, though restricted to cyber-physical systems, is given in (Etesami & Basar, 2019). Other game-theoretic approaches are discussed in (Bommannavar, Alpcan, & Bambos, 2011), (He, Zhuang, & Rao, 2012), (Luo, Szidarovszky, Al-

Nashif, & Hariri, 2010), (Nguyen, Alpcan, & Başar, 2009), (Sallhammar, Helvik, & Knapskog, 2006), (Sajjan, Sankardas, & Dipankar, 2010), (Tabatabaei, 2016), (Zhang, Wang, & Zhuang, 2021), (Li, Peng, Zhu, & Basar, 2021), (Jie, Choo, Li, Chen, & Guo, 2019), and (Han, Niyato, Saad, & Başar, 2019). However, all these works use game-theoretic concepts only and do not combine them with formal methods (in the sense as described in the introduction).

Nevertheless, there are a few papers combining game theory and model checking, all of them coming from a group at the University of Oxford: Kwiatkowska (Kwiatkowska, Model Checking and Strategy Synthesis for Stochastic Games: From Theory to Practice, 2016) gives a good summary of the approach, while (Kwiatkowska, Norman, & Parker, Verication and Control of Turn-Based Probabilistic Real-Time Games, 2019), (Svorenova & Kwiatkowska, 2016), (Chen, Forejt, Kwiatkowska, Parker, & Simaitis, 2013) and (Simaitis, 2013) cover various details; Wiltsche (Wiltsche, 2015) uses model checking in the frame of a game-theoretic model for strategy synthesis. All the papers give some small examples to illustrate the viability of the method, but do not cover threat analysis for real-world situations. This group of researchers has implemented a stochastic model checking tool with an extension for game models: PRISM-games (PRISM-games), which we will use in our approach.

3 FORMALIZING ATTACKER AND DEFENDER BEHAVIOR

3.1 Environment Information

In order to formalize the behavior of each party, a number of facts about the environment must be taken into consideration:

- The configuration of the system: this information is initially only available to the defender in its entirety; the attacker may have a restricted view of the system's exposed topology and access points. During the attack process, the attacker may gain more and more insight into the victim's configuration.
- The goal of the attacker: this information is only available to the attacker in the beginning; again, during the course of events the defender may become more and more aware of the attacker's goal and progress achieved so far.
- The possible actions available to the attacker and the defender, respectively.
- The skill level of the attacker and the defender: the skill level influences the availability of certain actions and their success probabilities.
- The budget of the attacker and the defender: the budget, too, limits the availability of actions.
- The attacker's initiative determines the number of actions he can perform. It generally decreases during the game. If, for example, the initiative reaches 0, the attack stops (the attacker resigns).

3.2 Formalizing Attacker Behavior

Based on the definition of the environment, the skill level and the available budget, we describe attacker behavior as an adapted form of a Markov Decision Process (MDP). Attacker behavior is formalized as a 4-tuple $MDP_{att} = (S_{att}, A_{att}, P, C_a)$ where:

$S_{att} = \{s_0, s_1, \dots s_n\}$	is a finite set of states; s_0 is defined as the starting state.
$A_{att} = \{a_1, a_2, \dots a_n\}$	is a finite set of attacker actions.
$P: S \times A_{att} \times S \rightarrow [0..1]$	P(s, a, s') is the probability that action a executed in state s will lead to state s'.
$C_a(s, s')$	defines the consequences of a transition from state s to state s' due to action a.

Simply put: Starting from state s_0 the attacker non-deterministically chooses an executable action from A_{att}: an action is executable in state s if a predefined condition evaluates to true. In other words, an action is executable if it matches the skill level (for some more complex actions a higher skill level is required), if it can be executed within the attacker's budget (special tools might be necessary for the action, which must be purchased), if it fits to the stage of the game (some actions can only be executed if other preceding actions were successful), and if it aims at a part of the configuration already known to the attacker (the attacker has to gain insight into the configuration before being capable of attacking internals). After execution of the chosen action the successor state is determined based on the probability function P that defines the success probability of the action. For reasons of simplicity, there are only two possible successor states depending on the success or failure of the action. Finally, the function $C_a(s,s')$ defines the consequences of the state transition initiated by action a. Such consequences may be a change in budget or in initiative, an impact on the attacker's insight into the defenders configuration, some damage to the defender's configuration, or changes of P for future

transitions. A state transition together with its consequences is referred to as a move.

