THESIS ON INFORMATICS AND SYSTEM ENGINEERING C108

Reliable and Efficient Determination of the Likelihood of Rational Attacks

ALEKSANDR LENIN

TUT PRESS

Tallinn University of Technology
Faculty of Information Technology
Department of Informatics

Dissertation was accepted for the defense of the degree of Doctor of Philosophy in department of informatics on November 13, 2015

Supervisor:	Prof. Dr. Ahto Buldas, Chair of Information Security, Department of Informatics, Faculty of Information Technology, Tallinn University of Technology
Opponents:	Prof. Dr. Sjouke Mauw, Chair of Security and Trust of Software Systems, Faculty of Science, Technology and Communication, University of Luxembourg
	Dr. Christian W. Probst, Associate Professor, Department of Applied Mathematics and Computer Science, Technical University of Denmark

Defense of the thesis: December 21, 2015

Declaration:
Hereby I declare that this doctoral thesis, my original investigation and achievement, submitted for the doctoral degree at Tallinn University of Technology has not been submitted for any academic degree.

/Aleksandr Lenin/

This thesis was typeset with XƎLaTeX using the Source Sans Pro typeface.
ISSN 1406-4731
ISBN 978-9949-23-870-5 (print)
ISBN 978-9949-23-871-2 (online)

INFORMAATIKA JA SÜSTEEMITEHNIKA C108

Ratsionaalsete rünnete tõepära efektiivne ja usaldusväärne kindlakstegemine

ALEKSANDR LENIN

TTÜ KIRJASTUS

TO MY FATHER

List of Publications

This dissertation is based on the following publications:

I. Buldas, A., Lenin, A.: New efficient utility upper bounds for the fully adaptive model of attack trees. In: Das, S.K., Nita-Rotaru, C., Kantarcioglu, M. (eds.): GameSec 2013. LNCS 8252, 192–205. Springer (2013)

II. Lenin, A., Willemson, J., Sari, D.: Attacker profiling in quantitative security assessment based on attack trees. In Fischer-Hübner, S., Bernsmed, K. (eds.): NordSec 2014, LNCS 8788, 199–212. Springer (2014)

III. Lenin, A., Buldas, A.: Limiting adversarial budget in quantitative security assessment. In: Poovendran, R., Saad, W. (eds.): GameSec 2014, LNCS 8840, 155–174. Springer (2014)

IV. Pieters, W., Hadziosmanović, D., Lenin, A., Morales, A.L.M., Willemson, J.: TRE$_S$-PASS: Plug-and-play attacker profiles for security risk analysis. In: 35th IEEE Symposium on Security and Privacy, IEEE Computer Society (2014)

V. Lenin, A., Willemson, J., Charnamord, A.: Genetic Approximations for the Failure-Free Security Games. In: Khouzani, M., Panaousis, E., Theodorakopoulos, G. (eds.): GameSec 2015, LNCS 9406, 311–321. Springer (2015)

All publications are reprinted in the appendices of the thesis.

Author's Contribution to the Publications

I. The author suggested and justified the use the expenses propagation method to compute the lower bound of adversarial expenses from which the upper bound of adversarial utility can be derived. The suggested approach turned out to be even more precise than the method which used utility propagation.

II. The author contributed by developing and formalizing the constraint based approach to attacker profiling and created the corresponding genetic algorithm named ApproxTree+ by integrating attacker profiling and budget limitations into the existing ApproxTree algorithm.

III. The author suggested the idea of considering limited adversarial budget in the improved failure-free model, studied the properties of this approach, created corresponding algorithms for analysis of the limited budget models.

IV. The author introduced the idea of the constraint based approach to attacker profiling, as well as contributed to the approach based on Item Response theory by specifying the cost functions to compute adversarial success likelihood.

V. The author has developed the genetic algorithm for the improved failure-free models, set up the experiments to look for empirical evidence of the optimal heuristic criteria for the control parameters of the suggested genetic algorithm.

Abbreviations and Definitions

Abbreviation	Description
WMSAT	Weighted monotone satisfiability
PDAG	Propositional directed acyclic graph
DAG	Directed asyclic graph
BDD	Binary decision diagram
PRNG	Pseudo-random number generator
CEO	Chief executive officer
FAIR	Factor analysis of information risk
CEO	Chief executive officer
RSA	Rivest-Shamir-Adleman (cryptosystem)
ROSI	Return of Security Investment
NP	Nondeterministic polynomial time
PSPACE	The set of decision problems that can be solved by a Turing machine using a polynomial amount of space
SAT	Boolean satisfiability

Symbol	Description
$\mathcal{F}$	Boolean formula
$\mathcal{X}$	The set of variables of $\mathcal{F}$
$\mathcal{P}$	Prize of the satisfiability game
$x_\imath$	A variable in $\mathcal{F}$
$\mathcal{C}_{x_\imath}$	Cost of $x_\imath$
$\mathcal{E}_{x_\imath}$	Expenses of $x_\imath$
$x_\imath$	A variable in $\mathcal{F}$
$p_{x_\imath}$	Success probability of $x_\imath$
$\textsc{Pr}[x_\imath]$	Success probability of $x_\imath$
σ	Attack suite
$\mathcal{F}\|_{x_\imath=v}$	Boolean formula derived by condition $x_\imath = v$ from $\mathcal{F}$
λ	Adversarial budget, line of a strategy
β	Branch of a strategy

continued on the next page ...

Symbol	Description
$\mathcal{S}$	Adversarial strategy
$\mathcal{P}_{\mathcal{G}}(\mathcal{S})$	Prize of a strategy $\mathcal{S}$ in satisfiability game $\mathcal{G}$
$\mathcal{U}(\mathcal{G}, \mathcal{S})$	Utility of strategy $\mathcal{S}$ in satisfiability game $\mathcal{G}$
$\mathcal{E}(\mathcal{G}, \mathcal{S})$	Expenses of strategy $\mathcal{S}$ in satisfiability game $\mathcal{G}$
$\mathcal{W}_{\mathcal{S}}$	The set of winning branches of the strategy $\mathcal{S}$
$\mathcal{L}_{\mathcal{S}}$	The set of non-winning branches of the strategy $\mathcal{S}$
$\mathcal{U}(\mathcal{G})$	Utility of a satisfiability game $\mathcal{G}$

Contents

Introduction

We live in the world in which the society is highly dependent on advanced diverse IT infrastructures which people use for performing daily activities and improving quality of life, private sector enterprises use it to provide services and operate, while governments rely on it to provide public services and ensure the welfare of the citizens. Such big and complex infrastructures are not vulnerability-free. Increasing numbers of IT security incidents all over the world have drawn attention to risk analysis methods capable of deciding whether the considered organization or infrastructure is sufficiently protected against relevant threats. The security controls are often costly and each security investment must be reasonable and properly justified. The security professionals have to justify the need for a security investment to their management and to explain them what are the benefits and what will an organization get for the money invested into security [11, 35].

There are no reliable and effective methods to assess whether the considered enterprise or infrastructure is secure or not – the existing computational methods are too complex to be a realistic candidate for practical use, while some of the existing methodologies place unnatural restrictions on the adversary and thus making the analysis results unreliable as they are capable of producing false-positive results.

Research Objectives and Hypotheses

The objectives of the research are:

- to improve the existing quantitative risk analysis models
- to create new computational methods which would not produce false-positive results – e.g. when the result of the computational method shows that the model is secure w.r.t. the definition of security, while in reality it is not
- to create robust computational methods capable of analyzing attack scenarios of practical size in reasonable time
- to create tools supporting the developed analysis methods

The research hypotheses are:

- there exist efficient computational methods to reliably calculate the upper bounds of adversarial utility

- if to eliminate adversarial limitations in the existing models, the computational methods become easier and more robust
- there exist efficient attack tree propagation rules that calculate reliable upper bounds and do not produce false-positive results
- genetic algorithms provide reasonably good approximation of the result, which is good enough to be used for the practical cases

Methodology

In our research, we use a combination of the analytic approach based on the principles of design science , and the experimental approach based on the principles of applied science . We cannot measure security in the physical world, but we can build a model of the world, provide a definition of security in this model, and create computational methods which verify if the model is secure w.r.t. the definition of security in the model. It is not possible to verify if the result of such an assessment corresponds to the real state of the analyzed organization in the physical world, but the model as well as the computational methods used in the model are falsifiable and thus it is still possible to determine the cases when either the model or the computational methods are incorrect. Design science allows to study and prove the properties of objects. In this research, the design science principles are used to create the improved failure-free model as well as its modification which considers limited adversarial budgets.

There are cases when the appropriate analytic technique does not exist, as in the case of the genetic algorithms. The efficiency and precision of genetic algorithms depend on a variety of loosely connected control parameters and thus there is no feasible analytic way to come up with optimal valuations for the control parameters. In this case, a feasible approach is to rely on applied science principles and verify the hypothesis by conducting experiments and collecting empirical evidence to prove or disprove the hypothesis. It is sufficient to find at least one solution to show that the research hypothesis does not hold. It is much harder and sometimes impossible to prove that the research hypothesis holds for the general case (for every possible valid input) using applied science. When the set of possible inputs is very large and it is unfeasible to find corresponding empirical evidence to prove the hypothesis for every considered input, a set of experiments is conducted on a reasonably large subset of possible inputs, and in case the hypothesis holds for this subset, an assumption may be made that there is a reasonable chance that the same hypothesis will hold in the general case as well. In this research, the principles of applied science are used to determine the optimal control parameters of the genetic algorithms for the parallel and the improved failure-free models.