3.3 Formalizing Defender Behavior

Defender behavior is formalized in the same way as attacker behavior by means of a Markov Decision Process. $MDP_{def} = (S_{def}, A_{def}, P, C_a)$ where:

$S_{def} = \{s_0, s_1, \dots s_n\}$	is a finite set of states; s_0 is defined as the starting state.
$A_{def} = \{a_1, a_2, \dots a_n\}$	is a finite set of defender actions.
$P: S \times A_{def} \times S \rightarrow [0..1]$	P(s, a, s') is the probability that action a executed in state s will lead to state s'.
$C_a(s, s')$	defines the consequences of a transition from state s to state s' due to action a.

The definitions correspond intuitively to the definitions of the attacker *MDP*, especially concerning skill level, budget and the like. The concept of insight plays a somewhat different role in the defender definition: initially, the defender does not even know that he is being attacked. In this phase the defender can only take preventive actions of a general nature; therefore we call this phase the prevention phase. However, a possible consequence of a defender action may be the detection of an ongoing attack – which moves the situation from the prevention phase to the response phase and can trigger a significant change in the availability and the success probability of further defender actions.

4 FORMALIZING THE INTERRELATIONSHIPS BETWEEN ATTACKER AND DEFENDER

The next step in formalizing a realistic threat scenario is the combination of MDP_{att} and MDP_{def}. The obvious mathematical constructs for this come from game theory. The following general assumptions about the characteristics of this game are:

- The game is **non-cooperative** (quite obvious).
- It is a **two-player game** with attacker and defender (obvious, too).
- It is an **extended form** of a **zero-sum game**. There is always a winner and a looser, though the rewards may not match exactly: either the attacker achieves his goal and wins, or the defender wins, if the attacker's initiative reaches zero (he „gives up").
- The game includes **non-deterministic** elements: each player can in most situations choose between a number of possible actions (non-deterministically) restricted only by some predefined constraints.
- It is a **stochastic** game: the success of a player's action is defined by a probability distribution.
- It is a game with **imperfect information**: only a very restricted set of information (or no information at all) about the current state of the play is seen by the players; during the game, parts of this information may become unveiled.
- It is a game with **incomplete information**: the players can only guess about the way the game is played (which means that not only the state of the game, but objectives and strategies, too, are unknown; even the rules of the game are obscured). In the beginning of the game, for example, the defender does not even know that the game has already started, and an attack is well under-way; we call this the prevention phase of the game. When information about the attack is at least partly unveiled to the defender, we say that the defender moves from the prevention phase to the response phase.
- For the sake of simplicity we restrict the game to be **turn-based**: only one player can make a move at a time. Furthermore the moves are executed alternately as defined in PenQuest's game mechanics (Luh, Temper, Tjoa, Schrittwieser, & Janicke, 2019). This property makes the game an extensive form game: the sequence of moves can be structured in a tree-like manner. For future work, this restriction could be loosened: the first move (or even more moves) could be carried out in parallel.