Research Results

The results of research and contributions of this dissertation are the following:

- A new security analysis model (the improved failure-free model) was created. It takes into account a much broader scope of attackers compared to the previous models and is therefore more reliable.
- An algorithm for finding optimal adversarial strategies in the new model was developed. It was proven that the problem of finding an optimal strategy in the new model is NP-complete.
- Computational methods which calculate upper bounds of adversarial utility as well as the exact utility were created.
- Limited adversarial budget model was created based on the improved failure-free model, which takes real-world limitations of adversarial power into account. Such models can be used if there is a reliable knowledge about the adversarial capabilities.
- Based on the same idea a new concept of attacker profiling was introduced. It allows to analyze various organizations with the same sets of attacker profiles, and an organization may be analyzed using various attacker profiles. This provides flexibility to operational security risk analysis.
- Genetic approximation algorithms were developed to approximate the exact adversarial utility from below. It allows to estimate the difference between the exact result and the upper bound.

Novelty

The theoretical novelty lies in the new model for operational security risk analysis and the computational methods which do not produce false-positive results, but reliable upper bounds. The second theoretical novelty is the attacker profiling as the technique of separation of the properties of the protected infrastructure from the properties of the threat agents, as well as the two approaches to applying attacker profiling in quantitative analysis of operational security risks.

The practical novelty lies in the efficient algorithms implemented in the two analysis tools – the ApproxTree+ and the AttackTreeAnalyzer. The tools are capable of analyzing attack trees of practical sizes (dozens of thousands of leaves) in reasonable time which enables the practical use of the relevant analysis techniques.

Outline of the Thesis

The structure of the dissertation is the following:

Chapter 1 provides background information about security modeling, building reliable models and computational methods, as well as challenges of operational security risk metrics.

Chapter 2 introduces the threat modeling techniques: the fault trees and the attack trees, and the risk assessment methods based on these modeling techniques.

Chapter 3 outlines the state of the art by describing the relevant quantitative risk assessment methods based on attack trees.

Chapter 4 introduces the improved failure-free model – an analysis technique and efficient computational methods which provide reliable upper bounds of adversarial utility. This chapter is based on the author's publications [I]. The relevant research questions are:

- Does the model and its computational methods become more simple and reliable if to eliminate limitations placed on the adversary?
- What are the adversarial limitations in the existing models?
- Do optimal strategies exist in the new model?
- What is the complexity of finding an optimal strategy?
- How to compute precise outcome in the new model?
- Are there any efficient techniques based on value propagation which could be used to obtain reliable upper bounds?
- Which values can be propagated?
- What are the limitations of the propagation methods?
- What could be done to propagate values in attack trees having common sub-trees?

Chapter 5 introduces the so-called limited improved failure-free model, which considers limited adversarial budgets – a natural assumption which reflects real-life limitations of the adversaries. This chapter is based on the author's publications [III]. The relevant research questions are:

- Does the model and its computational methods remain easy, efficient and reliable if to consider budget limitations?
- Does it make sense to consider budget limitations?
- What are the adversarial strategies in the case of a limited budget?
- Which strategies are optimal and how to find them?
- Do the budgeted computational methods produce false-positive results?
- What are the conditions and assumptions under which the budgeted

model produces more precise result compared to the improved failure-free model?
- Does the efficiency of the computational methods allow to use the budgeted model in practice?

Chapter 6 introduces attacker profiling – the concept of separation of the properties of the protected infrastructure from the properties of the threat agents. Such a separation adds flexibility to operational security risk analysis as a single set of attacker profiles can be used to assess risks of various organizations, and likewise the risks of an organization may be assessed using various types of attacker profiles. This chapter is based on the author's publications [II,IV]. The relevant research questions are:

- How to take into account the fact that the metrics of operational security risks is determined by a set of underlying components?
- What are the relevant operational security risk metrics we are interested in?
- Which factors form the threat and vulnerability landscapes?
- How do these factors influence the quantitative annotations on the attacks?
- What are the possible approaches to handle these relations?

Chapter 7 describes the genetic algorithm which computes the approximated estimation of the adversarial utility and the corresponding analysis tool named ApproxTree+. This chapter is based on the author's publications [II,III]. The relevant research questions are:

- Does the integration of attacker profiling into existing risk assessment methods bring any computational complexity along with it?
- How to integrate attacker profiling into ApproxTree?
- How to integrate budget limitations into ApproxTree?
- How would the resulting algorithm look like?
- Does such integration bring any performance overhead along with it?

Chapter 8 describes the genetic approximations for the improved failure-free model as well as the analysis tool named Attack Tree Analyzer. This chapter is based on the author's publications [V]. The relevant research questions are:

- Is it possible to bypass the limitation of being able to analyze only independent trees by providing a feasible approximation to the result in the improved failure-free model?
- Which factors need to be defined in order to determine a genetic algo-

rithm?

- Which size of the initial population is optimal?
- Is there any optimal value for the mutation rate?
- When to terminate the reproduction process?
- What does the choice of cross-over type affect?
- How good is the approximation compared to the exact outcome?
- Can this approach be used in practice?

Chapter 9 summarizes the results of this dissertation and presents plans for future research.

Chapter 1

Theoretical Background

Information security has gained importance over the past decades. Information systems are used for mission critical tasks such as controlling and managing industrial processes, handling sensitive information such as i-votes, personal data, personal health records, payment transaction data, business sensitive information or state secrets. The amount of sensitive information that is stored digitally and transmitted across digital communication channels grows every year and likewise does the number of possibilities to attack systems which handle it. Technology evolves rapidly and eventually gets adopted into systems handling sensitive information. New technology brings along new vulnerabilities, as well as outdated technology, which is no longer supported. Environmental as well as human-made threats may take advantage of the existing vulnerabilities and result in direct or indirect damage to the affected parties and even affect human lives. The threat landscape has become even more dynamic and diverse (due to globalization), attacks have become more sophisticated and nowadays information security is a requirement, not a desirable feature.

Information security aims at protecting assets and preventing or at least reducing possible damage by deploying security controls and defensive measures. Security is not a state which can be achieved, but a process, where the situational awareness plays the key role. The threat environment changes rapidly and information security must be flexible to react to these changes, as the set of defensive measures which kept the organization and its assets at the required security level yesterday may fail to be same efficient today. Thus, the protected organization, as well as surrounding threat landscape needs instant monitoring and automated near-real-time analysis.

To the best of the author's knowledge, there are no scientifically justified and widely accepted metrics of strength against attacks, but such metrics exists in many other engineering areas – for instance, in civil engineering. Designing a new building or construction engineers need to make sure that it will not break. It is possible to calculate the precise stress values which a construction will be experiencing during exploitation taking numerous conditions into account by solving equations containing thousands of variables, but this approach cannot be called trivial. In order to verify that the construction will not break engineers calculate the upper bound of stress at which the construction definitely will not break, and they are not interested in complex calculations of the exact value of stress at which the construction breaks. It is desirable to have a simple and reliable method to calculate if the analyzed organization is secure, similar to the one used in civil engineering.

1.1 How to Measure Security?

How can we measure security? We have no evidence that security in the physical world is measurable at all – we have no measuring devices or sensors which could measure security of a given organization in a straightforward way. However we can build a model of the world, provide a definition of security in the model, and create corresponding computational methods which would verify that the model is secure w.r.t. the definition of security in the model. We will not be able to verify that the result of such an assessment corresponds to the real state of the analyzed organization, but the model as well as the computational methods are falsifiable. Suppose that in the physical world we have observed that the target organization was successfully attacked, but the result of analysis says that the organization is secure. If the attacks were not foreseen by the model – the model was falsified. If the attacks were foreseen by the model, but the result of analysis, obtained in the model, differs from the one observed in the physical world – the computational methods are invalid. The model as well as the computational methods are falsifiable when a security incident happens in the real life and its outcomes are observable. Given no observable outcomes from the physical world the best we can do is to assume that the model is correct, unless it will be falsified.

What should the model of the world contain? Obviously, we do not need to model the entire world, but only the related factors which affect the security of the analyzed organizations. The goal of securing organizations is to minimize and control possible damage caused by the relevant threats. Thus, there has to be a way to model environmental and human-made threats of the physical world in our model of the world. Since threats are events, the model should use event algebra to manipulate threats . Thus we have a model of the world which models physical world threats as events which happen with certain probabilities. Some of these events result in damage. As human-made threats are attack related and depend on attacker goals and motivations, our model of the world should describe attackers, their behavior and decision-making logic. We can make the organization more secure by deploying various operational security controls. There has to be a way to compare security controls, execute a cost-benefit analysis , and for this we need to be able to measure the efficiency of security controls.

1.2 Definition of Security

What would be a proper definition of security within the model? A very abstract definition of security might look like this: *The secure organization has such properties that the relevant threats cannot happen*. Intuitively such a definition corresponds to what we wish to achieve, but this is not achievable in the physical world, as environmental threats occur independently of our will – thus it turns out that we have chosen an incorrect definition of security.

We cannot stop eruptions of volcanoes or prevent earthquakes and floods. If there is no way to avoid nature threats and we cannot prevent them from happening, does it still make sense to secure organizations? Does it make sense to invest into installing burglar-resistant doors and locks and invest into installing the security surveillance system in the premises to fight the burglary threat, considered that there is a threat of an earthquake the damage from which exceeds the damage from a typical burglary thousands of times? Do we really reduce the overall potential damage fighting with "smaller threats" while there are global threats out there? Environmental and human-made threats are independent of each other and therefore the event of a burglary and the event of an earthquake are independent events. Their corresponding "contributions" into the total damage get summed up. Investing into installing burglar-resistant doors and surveillance system in the premises we are fighting specifically the burglary threat. As the burglary threat is independent of the environmental threats, fighting the burglary threat only, we still reduce the overall damage.

If to recall that the objective of information security is to minimize damage, we can treat *damage* as a measurable parameter and come up with the following definition of security: *The "secure state" is the state in which the sum of expected damage and expenses is minimized*. Such a state is not a secure state w.r.t. the first definition of security we have provided earlier, but this is the best state which is achievable in the current moment in time with resources available at this moment. Thus this state can be called *secure* as this is the best we can do to protect the organization at this moment in time. Maintaining security within organizations is a process the objective of which is to do everything we can to keep the protected organization secure at any given moment in time.

The total loss which organization may suffer from attacks is formed by the two components – the risk and the security investment – as shown by the following two equations:

$$\begin{aligned} Risk &= Probability\,of\,occurrence \times Damage \\ Loss &= Risk + Security\,Investment \end{aligned}$$

The risk is the expected damage that an organization may suffer in case a threat agent successfully exploits a vulnerability and materializes. For each modeled threat we need to know the two factors: the risk and the required security investment to mitigate the threat. Let us imagine a two-dimensional space shown in

Fig. 1.1, where every point in this space corresponds to a particular state of the organization annotated with expected loss. The elliptic lines represent points having the same value of loss. There is a point marked in black in the rightmost lower corner representing the current state of the organization and associated loss. Applying various security controls and deploying defensive measures it is possible to reduce risk. The security controls are shown as arrows which transition the related risks from one state to another. Applying security controls it is desirable to move in the direction of loss reduction (towards the imaginary "absolute security" area which corresponds to the risk with exposure 0€). The costs of security controls, being part of the total loss, pull the resulting state backwards in the direction of bigger losses. Thus, there is an area close to the imaginary point of "absolute security" which can never be reached even if we assume that the defender has got an unlimited amount of resources for defending, due to the fact that there are no security controls which would lead into this area. Thus, the point of "absolute security" is unreachable and it is very hard to get even close to that point. In some cases, after deploying a security control, the resulting loss may become even bigger which would result in moving in the opposite direction away from the "absolute security" point – e.g. if too expensive security measure is deployed, and the measure efficiency cannot justify its costs.

Which security measures would be efficient in this case and allow to reduce losses? We may denote the loss corresponding to the current state of the system as pD, where p is the probability of occurrence of the threat, and D is the associated damage. After deploying a security measure C with corresponding cost $\mathcal{E}$ the loss corresponding to the new state is expressed as $p'D' + \mathcal{E}$, where p' is the new probability of occurrence of the threat and D' is the new associated damage. It can be seen, that in order to move in the direction of loss reduction the inequality $\mathcal{E} \leqslant pD - p'D'$ must hold. Thus, the security control is effective if its cost does not exceed its control gap (reduced risk). It is impossible to estimate

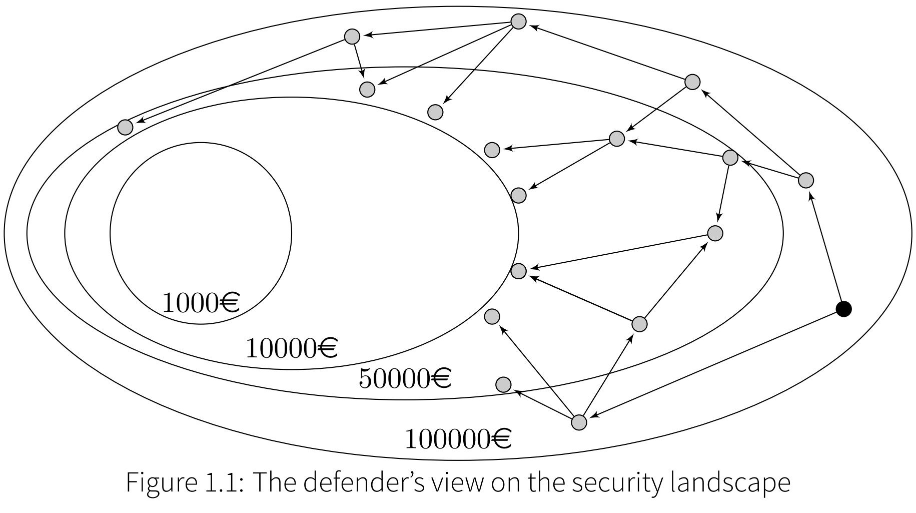


Figure 1.1: The defender's view on the security landscape

or measure all threats existing in the physical world. Obtaining the value of damage may be quite straightforward, but obtaining the value of probabilities may be quite tricky. The probability may not be measurable, but it is bounded – the value domain of probability is a bounded interval with lower limit 0 and upper limit 1. Even if the corresponding probabilities of certain threats are not measurable, we can still obtain values for the worst case and for the best case. Despite the fact that the probabilities of occurrence of some threats are non-measurable, it does not mean that the model or the definition of security, used in the model, are conceptually wrong – this is a problem of the threat phenomena itself. For instance, the probability of occurrence of an earthquake from the example above is not measurable or known, but nevertheless it is still possible to reduce the overall damage by fighting the threats we can fight against, for which we can measure relevant parameters – for instance, the burglary threat. Thus, given a model of the world where various events happen with certain probabilities and some of these events result in damage, even if some of the events and associated probabilities are not measurable, it is still possible to make informed and meaningful decisions based on the suggested model, as was shown by the example of the burglary and the earthquake threats described above.

Which threats do we need to include into our model of the world in order to perform meaningful analysis? The amount of threats existing in the physical world may be very big, and analysts using the model are not expected to discover and model all of them. A possible way out is to try to discover measurable threats and model the most dangerous threats among them, which result in maximal losses.

1.3 Attacker Model

Extensive statistical data on environmental threats has been collected within the previous years. This statistical data may be used as grounds for deriving the probability of occurrence of a particular environmental threat in a given region and its corresponding expected loss. We may treat environmental threats as random events which are not the result of someone's decisions and do not depend on someone's will. Differently from the environmental threats, the human-made threats, or attacks, occur as the result of the attacker decision-making process which resulted in the decision to attack. Targeted attacks are victim-specific and they do not fit into any statistical patterns. It turns out, that we cannot treat human-made threats as purely stochastic events. In order to obtain the probability of occurrence of targeted attacks we need to model attacker behavior. Such an attacker model needs to describe the attacker decision-making process in a simplified form, which means that we could substitute the real attacker with an automaton, the behavior of which is known, is predictable and deterministic – the attackers' choices remain the same as long as the conditions and limitations remain intact. Attackers may vary by their motivations, intentions, objectives, strategic decision-making logic, available resources, etc. As far as we are concerned with the attacker model, it makes sense to classify attackers by the logic behind their decision-making process into two major groups: *rational attackers* and *irrational attackers*. Similarly to the environmental threats and human-made threats, the attacks launched by the irrational attackers are independent of the attacks launched by the rational attackers. The corresponding damage induced by each of the types of attackers gets summed up to form the total damage, and for this reason we can study these two types of attackers separately. This research primarily focuses on the problem of minimizing the damage from rational attackers. Damage from attacks of irrational attackers, environmental threats, as well as the problem of obtaining input data for the model, remain out of the scope of this dissertation.

The rationality of a human thought is one of the key problems in psychology of reasoning. The question if human behavior can be modeled in a logical positivist manner using (only) standard rules of logic, statistics and probability theory, is still an open question and ground for disputes and debates. Rationality seems to be the widely used assumption about the behavior of individuals in micro-economic models and economic analysis of human decision making. The proponents of such models however do not claim that rationality assumption is an accurate description of the human behavior in the physical world, but such an approach allows to formulate clear and falsifiable hypotheses. According to the American economist and statistician Milton Friedman, the only way to judge the success of a hypothesis in such models is through empirical tests [10]. The rational choice theory studying the determinants of the individual choices has become increasingly popular in social sciences other than economics, such as sociology and political science in recent decades [34].