Under these assumptions we can formalize attacker-defender scenarios as stochastic 2-player games. To improve comprehensibility we will use an extension described by Marta Kwiatkowska (Kwiatkowska, Model Checking and Strategy Synthesis for Stochastic Games: From Theory to Practice, 2016), called a 2½-player game, where probabilistic states are introduced between the attacker and defender states to represent the probabilistic transitions explicitly. This is without loss of generality as in the above mentioned paper Kwiatkowska has shown that for any stochastic 2½-player game there is an equivalent 2-player game without explicit probability states. In a 2½-player game the states of the game are composed of the attacker states (S_{att}), and the defender states (S_{def}), each controlled by the

respective players, as well as probabilistic states (S_{prob}). Starting from one of the player states, say the attacker, the player chooses one of his available actions and the model moves to a probabilistic state. From the probabilistic state usually two different successor states to the set of defender states can follow depending on the probability of the action's success: one following the success of the action (with probability p) and one when the action failed (with probability 1-p). The sequence of game states then is $s_{att} - s_{prob} - s_{def} - s_{prob} - s_{att} - s_{prob} \ldots$ with $s_{att} \in S_{att}$ being an attacker state, $s_{prob} \in S_{prob}$ being a probabilistic state and $s_{def} \in S_{def}$ being a defender state.

We now define the stochastic 2½-player game: $\Gamma_{PQ} = (\Pi, S, s_0, \delta, A, \alpha, C, \beta)$

$\Pi = \{att, def\}$	the players: attacker and defender
$S = S_{att} \cup S_{def} \cup S_{prob}$	a finite non-empty set of states partitioned in three subsets: S_{att} the attacker states S_{def} the defender states S_{prob} the probabilistic states. $S_{player} = S_{att} \cup S_{def}$
$s_0 \in S$	the initial state
$\delta: S \times S \rightarrow [0,1]$	a stochastic transition function assigning probability values to state transitions. Player states map on probabilistic states: $\delta(s, s') \in \{0,1\}$ $\delta(s, s') = 1$ for at least one $s' \in S_{prob}$. Transitions from probabilistic states to player states are assigned a success probability; for all $s \in S_{prob}$ holds $\sum_{s' \in S} \delta(s, s') = 1$
$A = A_{att} \cup A_{def}$	a finite set of actions of the players
$\alpha: S_{player} \times S_{prob} \rightarrow A$	a function assigning the selected action to the transitions from the players to the probabilistic states.
C	a finite set of consequences
$\beta: S_{prob} \times S_{player} \rightarrow C$	a function assigning the consequences of an action to the transitions from a probability state to the succeeding attacker or defender state

The transition function δ defines the success/ failure probabilities in case of transitions from a probabilistic state to a player state; in case of the transition from a player state to a probabilistic state, it simply defines which states are possible (1) and which not (0). In the latter case and if $\delta(s_i, s_{prob}) = 1$ the function α assigns the action selected by player i to the transition. The function β defines the consequences of an action (usually only if the action is successful). Such consequences may be changes in insight, damage, or changes in success probabilities or in availability of future actions.

5 EXAMPLE

5.1 Modeling

Let's assume the following scenario: an attacker wants to disrupt the victim's service operation by either encrypting some of the victim's essential data or by arbitrarily changing or even deleting this data. We will give here only a high-level overview with a restricted number of actions. Attacker activities are structured into phases by kill chains. A simplified kill chain for this attack would consist of three phases:

1. Reconnaissance
2. Gain Access
3. Destroy Data

For each of these stages the following actions are available to the attacker:

1. Reconnaissance: use open-sources intelligence, execute a vulnerability scan, perform an active scan
2. Gain Access: phishing, remote server connection, brute force password cracking
3. Destroy Data: data encryption, data manipulation, wipe disk

For each attacker activity we must define possible target assets, required skills, the necessary budget, the effects of a successful action on insight in the defender's configuration and on the damage status with regard to CIA, and the success probability. The skill requirements are rated at a scale from 1 to 5, according to the STIX threat actor vocabulary *ThreatActorSophistication*. The available budget is rated on a scale from 0 to 10, insight and damage are rated on a scale from 1 to 3 Tables 1 to 3 show these assignments for the attacker actions.

Table 1: Attacker Stage 1 – Reconnaissance Actions.

	TargetAssets	SkillReq	Budget	Insight	Damage CIA	SuccessProb
OpenSrcInt	WebServer MailServer ApplicationServer MobileDevices Database FileServer ClientPC CloudApplication DMZ LAN	1	-	+1	0,0,0	0.8
VulnScan	Same as above	2	-	+1	1,0,1	0.6
ActiveScan	Same as above	3	-	+1	1,0,1	0.5

Table 2: Attacker Stage 2 – Access Actions.

	TargetAssets	SkillReq	Budget	Insight	Damage CIA	SuccessProb
Phishing	MailServer ClientPC	2	2	+1	2,2,1	0.6
Remote Server Connection	WebServer MailServer ApplicationServer Database FileServer CloudApplication	2	-	-	2,2,1	0.5
Brute Force	Same as above	1	1	-	2,2,1	0.5

Table 3: Attacker Stage 3 – Destruction Actions.

	TargetAssets	SkillReq	Budget	Insight	Damage CIA	SuccessProb
Encryption	WebServer MailServer ApplicationServer MobileDevices Database FileServer ClientPC CloudApplication DMZ LAN	3	-	-	0,0,3	0.3
Manipulation	Same as above	3	-	-	0,3,1	0.3
Wipe Disk	Same as above	3	-	-	0,0,3	0.3

The defender activities are structured into two phases: the prevention phase and the response phase. There are the following actions at disposal in the respective stages:

1. Prevention phase: security awareness training, security response training, incident monitoring, mobile code restrictions, two-factor authentication, logon attempts restrictions, restrict external connections
2. Response phase: information spillage handling fail-safe procedures, asset recovery

The definitions of target assets, required skills, budget, insight into the attacker's intent, and success probabilities are shown in Tables 4 and 5. Moreover, a defense action can change the success probability of a defender action. A security awareness training, for example, reduces the success probability of a phishing attack (table 2, first line) by half.

A formal model as described above can be formulated in a model checking tool. To this end we chose PRISM-games (Kwiatkowska, Parker, & Wiltsche, PRISM-games: Verification and Strategy Synthesis for Stochastic Multi-player Games with Multiple Objectives, 2018). This tool allows, among others, for the definition of different players and modules. The modules contain the actions and their consequences and can be written to mimic the non-determinism as well as the stochastic nature of the activities derived from the success probabilities.

Equipped with this information, one can define the game: Foremost, we must decide on the parameters characterizing the players: their skill levels, their budgets and the attacker's initiative. The attacker and defender states correspond to the phases of the attack and the defense, the probabilistic states between an attacker and a defender state decide on the success or failure of an action. The transition function δ defines the eligibility of an action in a certain state (depending on skill, budget and other preconditions) when going from a player state to a probabilistic state and the success probability otherwise. The set of actions is given by the tables and the function α non-deterministically decides on the choice of an action by a player. The consequences of a successful action and the function β are defined in the tables, too.

Table 4: Defender Stage 1 – Prevention Phase.

	TargetAssets	SkillReq	Budget	Insight	ChangeAttProbs	SuccessProb
Security Awareness Training	Users	1	1	false	St2/1→ /2	0.8
Security Response Training	Staff	2	1	false		0.8
Incident Monitoring	System	2	2	true		0.7
Mobile Code Restrictions	Mobile	2	1	false		0.7
Two-Factor Authentication	System, Users	2	2	false	St2/3→0.01	0.6
Logon Attempts Restrictions	System	1	1	false	St2/3→0	0.6
Restrict External Connections	System	2	1	false	St2/2→0.1	0.6

Table 5: Defender Stage 2 – Response Phase.

	TargetAssets	SkillReq	Budget	Insight	ChangeAttProbs	SuccessProb
Information Spillage Handling	System	2	2	true		0.7
Fail-Safe Procedures	System, Staff	**3**	**2**	true		0.4
Asset Recovery	System, Staff	**3**	**3**	true		0.5

The current version of the PenQuest model contains approximately 250 different attack actions and 70 defense actions that can be used in modeling attacker and defender strategies. So far we use a subset of these actions in our model checking approach.