Rationality is the state of being reasonable based on the facts or reason, which implies that ones actions are logically consistent with ones preferences and reasons for acting. The rational choice theory states that on individual level rational agents, according to their personal preferences and constraints, choose among all available actions the one which they prefer the most. The theory bases on a set of assumptions on individual preferences that have to be satisfied:

- *Agents can make preferences over the set of possible alternatives or actions.* In order to make preferences over actions, they must be comparable. An agent cares only about outcomes resulting from each possible action, not the actions themselves – the actions are only means for obtaining a particular outcome. Thus, the outcome of an action can be used as the metrics to compare different actions to one another. For the outcomes to be comparable they should be quantified, and a partial order relation should be defined on them. Otherwise it would be impossible to compare the actions and choose the "best" among them.

- *Agent preferences are complete.* An agent can always decide which of two alternatives is preferable or that neither is preferred to the other.

- *Agent preferences are self-interested*. Following this assumption, an individual agent acts in a selfish manner. Such behavior may still be irrational w.r.t. the set of individual agents as a group.
- *The decision-making process of an agent is driven by a particular goal*. Without a clearly defined goal it would be hard to select relevant actions which would contribute the the agent's goals and drive his decision making process.
- *Agent preferences are transitive*. If action A is preferred over action B and similarly action B is preferred over action C, then A is preferred over C.
- *Agent preferences are consistent across time*. This means that the rational behavior is deterministic – the preference remains the same as long as the conditions and limitations remain intact.

The minimal required and sufficient conditions that have to be met in order for the behavior to be called *rational* are the following: the behavior must be driven by a certain goal and must be consistent across time in different choice situations [13]. Rational behavior is opposed by the stochastic (inconsistent across time), impulsive behavior driven by emotions, beliefs, ideas, which we call *irrational behavior*. For the rational choice to be possible, the agent's goal and a set of alternatives needs to be specified, without them it may not be possible to empirically test or falsify the rationality assumption. Rational attackers have two alternatives: to attack or not to attack. Following the rational choice theory, a rational attacker chooses attacking if this is beneficial for him, as shown in Fig. 1.2. Thus the probability of occurrence of an attack is equal to one if the action is beneficial, and is equal to zero if the considered action is not beneficial. In order to determine an optimal

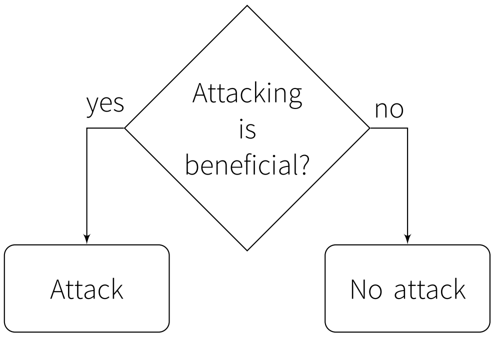


Figure 1.2: Behavioral model of a rational attacker

action, the rational choice theory requires that the formulation of the problem is quantified (e.g. outcomes, corresponding to particular actions, need to be comparable) and the key assumptions of the rational choice theory are satisfied. It can be seen that the behavioral model outlined in Fig. 1.2 satisfies the key assumptions of the rational choice theory.

Determining the conditions under which attacking the considered organization is beneficial, denoted by the choice node in Fig. 1.2, is the main focus of this dissertation. If we could find a way to easily determine if attacking is beneficial or not we would get a simple tool to assess whether the organization is secure against rational attackers or insecure. If attacking is not beneficial, we can assume that rational attackers will not be interested in attacking such a target, and on the contrary, if the results of analysis show that attacking is beneficial, such an organization is not secure, as it may be a fruitful target for rational attackers and attacks are likely to occur.

If we could quantify the outcomes of certain possible actions of an attacker and evaluate them in terms of costs and benefits, a rational attacker would be expected to take into account available information, probabilities of events, potential costs and benefits of the alternatives to determine preferences and choose an action corresponding to the maximal profit (the difference between revenue and cost) and to act consistently in choosing this self-determined “best” action. Such an agent, taking into account the trade-off between costs and benefits, prefers an action that maximizes personal advantage [10]. The cost-benefit analysis was applied to security modeling by Buldas *et al.* [4] where the authors discussed the criteria of rational choice of security measures considering economic feasibility of attacking. The model assumed an attacker who maximizes his utility (expected profit). In order to launch an attack some amount of resources, denoted by *cost*, need to be invested. Successful attacks bring an attacker some revenue denoted by *prize*. Thus the utility that an attacker gains from attacking is the difference between the expected revenue and the expenses required for attacking. The resulting utility is the metrics by which an attacker may judge whether attacking is beneficial or not – if the resulting outcome is positive, the rewards exceed expenses and thus attacking is beneficial. This model, being very general, can be applied to practically all types of rational attackers and can be used to analyze the feasibility of attacking by the considered attacker types.

It seems that the goal of the attacker, as well as underlying motivation, play the key role in determining if attacking is beneficial or not. Thus attacking an organization may be beneficial for fame-hunters, however attacking the very same organization may be not beneficial for profit-oriented attackers. Differently from the fame-oriented attackers, where we would need to measure fame on the same scale as cost (which is usually expressed in monetary units), it is relatively easy to measure the profit of rational profit-oriented attackers on the monetary scale as the value of assets which the attackers are targeting. Attacks launched by various types of attackers are independent of each other, thus attacks of rational profit-oriented attackers are independent of attacks of rational fame-hunters. Due to this various types of attackers may be studied separately from one another. In this thesis the focus is placed on minimizing damage from rational profit-oriented attackers performing targeted attacks against the target organizations. Dealing with the problem of minimizing the damage produced by rational profit-oriented attackers, we are still minimizing the total losses.

Rational profit-oriented attackers are driven by monetary profit. The objective they wish to achieve is known to them prior to attacking, as well as the value of an asset they are targeting, and the expected revenue which an attacker might get, for instance, by selling the stolen information or assets. Typically, in the case of a targeted attack the reconnaissance phase precedes the infiltration phase, during which an attacker collects knowledge about the target organization and the ways to attack it. When all the relevant knowledge has been collected, an attacker needs to decide, if it is worth attacking the considered organization, or he would be better off not trying attacking at all. With this respect the decision if it is worth attacking or not, is similar to project management, when one needs to take into account costs and potential benefits to decide if the project would be beneficial or not.

1.4 Threat Model

In order to assess whether attacking a target organization is beneficial or not, we need to look at the attacking process from the viewpoint of rational profit-oriented attackers who launch targeted attacks against the considered organization. Attacking is beneficial if the expenses (denoted by $\mathcal{E}$) do not exceed the expected profit (denoted by $\mathcal{P}$) as long as $\mathcal{E} < p\mathcal{P}$ (where p denotes the success probability of the attack).

Let us consider the threat of the loss of market share due to an intellectual property theft. This problem formulation is too abstract for an attacker to make any meaningful decision whether it is profitable to attack or not. The only parameter known to the attacker in this setting is profit. The expenses and probability of success of such an attack cannot be determined from this description of the threat.

In order to estimate the expenses and the success probability, a structured description of the attack is required. It is possible to refine the attacker's goal iteratively into sub-goals and so forth increasing the granularity of threat modeling, until we reach a level of the so-called elementary attack steps, the cost and success probabilities of which can be easily obtained in a relatively straightforward way. An attack can be structurally represented in the form of an *attack tree* with conjunctive and disjunctive nodes, and leaves corresponding to the elementary attacks. Thus, an attack tree corresponds to the monotone Boolean function, where the conjunctive and disjunctive refinements in the attack tree correspond to the conjunction and disjunction operators, and every leaf in the attack tree corresponds to a particular variable in the Boolean function.

An attacker may succeed in an attack in multiple ways by launching various combinations of elementary attack steps, which we call *attack suites*. An attack suite only determines the set of elementary attack steps considered by an attacker. When the attacker launches an attack from the suite and it succeeds, the corresponding variable in the monotone Boolean function is assigned with value *true*. Thus we can say that there are certain attack suites which satisfy the monotone Boolean function of an attack tree in case an attacker tries all the elementary attack steps from the attack suite and they succeed. When this happens, the Boolean function of an attack tree is satisfied, which means that the attacker has successfully executed an attack against the target organization and has materialized the primary threat described by an attack tree. The order in which the attacker launches the attacks steps, is determined by an *attack strategy* which expresses the logic behind the attacker's decision making process. Simply stated, a strategy is the rule which in every state of attack suggests the next elementary attack step to try, or to discontinue attacking.

Originally, attack trees were used for visualization purposes [33]. Most of the earlier studies focus on analyzing a single parameter at a time. These attack tree based analysis methods could be used to calculate the cost, probability of success, and similar parameters by using the technique called value propagation. The analyzed parameter was propagated in the bottom-up manner from the leaves up to the root node of an attack tree. The value of the parameter for each individual node was obtained from the corresponding values of its inputs. The result obtained for the root node was the result of analysis. A substantial step forward was taken by Buldas *et al.* [4] who introduced the idea of game-theoretic modeling of the adversarial decision making process based on several interconnected parameters like the cost, risks and penalties, associated with different atomic attacks.