5.2 Applying Model Checking Techniques

Having modeled the example scenario as a game in the tool PRISM-games, quantitative questions about the threat situation can be formulated. The answers to these questions can then be calculated by model checking. Model checking evaluates all possible paths the game could take and on the way collects the necessary information to compute maximum and minimum success probabilities of the attacker and the defender. In PRISM-games these questions are formulated in PRISM's property specification language, which subsumes several well-known probabilistic temporal logics.

The questions from the introduction were answered like follows:

- What is the maximum probability that an attacker with a specific goal and profile will ultimately succeed?
  Given an attacker initiative of 4 (on a scale from 1 to 10) and a defender budget of 7 (on a scale from 1 to 10) model checking yield 0.42 in the example, meaning that in the given situation the attacker has a 42%-chance of success at best; while his minimum success probability is 0.0 (no success)
- What is the maximum defender success probability in the given situation with a defender budget of 7 and an attacker initiative of 4, as above?
  Model checking results yield 0.93 in the example, meaning that the residual rest of risk against an attacker with initiative 4 is rather small in the given situation.
- How does a significant lower defender budget influence the defender's minimum success probability?
  Model checking results show that lowering the defender budget from level 7 to level 5 does not influence the defender's success probabilities; lowering it to level 4, however, significantly

reduces the minimum probability of defender success (0.40 instead of 0.58).

In the example – as implemented in our model – the most important parameters are the attacker's initiative and the defender's budget. The attacker's initiative models his persistence: a lower value lets him give up earlier and hence lowers his success chances. While the defender's budget influences a situation only as long as it makes sense to invest. Spending money for security measures that do not have any influence in the modeled scenario will not change the probabilities.

6 CONCLUSIONS AND FUTURE WORK

The formalism for modeling attack-defense scenarios as described in this paper makes threat analysis possible in a strictly formal way and thus renders possible the application of formal methods and tools. The main innovation here is that the combination of game theory and model checking is applied to tactical threat analysis for real-world situations. To keep the analysis as close to practice as possible we collected attack and defense actions from accepted data sources and vocabularies. As mentioned above the model so far contains approximately 250 different attack actions and 70 defense actions that can be used in modeling attack and defender strategies. Subsuming all these aspects into one single model would clearly lead to a too large model, and hence render model checking impossible by leading to state explosions. To keep the model in a manageable size we break down the model into scenarios along the lines of kill-chains (Hutchins, Cloppert, & Amin, 2011). Model checking these scenarios is possible on standard PCs with 4.6 GHz and 32GB RAM. For more elaborate scenarios more computing power will be necessary.

Future work will consider parallel execution of attacker and defender actions in certain situations, especially in the beginning. This would mean giving up the turn-based property of the game partially. So far, the consequences of this are not yet analyzed fully. Another idea for future development is strategy synthesis (for defenders): Given a configuration with real-world limitations such as a budget, skill as well as anticipated threats, the optimal defense strategy could be computed.

REFERENCES

Aslanyan, Z., Nielson, F., & Parker, D. (2016). Quantitative Verification and Synthesis of Attack-Defence Scenarios. IEEE 29th Computer Security Foundations Symposium, CSF 2016, (S. 105-119). doi: 10.1109/CSF.2016.15

Bommannavar, P., Alpcan, T., & Bambos, N. (2011). Security risk management via dynamic games with learning. IEEE International Conference on Communications (ICC), (S. 1-6). doi: 10.1109/icc.2011.5963330

Buldas, A., Gadyatskaya, O., Lenin, A., Mauw, S., & Trujillo-Rasua, R. (2020). Attribute Evaluation on Attack Trees with Incomplete Information. Computers and Security 88/101630.

Chen, T., Forejt, V., Kwiatkowska, M., Parker, D., & Simaitis, A. (2013). Automatic Verification of Competitive Stochastic Systems. Formal Methods in System Design, 43/1, S. 61-92.