1.5 Computational Methods

Attackers may choose various attack strategies, and for this reason computing the quantitative parameters of the game, like expenses and success probability, is a complex combinatorial task. Methods based on value propagation are easier in the computational complexity sense – they can operate in time linear in the size of an attack tree and therefore it would be fruitful to design computational methods which utilize efficient value propagation techniques. The computational methods should be reliable. If the security assessment is based on the assumption that the attacker will not attack if it is not profitable for him, every model, which tries to calculate the precise adversarial outcome using complex computation methods will always contain a margin for an error and thus the entire attacker model will fail. The reason for this is that in the case of the exact result, possible errors may propagate in both directions –- in the direction of lesser values, as well as in the direction of the greater values. Hence, the computational methods which calculate the exact result are capable of producing false-positive results if, for instance, the quantitative annotations of operational security risk are overlooked.

A reliable computational method does not need to calculate the exact result, but approaches the result "from above" thus calculating its upper bound. The upper bound is reliable as its possible error margin extends only in one direction – in the direction of the lesser values. In a reliable model the security assessment is based on the upper bound analysis and the corresponding computational methods calculate the upper bounds as the result.

1.6 Modeling Granularity

As will be further discussed in Chapter 3, the very first computational models of attack trees were rather simplistic and left a fair amount of actions, available to the attacker in the real life, behind the scenes. As time went on, subsequent models tried to increase the granularity of attacker behavior modeling and thus to bring the model more close to reality allowing the attackers to perform more and more actions that they can do in real life. However, every attempt to increase the modeling granularity came at the expense of a huge increase in computational complexity which rendered the models and their corresponding computational methods not usable for analyzing operational security risks in practice. However, the failure-free model by Buldas *et al.* [6] made the first step in the opposite direction, increasing the abilities of attackers even more allowing the attackers to perform actions that are impossible in the real life. Surprisingly, this has lowered the complexity of the computational methods and made the models easier to handle and analyze. It turns out that the more closely the model tries to reflect capabilities and limitations of real-life attackers, the more complex the computational methods become. However if to increase the capabilities of attackers and provide them with possibilities to execute actions that are impossible in real-life, the complexity drops. It makes sense to assume that if we increase capabilities of attackers even more, the computational methods will become even more easy, reliable and robust. This gives a chance to come up with a reliable model and corresponding computational methods, which would not be too complex and thus would be applicable to be used in practice. The idea behind this research is to take the failure-free model by Buldas *et al.* [6] as the baseline and eliminate limitations placed on the adversaries in this model assuming that the computational methods become more simple, and the corresponding analysis method will become more practice-oriented and applicable for analyzing real-life case studies.

1.7 Conclusions

The research aims at creating a simple and reliable method to analyze if the considered organization is sufficiently secure to withstand targeted attacks of rational attackers. We have discussed that security is not measurable in the physical world and thus the best we can do is to model the reality to the desirable precision, provide a definition of security within the model, and create corresponding computational methods which would verify if the model is secure w.r.t. the definition of security in the model. The model itself, as well as the computational methods, need to be falsifiable in order to be able to determine cases when either the model, or the computational methods, are invalid. The model contains the threat model, which uses event algebra to manipulate threats, and the attacker model, which represents the adversarial decision making logic, which forms the attacker strategies considered in the analysis. The threat model contains structured descriptions of the relevant human-made threats in the form of attack trees, annotated with quantitative parameters of operational security risks, such as the cost of an attack, probability of success, as well as contain the global annotation – prize, required for the attacker model to decide whether it makes sense to attack or not. The model assumes rational attackers, driven by the potential monetary profit, who attack iff it is profitable for them and decide not to attack otherwise. Therefore the definition of security used in the model states that the considered organization is secure if it has such properties that render the target unattractive for rational profit-oriented attackers – for instance, when attack expenses exceed potential benefits and therefore attacking such a target is not profitable. The computational methods used in the model must be easy enough to be able to come up with the result in reasonable time so that the analysis technique could be used in practice. The failure-free model by Buldas *et al.* [6] is taken as an absolute baseline, and the limitations, placed on attackers in this model, are eliminated. Reliable computational methods, which produce upper bounds in result and do not produce false-positive outcomes are built on top of it. The outcome of this research is a simple reliable upper-bound oriented approach to security engineering.

CHAPTER 2

THREAT MODELING

Scientific research can be conducted in a variety of ways. If the object of study is an observable, controllable, and measurable physical object it is possible to study it by conducting experiments and collecting empirical evidence about the properties and behavior of the object. The object must be observable, because it is impossible to experiment with something that we are unable to see or to sense. The object must be controllable in order to study it in different states and conditions, and it must be measurable, as in order to control something we need to be able to measure it.

However, there are cases when the object of study is inaccessible – e.g. we cannot access the core of the Earth in order to conduct experiments on how it influences the magnetic field of the Earth. Sometimes the object is not measurable – e.g. we cannot measure the distribution of rock density under the surface of the Earth. Sometimes the object is not under our control – e.g. we can observe and measure weather, but we can not control it. There are cases when the study object is indeed observable, controllable, and measurable, but experimenting with it is too costly – e.g. one would not build dozens of satellites and deliberately break them to test the survivability and fault-tolerance of the costly scientific measurement equipment.

In the cases when conducting experiments is unfeasible or economically impractical, scientists build models – simplified copies of the objects of study – and try to understand the phenomenon by studying the properties of the models, conducting experiments with the models reflecting some real-life situations. Every model is just a reflection of some physical object, and contains only some features of the original object which scientists consider to be relevant for the study. If some rule or property has been proven to be valid in the model, it does not mean that the same rule or property will hold for the real object. For this reason models are then verified in practice. When the experiments with the real prototype are unfeasible, the best what scientists can do is to assume that the model's results are correct, unless either the model, or the computational methods are falsified.

Security is no exception. The problem of security is that it is not directly measurable – we have no sensors or tools to measure security of an organization in a straightforward way. Indeed some experiments are conducted, like quantitative penetration testing [2], but they allow to measure the difficulty of attacking and can prove insecurity of the considered organization. We can conduct dozens of penetration tests, but the fact that none of them could reveal any viable attack vectors does not mean that they do not exist and that the organization is secure. Most probably the penetration testing results could not reveal any feasible attack vectors due to the fact that penetration testers did not consider all the assets of the organization, or the skill level of the penetration testers was insufficient to reveal real vulnerabilities. It is much harder to prove that an organization is secure. The complexity arises from the attack-defense asymmetry – in order to show that the organization is insecure, it is sufficient to show just one successful attack against it, but in order to show that the organization is secure it is required to consider all potential attacks, and show that viable attack vectors do not exist.

In one of the earliest publications on security modeling [31, 32] the authors outline the merits of using software development patterns in software engineering and argue that a similar approach should be followed in the security engineering. The authors outline several possibilities how the security patterns can be represented: security policies, Common Criteria, and attack trees. More recently, Opdahl *et al.* [17] have compared the usability of the misuse cases and attack trees by conducting two separate experiments. The authors argue that attack trees turned out to be more effective for threat identification when the participants tried to identify threats without the help of use-case diagrams which would help to identify misuse cases.

Security modeling came into practice not so long ago. Various studies and experiments show that the treat modeling technique known as threat trees or attack trees is a promising modeling technique which is useful not only for threat identification and visualization, but for the threat analysis as well. This chapter describes the attack trees and attack tree based analysis in greater detail.

2.1 Attack Trees

Hierarchical methods for security assessment have been used for several decades already. Called *fault trees* and applied to analyze general security-critical systems in early 1980-s [39], they were adjusted for information systems and called *threat logic trees* by Weiss in 1991 [40]. In the late 1990-s, the method was popularized by Schneier under the name *attack trees* [33].

An attack tree is a structured hierarchical description of a primary threat. It is an outcome of iterated refinement procedure during which analysts think about all possible ways how the considered primary threat can materialize, and express this knowledge in the form of an attack tree. An attack tree is a tree structure where nodes may represent two types of refinements – the conjunctive and the disjunctive refinement, and the leaves represent atomic attacks which are not refined any further. Figure 2.1 show an example of an attack tree. An attack tree is a graphical representation of a monotone Boolean function, where the conjunctive and disjunctive nodes in the tree correspond to the conjunctive and disjunctive operators in the Boolean formula, and the leaves in the attack tree correspond to the variables in the Boolean formula.