Chong, S., Guttman, J., Datta, A., Myers, A., Pierce, B., Schaumont, P., Zeldovich, N. (2016). Report on the NSF Workshop on Formal Methods for Security. https://dash.harvard.edu/bitstream/handle/1/34604536/6897521.pdf

Etesami, S. R., & Basar, T. (2019). Dynamic Games in Cyber-Physical Security: An Overview. Dynamic Games and Applications, S. 884–913. doi: doi.org/10.1007/s13235-018-00291-y

Gadyatskaya, O., Hansen, R. R., Larsen, K. G., Legay, A., Olesen, M., & Poulsen, D. B. (2016). Modelling Attack-defense Trees Using Timed Automata. In M. Fränzle, & N. Markey (Hrsg.), FORMATS 2016, LNCS 9884. 9884, S. 35–50. Springer LNCS. doi: 10.1007/978-3-319-44878-7 3

Guelzim, T., & Obaidat, M. S. (2015). Formal methods of attack modeling and detection. In Modeling and Simulation of Computer Networks and Systems - Methodologies and Applications (S. 841-860). Elsivier.

Hamilton, S., Miller, W., Ott, A., & Saydjari, O. (2002). The role of game theory in information warfare. 4th Information Survivability Workshop, (S. 1-4).

Han, Z., Niyato, D., Saad, W., & Başar, T. (2019). Game Theory for Next Generation Wireless and Communication Networks - Modeling, Analysis, and Design. Cambridge University Press.

He, F., Zhuang, J., & Rao, N. (2012). Game Theoretic Analysis of Attack and Defense in Cyber-Physical Network Infrastructures. Proceedings of the Industrial and Systems Engineering Research Conference. doi: 10.1.1.719.2652

Hutchins, E., Cloppert, M., & Amin, R. (2011). Intelligence-driven computer network defense informed by analysis of adversary campaigns and intrusion kill chains. Lead. Issues Inf. Warf. Secur. Res. 1/80.

Jie, Y., Choo, K.-K. R., Li, M., Chen, L., & Guo, C. (2019). Tradeoff gain and loss optimization against man-in-the-middle attacks based on game theoretic model. Future Generation Computer Systems, Volume 101, S. 169-179. doi: doi.org/10.1016/j.future.2019.05.078

Joint Task Force Transformation Initiative. (2015). SP 800-53 rev. 4. Recommended Security Controls for Federal Information Systems and Organizations. Gaithersburg.

Keyser, T. (2018). Security policy. In The Information Governance Toolkit (S. 57-62). CRC Press.

Kordy, B., Mauw, S., Melissen, M., & Schweitzer, P. (2010). Attack–Defense Trees and Two-Player Binary Zero-Sum Extensive Form Games Are Equivalent. In T. Alpcan, L. Buttyán, & J. Baras (Hrsg.), Decision and Game Theory for Security. GameSec 2010. . Lecture Notes in Computer Science, vol 6442. Springer. doi: 10.1007/978-3-642-17197-0_17

Kordy, B., Mauw, S., Radomorivic, S., & Schweitzer, P. (2014). Attack–Defense Trees. Journal of Logic and Computation, Volume 24/1, S. 55–87. Von http://logcom.oxfordjournals.org/cgi/reprint/exs029?

Kwiatkowska, M. (2016). Model Checking and Strategy Synthesis for Stochastic Games: From Theory to Practice. Proceedings of the 43rd International Colloquium on Automata, Languages, and Programming (ICALP'16). Schloss Dagstuhl - Leibniz-Zentrum fuer Informatik.

Kwiatkowska, M., Norman, G., & Parker, D. (2019). Verication and Control of Turn-Based Probabilistic Real-Time Games. The Art of Modelling Computational Systems: A Journey from Logic and Concurrency to Security and Privacy, volume 11760 of LNCS, S. 379-396.

Kwiatkowska, M., Parker, D., & Wiltsche, C. (4 2018). PRISM-games: Verification and Strategy Synthesis for Stochastic Multi-player Games with Multiple Objectives. International Journal on Software Tools for Technology Transfer, 20(2), S. 195–210.