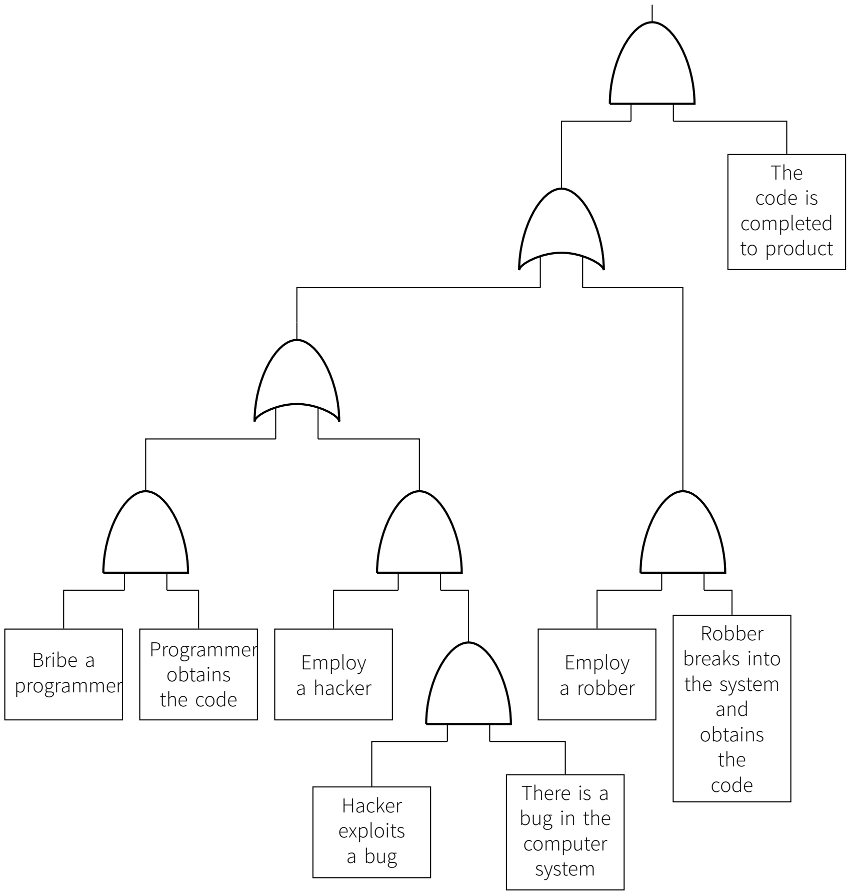

The code is completed to product
Bribe a programmer
Programmer obtains the code
Employ a hacker
Employ a robber
Robber breaks into the system and obtains the code
Hacker exploits a bug
There is a bug in the computer system

Figure 2.1: A sample attack tree for a software development company from [4]

2.2 Attack Tree Analysis

One of the simplest application of attack trees is purely descriptional. Attack trees may be the outcome of the threat identification phase in risk assessment. Such a structured hierarchical description of the threat may be shared with a security team allowing the analysts to make informed decisions about the security of the analyzed organization. Such an approach is limited only to qualitative assessment of security. Based on such an assessment, it is difficult to talk about optimal level of security or return of security investments (ROSI). In order to do this we need to quantify claims made during the analysis. Apart from purely descriptional purposes, attack trees can be used to analyze some security attributes of an organization, such as attack likelihood and costs of executing an attack. Already the first descriptions of attack trees introduced computational aspects [40, 33].

Most of the earlier studies focus on the analysis of a single parameter only. The analysis is executed in two steps. First, every leaf node in an attack tree is annotated with an estimation of the concerned quantitative attribute such as cost, or success likelihood. In the second stage, an iterated bottom-up value propagation technique is executed, which annotates intermediate nodes with values derived from the values of its children. The quantitative annotation on the root node is the result of such an analysis. The rules in accordance with which the annotation on the node is computed from the annotations of its children is determined by the nature of the analyzed attribute and is thus case-specific.

An attack tree is just a hierarchical representation of the attack paths which lead to primary threat materialization. The conjunctive node means that all the steps in the node have to be tried and succeed in order to succeed in the node. It contains no information whatsoever about the sequencing in which an attacker can launch attack steps, or if he is allowed to repeat attacks after they fail. Another approach to attack tree analysis is not to rely on the attack tree representation, but treat it merely as a description of the possible attack paths in the form of a monotone Boolean function. The sets of variables, which being assigned the value true, satisfy the Boolean function - attack suites - are treated as a sets of attack steps, which an attacker may launch in any order, and repeat failed attacks an arbitrary number of times. The ordering, in which an attacker tries to execute attack steps from the suite, as well as the rules when and how an attacker can repeat failed attacks again, are determined by an attack strategy. An attack strategy is a rule which in every state tells which attack step to try next, or to discontinue attacking. Therefore, every attack suite corresponds to an entire set of possible strategies. The analysis considers the set of optimal attacker strategies. The result of analysis the outcome that the attacker can achieve by executing attack steps in the order suggested by an optimal strategy.

2.3 Attack Tree Foundations

Attack tree is not a unique representation of possible attack suites. This is because the same Boolean function can be represented by many different Boolean formulae.

Mauw and Oostdijk [24] provided an unambiguous semantics for attack trees which based on a mapping to attack suites and does not depend on the representation of an attack tree, but only on its Boolean function. The authors suggested to disregard the fact that the structure of an attack tree carries information about the interpretation and grouping of attacks, and suggested to treat it as a collection of possible attacks which the authors call an attack suite. The authors acknowledge the possibility of an attacker to re-run attack steps an arbitrary number of times and thus they define an attack as a multi-set of attack steps. Rather than considering edges from a node in an attack tree to its children, the authors consider connections from a node to a multi-set of nodes. Such a connection is called a bundle, and a node may contain several bundles. All the nodes in a bundle must be executed in order to execute an attack. Execution of any bundle of a node is sufficient to execute an attack corresponding to this particular node.

The authors have defined a class of allowed semantics-preserving transformations of attack trees using the following reduction rules.

- If a bundle contains a node with only one sub-bundle, this node can be deleted and its sub-bundle can be lifted one level to become part of the bundle of the considered node
- If a bundle contains a node with two or more sub-bundles, the bundle can be replaced with the two copies of it, where the first copy contains the first sub-bundle, and the second copy contains the second sub-bundle.

These rules guarantee that the analysis of the two equivalent attack trees, even if they have different structure, will result in the same outcome. The authors argue that the structural information lost during interpretation of an attack tree as an attack suite is a residual of the modeling strategy. Attack suites therefore form an appropriate level of abstraction.

Chapter 3

Quantitative Risk Analysis

This chapter chronologically outlines quantitative attack tree analysis techniques, related to this research. All the models outlined below are somewhat similar to one another in terms of assumptions regarding the attacker behavior. All the models follow the *rational attacker's paradigm* suggested by Buldas *et al.* [4]. The paradigm assumes profit-oriented rational attackers who:

- attack only if it is profitable for them
- choose the most profitable ways of attacking (e.g. those with the highest outcome)

Thus, the decision about the security of the enterprise is undertaken based on the value of the outcome. If this value is positive, the enterprise is not secure and is a fruitful target for rational profit-oriented attackers, as profitable attack vectors exist which result in positive outcome for the attacker. On the contrary if the outcome is negative the considered enterprise is considered to be secure enough as the expenses of attacking such an enterprise exceed the potential profit and following the rational attacker paradigm, a rational attacker will decide not to attack such an enterprise.

3.1 Multi-parameter Attack Tree Analysis

The multi-parameter attack tree analysis technique [4] is notable for applying elementary game theory and rational economic reasoning to quantitative attack tree analysis. It is a risk analysis method for studying the security of institutions against rational profit-oriented attacks. The method allows to estimate the cost and the probability of success of attacks and by means of elementary game theory decide if the considered institution is a realistic target for attacking. This approach was a substantial step forward compared to the existing attack tree analysis, which assumed that quantitative annotations on the attack tree are independent from one another. The idea of multi-parameter analysis is that it is possible to analyze a set of dependent parameters – a set of dependent quantitative annotations is assigned to every leaf in the attack tree, then the propagation algorithm begins computing the same parameters for all the internal nodes as well, until the root node has been reached. The resulting vector obtained for the root node is used to compute the outcome value. The decision about the security of the system is undertaken based on the value of the outcome.

The decision if it is beneficial to attack or not is based on the following considerations. In order to launch an attack the attacker needs to invest some resources (bribe employees, buy a botnet, buy some equipment, etc.) denoted as *Costs*. After this the attacker launches an attack which may succeed with some probability p and in this case the attacker gets profit denotes as *Gains*. If the attack was successful, the attacker may get caught with probability q and in this case the attacker has to pay penalty denoted as *Penalty*. If the attack was not successful, the attacker may get caught with probability q^- and has to pay penalty denoted as Penalty$^-$. The attacker's decision making process based on this set of assumptions called the *rational attacker's paradigm* is modeled as a single player game plaid by the attacker. Fig. 3.1 shows the attack model in the form of an event tree.

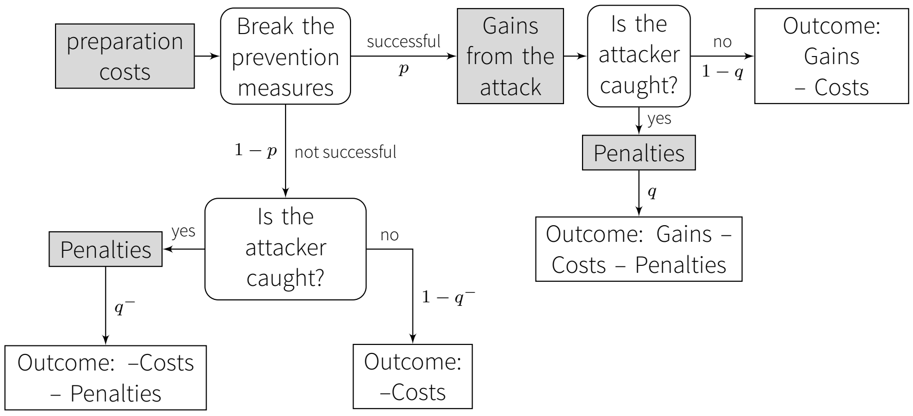


Figure 3.1: Event tree diagram from the attacker's point of view [4]

In the diagram rounded corner boxes represent events (probabilistic conditions), the dashed boxes correspond to gains and losses of the attacker, the arrows represent the state transitions during attack. Rectangular boxes represent possible outcomes of the attacker. There are four possible outcomes in the model shown in Table 3.1. The analysis starts by determining the primary threats (threats which directly result in damage to the affected party) with the subsequent constructing of attack trees for each of the identified primary threats. For the leaf nodes (atomic attacks) in the attack tree experts based on assumptions about real environment have to evaluate a tuple of four quantitative parameters (Costs$, p, \pi, \pi^-$), where $\pi = q \cdot$ Penalties and $\pi^- = q^- \cdot$ Penalties$^-$. Having this done, the computational procedure uses the bottom-up approach, in the case of which the corresponding quantitative annotations on some given intermediate node are computed from

Table 3.1: Outcomes in the model

Attack successful?	Attacker caught?	Outcome
yes	no	Gains - Costs
yes	yes	Gains - Costs - Penalties
no	no	-Costs
no	yes	-Costs -Penalties

the corresponding quantitative annotations of its child nodes. Additionally the Outcome value is computed by applying 3.1.

$$\text{Outcome} = -\text{Costs} + p \cdot (\text{Gains} - \pi) - (1-p) \cdot \pi^{-} \quad . \tag{3.1}$$

The quantitative annotations of nodes are determined based on the corresponding values of the child nodes in the following way:

- For an OR node with child nodes annotated with $(\text{Costs}_i, p_i, \pi_i, \pi_i^{-})(i = 1,2)$ the parameter $(\text{Costs}, p, \pi, \pi^{-})$ of the parent node is computed as:

$$(\text{Costs}, p, \pi, \pi^{-}) = \begin{cases} (\text{Costs}_1, p_1, \pi_1, \pi_1^{-}), \text{ if Outcome}_1 > \text{Outcome}_2 \\ (\text{Costs}_2, p_2, \pi_2, \pi_2^{-}), \text{ if Outcome}_1 \leqslant \text{Outcome}_2 \end{cases}$$

- For an AND node with child nodes annotated with $(\text{Costs}_i, p_i, \pi_i, \pi_i^{-})(i = 1,2)$ the parameter $(\text{Costs}, p, \pi, \pi^{-})$ of the parent node is computed as:

$$\text{Costs} = \text{Costs}_1 + \text{Costs}_2, \qquad p = p_1 \cdot p_2, \qquad \pi = \pi_1 + \pi_2,$$
$$\pi^{-} = \frac{p_1(1-p_2)(\pi_1 + \pi_2^{-}) + (1-p_1)p_2(\pi_1^{-} + \pi_2)}{1 - p_1 p_2} + \frac{(1-p_1)(1-p_2)(\pi_1^{-} + \pi_2^{-})}{1 - p_1 p_2} \quad .$$

The outcome value at the root node is considered to be the final outcome of the attack and the whole tree is considered to be beneficial for a rational attacker if the outcome is positive.

3.1.1 SHORTCOMINGS

The model has three major shortcomings.

Assumption about independency of attack steps. The model works only with independent attack trees assuming that the attack steps in the leaves of the attack tree are independent, however in real life attack trees it may not be the case. For example, consider the following attack trees:

Attack step B in the attack tree shown on the left in Fig. 3.2 contains a fan-in and resides under conjunctive refinement. Thus, the corresponding expenses of attack

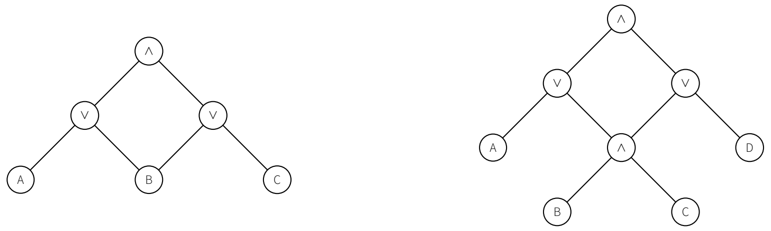


Figure 3.2: Examples of dependent attack trees

step B may invest twice into the total expenses propagated to the root node which may result in imprecise result. The same happens in the case of the attack tree shown on the right in Fig. 3.2 where the entire sub-tree $B \wedge C$ is dependent.

Incompatibility with Mauw-Oostdijk framework. Mauw *et al.* state in Foundations of Attack Trees [24] that attack tree semantics of the tree should remain unchanged when the underlying Boolean formula is transformed into an equivalent one. Semantics provided by Buldas *et al.* is not consistent in this sense.

Limited attacker capabilities. Despite the novelty of applying the elementary game theory to attack tree analysis, the model is rather simplistic. The notions of attacker behavior, preferences and strategy are missing in the model. The propagation rules allow an attacker only to decide if it is beneficial to attack or not. This model does not provide enough flexibility for the attacker to decide in which order to launch attacks or when to withdraw. In real life, attackers may play adaptively and plan their next move based on the result from the previous moves and collected knowledge, they may re-run failed attack steps again an arbitrary number of times when it is profitable.

3.2 Parallel Model

The Parallel model [14] allows to determine the attacker's exact expected outcome based on a multi-parameter attack tree. The model treats the attack tree as a monotone Boolean function $\mathcal{F} = \{x_1, x_2, \ldots, x_n\}$ in which attack tree leaves correspond to the variables in the Boolean function. The attack is represented as a set of elementary attack steps $\sigma \subseteq \mathcal{X}$ called the *attack suite* which is a set of variables in the Boolean function corresponding to the set of attack steps considered and launched by the attacker. The model considers only satisfying attack suites – such sets of variables, which when set to value 1, satisfy the Boolean function. For each attack suite the corresponding outcome value is computed using the following formulae:

$$\text{Outcome}_\sigma = p_\sigma \cdot \text{Gains} - \sum_{x_i \in \sigma} \text{Expenses}_\imath \ ,$$
$$p_\sigma = \sum_{\substack{\rho \subseteq \sigma \\ \mathcal{F}(\sigma := true) = true}} \prod_{x_i \in \rho} p_\imath \prod_{x_\jmath \in \sigma \setminus \rho} (1 - p_\jmath) \ .$$

Attack suites σ are allowed to be redundant and contain subsets $\rho \in \sigma$ sufficient for satisfying the Boolean function $\mathcal{F}$. Redundant attack suites may seem to be counter-intuitive and not a rational choice. Despite the fact that bigger investments are required in order to launch and run a redundant attack suite, they have bigger probability of success compared to non-redundant attack suites due to the existence of subsets ρ which contribute to the probability of success of an entire attack suite. Running such attack suites may still be profitable and for this reason authors consider redundant attack suites in their analysis.
During analysis the attacker's expected outcome is maximized over the entire set of satisfying attack suites:

$$\text{Outcome} = \max \Big\{ \text{Outcome}_\sigma : \sigma \subseteq \mathcal{X}, \mathcal{F}(\sigma := true) = true \Big\} \ .$$

The most profitable attack suite is the result of analysis and the attacker is expected to launch all the attack steps from the attack suite simultaneously in parallel.

The authors prove that their model is consistent with Mauw and Oostdijk foundations [24] as the model works with the Boolean function of the attack tree and does not depend on its representation. This allows the authors to achieve consistency with the foundations using propositional Boolean semantics and fix the second shortcoming of the Buldas *et al.* model [4] – *Incompatibility with Mauw-Oostdijk framework*. Besides, the authors have shown that the parallel model provides at least the same outcome as Buldas *et al.* model. As the model works with Boolean functions directly and does not depend on its actual form it is capable of analyzing dependent attack trees such as shown in Fig. 3.2. Thus the model fixes the first shortcoming of the Buldas *et al.* model [4] – the *assumption about independency of attack steps*.

3.2.1 Shortcomings

The Parallel model has fixed the two of the three shortcomings of Buldas *et al.* model [4], but the third shortcoming, the limited attacker capabilities, still remained unsolved. Besides, the model introduces one extra shortcoming – complexity.

Limited attacker capabilities. The attacker strategy in which the attacker executes all the attacks in the attack suite simultaneously and has only one trial (attacks cannot be repeated) seems not to be very close to reality for the same reasons that are outlined in the shortcomings of the Buldas *et al.* model [4].

Computational complexity. The model maximizes outcome over the set of all satisfying attack suites which results in a rather huge amount of potential satisfying attack suites in the case of a big monotone Boolean function. Another source of complexity is redundant attack suites. To compute the success probability of such a suite one has to take into account all possible subsets of the attack suite which is a complex task. The authors admit that the complexity of the suggested approach is exponential in the worst case even with all the optimizations they could come up with [14].