Li, T., Peng, G., Zhu, Q., & Basar, T. (2021). The Confluence of Networks, Games and Learning - A game-theoretic framework for multi-agent decision making over networks. IEEE control system magazine, special issue on Distributed Nash Equilibrium Seeking over Networks.

Luh, R., Temper, M., Tjoa, S., Schrittwieser, S., & Janicke, H. (2019). PenQuest: a gamified attacker/defender metamodel for cyber security assessment and education. Journal of Computer Virology and Hacking Techniques. doi: https://doi.org/10.1007/s11416-019-00342-x

Luo, Y., Szidarovszky, F., Al-Nashif, Y., & Hariri, S. (2010). Game Theory Based Network Security. Journal of Information Security 1/1, S. 41-44. doi: 10.4236/jis.2010.11005

McCabe, J. D. (2007). Network Analysis, Architecture, and Design. Morgan Kaufmann.

MITRE Corporation. (A). CAPEC—Common Attack Pattern Enumeration and Classification. https://capec.mitre.org/

MITRE Corporation. (B). MITRE ATT&CK. https://attack.mitre.org/

MITRE Corporation. (C). MITRE D3FEND. https://d3fend.mitre.org/resources/D3FEND.pdf

MITRE Corporation. (D). STIX—Structured Threat Information Expression | STIX Project Documentation. https://oasis-open.github.io/cti-documentation/

Nanda, M., & Jeppu, Y. (2018). Formal Methods for Safety and Security. Springer.

Nguyen, K. C., Alpcan, T., & Başar, T. (2009). Stochastic games for security in networks with interdependent nodes. Proceedings of the 2009 International Conference on Game Theory for Networks, GameNets '09.

Pekergin, N., Tan, S., & Fourneau, J.-M. (2016). Quantitative Attack Tree Analysis: Stochastic Bounds and Numerical Analysis. International Workshop on Graphical Models for Security, GraMSec 2016. doi: DOI: 10.1007/978-3-319-46263-9_8

PRISM-games. https://www.prismmodelchecker.org/games/

Sajjan, S., Sankardas, R., & Dipankar, D. (2010). Game theory for cyber security. CSIIRW '10: Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research. doi: doi.org/10.1145/1852666.1852704

Sallhammar, K., Helvik, B. E., & Knapskog, S. J. (2006). On stochastic modeling for integrated security and dependability evaluation. Journal of Networks 1/5, S. 31-42. doi: 10.4304/jnw.1.5.31-42

Shapley, L. S. (1953). Stochastic games. PNAS 39/10, S. 1095-1100.

Shostack, A. (2014). Threat Modeling: Designing for Security. Wiley.

Simaitis, A. (2013). Automatic Verification of Competitive Stochastic Systems. Dissertation. University of Oxford.

Svorenova, M., & Kwiatkowska, M. (2016). Quantitative Verification and Strategy Synthesis for Stochastic Games. European Journal of Control 30, special issue 15th European Control Conference, ECC16, S. 15-30. doi: doi.org/10.1016/j.ejcon.2016.04.009

Swiderski, F., & Snyder, W. (2004). Threat Modeling. Microsoft Press.

Tabatabaei, M. (2016). Games and Strategies in Analysis of Security Properties. Dissertation. Université du Luxembourg.

Tarandach, I., & Coles, M. J. (2020). Threat Modeling: A Practical Guide for Development Teams. O'Reilly.

Wideł, W., Audinot, M., Fila, B., & Pinchinat, S. (2019). Beyond 2014: Formal Methods for Attack tree-based Security Modeling. ACM Computing Surveys. 52/4, Article 75. doi: https://doi.org/10.1145/3331524

Wiltsche, C. (2015). Assume-Guarantee Strategy Synthesis for Stochastic Games. Dissertation. University of Oxford.

Zhang, J., Wang, Y., & Zhuang, J. (2021). Modeling multi-target defender-attacker games with quantal response attack strategies. Reliability Engineering & System Safety, Volume 205.