3.3 Serial Model

The model [15] is an extension of the parallel model [14] which introduced temporal order to the attacker's decision making process. The behavioral model of the attacker is semi-adaptive, the linear order of attack steps is fixed in advance. The attacker executes the attack steps one by one, possibly skipping some elementary attacks and stopping attacking only if the value of the Boolean function $\mathcal{F}$ has been completely determined by the successes and failures of the previously executed attack steps. The full strategy of the attacker in the model is the following:

1. Create an attack tree with the set of leaf nodes $\mathcal{X} = \{x_1, x_2, \ldots, x_n\}$.
2. Select an attack suite $\sigma \subseteq \mathcal{X}$.
3. Select a permutation α of σ.
4. Based on the attack suite σ and the selected permutation α compute the expected outcome.
5. Maximize the expected outcome over all possible choices for σ and α.

The attacker launches attack steps from the selected attack suite σ in order determined by permutation α. The expected outcome of the attack based on permutation α is determined as

$$\text{Outcome}_\alpha = p_\alpha \cdot \text{Gains} - \sum_{x_\imath \in \mathcal{X}} p_{\alpha,\imath} \cdot \text{Expenses}_\imath \ ,$$

where p_α is the success probability of the primary threat and $p_{\alpha,\imath}$ denotes the probability that the node x_i has any effect on the success or failure of the root node. An elementary attack does not have any effect on the success or failure of the root node iff there is a node on the path from that particular leaf to the root that has already been determined. Following these considerations, the next elementary attack should be tried iff all the nodes on the path are undefined.

The probability $p_{\alpha,\imath}$ for the path from the root node $\mathcal{R}$ to the leaf $x_\imath$: $(Y_0 = \mathcal{R}, Y_1, \dots, Y_m = x_\imath)$ is computed as follows:

$$\begin{aligned}
p_{\alpha,\imath} &= \text{Pr}[Y_0 = u \& Y_1 = u \& \dots \& Y_m = u] \\
&= \text{Pr}[Y_0 = u | Y_1 = u, \dots, Y_m = u] \\
&\quad \cdot \text{Pr}[Y_1 = u | Y_2 = u, \dots, Y_m = u] \\
&\quad \dots \cdot \text{Pr}[Y_{m-1} = u | Y_m = u] \cdot \text{Pr}[Y_m = u] \\
&= \text{Pr}[Y_0 = u | Y_1 = u] \cdot \text{Pr}[Y_1 = u | Y_2 = u] \\
&\quad \cdot \text{Pr}[Y_{m-1} = u | Y_m = u] \cdot \text{Pr}[Y_m = u] \ .
\end{aligned}$$

The attack process may be described as a single player game played by the attacker as shown in Fig. 3.3. Every state in the game is described by the Boolean function $\mathcal{F}$. This function in the initial state in the beginning of the game corresponds to the Boolean function of the attack tree. The attacker picks attack steps (corresponding to the variables in the Boolean function of the attack tree) one at a time. An attack step may succeed with certain probability or fail. An attacker is not allowed to re-run the failed attack steps, but rather launches attack steps in a pre-defined order. After the move has been chosen the attacker executes the attack step and the game transits to another state, which depends on whether the move was successful or not. The game ends when one of the following conditions is satisfied:

- $\mathcal{F}$ is satisfied, which corresponds to the condition when the attacker wins the game
- $\mathcal{F}$ is not satisfied and there are no legitimate moves left, which corresponds to the loss in the game
- when the attacker decides to quit playing the game

The authors prove that the serial model complies with the foundations of Mauw and Oostijk [24] and achieve better outcomes compared to the Parallel model [14] by taking temporal dependencies in the attacker decision making into account and as such bring the model closer to reality.

3.3.1 Shortcomings

Computational complexity. Unfortunately this reality comes at a price of immense increase in computational complexity. Having introduced temporal order to the attack suite the model maximizes the outcome over all possible combinations of satisfying attack suites and orderings of attack steps in the considered suite. The number of satisfying attack suites is large and the number of all possible orderings of attack steps in all of the attack suites is even larger. The necessity to consider all such combinations and calculate corresponding outcome value for every of them makes it doubtful that this model may ever be useful for practical use.

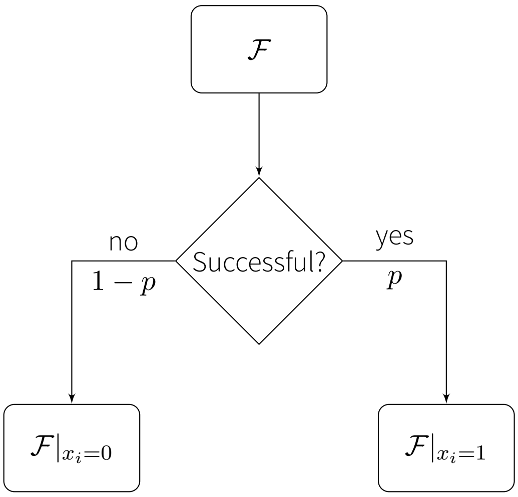


Figure 3.3: The attacker game in the serial model

Limited attacker capabilities. Despite providing attacker with greater flexibility and freedom to decide compared to the Parallel model [14], the serial model still suffers from limiting the attacker's decision making process way too much. In real life an attacker may try some elementary attack first and then based on its success or failure select the next attack to try or even decide to stop attacking altogether due to certain success or failure of the primary threat. However in the serial model the attacker is limited to executing attacks in a fixed pre-defined order which does not contribute to the modeling nor outcome precision.

3.4 ApproxTree

Even with all possible optimizations to the exact computation algorithm in [14] the authors claim that the model is practically limited to analyzing attack trees having at most 30 leaves and due to the exponential nature of the problem it is hard to expect substantial progress in the exact computations. The authors suggested a genetic algorithm to approximate the result [16].

The algorithm starts with generation of the initial population of individuals (satisfying attack suites) of size p:

1. Consider the root node.
2. If the considered node is a leaf, include it in individual σ and stop.
3. If the considered node is an AND-node, consider all its descendants recursively and go back to Step 2.
4. If the considered node is an OR-node, flip a fair coin for every descendant and decide if it should be included in σ. If no descendants have been in-

cluded, continue flipping coin until at least one descendant is chosen. Proceed with Step 2.

This procedure results in an attack suite σ which is guaranteed to be a satisfying attack suite, which is called *live individual* in the genetic algorithm.

Having obtained the first generation of individuals (which are not required to be distinct in the population), the reproduction process starts:

1. Every individual $\sigma_i \in p$ is crossed with every other individual producing $\binom{p}{2}$ individuals
2. Each individual is mutated with probability 0.1
3. The new individuals are joined with the exiting population
4. Every candidate is checked for liveness. Only the live ones are left
5. p fittest individuals are selected for the next generation

The overall complexity of the genetic algorithm running over g generations was estimated to be $\mathcal{O}(gp^2(\log p + n))$. The genetic algorithm turned out to be very scalable and allowed to analyze attack trees containing more than 100 leaves.

3.4.1 Shortcomings

It is a common knowledge that genetic algorithms are not precise and the result of the algorithm may not be the precise solution. The genetic algorithm approximates the precise result “from below” providing outcomes which are less or equal to the exact outcome. Such a method is capable of producing false-positive results providing the analysts with false feeling of being secure. The existing computational methods which approximate the outcome lack computational procedures to assess the margin for error.

3.5 Fully Adaptive Model

In order to avoid unnatural restrictions of the above mentioned models [4, 14, 15] Buldas *et al.* have created a fully adaptive model [6] which considers fully adaptive adversaries which are allowed to launch attack steps in arbitrary order based on the results of the previous trials. In order to make the computational method both reliable and efficient the authors do not try to compute the exact outcome, but only the upper bounds. The model was created in order to analyze the most complex case possible, when the attacker has no prior plan of attacking and picks attack steps adaptively based on the collected information, the current state of the game and the outcomes of the previous moves. The authors were aiming to come up with smart heuristics which would not produce false-positive results and at the same time make the computation methods easier.

Compared to analyzing with the model which computes the precise result, the model which computes upper bounds is more reliable. In the case of a negative precise outcome it is not guaranteed that there are no profitable ways to attack the system. However if the upper bound is negative any possible precise outcome is guaranteed to be negative as well and thus it is safe to conclude that there are no beneficial ways to attack an enterprise.

The exact utility in the fully-adaptive model can be computed using the following recurrent relation

$$\mathcal{U}(\mathcal{A}) = \max_{\jmath} \left\{ 0, -\mathcal{C}_\jmath - (1 - p_\jmath - q_\jmath) \cdot \Pi_\jmath + p_\jmath \mathcal{U}(\mathcal{A}_{x_\jmath=1}) + q_\jmath \mathcal{U}(\mathcal{A}_{x_\jmath=0}) \right\} \quad ,$$

with seed values $\mathcal{U}(\mathbf{1}) = \mathcal{P}$ and $\mathcal{U}(\mathbf{0}) = 0$, where $\mathbf{1}$ and $\mathbf{0}$ correspond to attack games with Boolean functions $\mathcal{F} \equiv 1$ and $\mathcal{F} \equiv 0$ respectively.

The authors have shown that in the case of atomic AND game the optimal strategy will suggest to pick the move with the smallest cost-nonsuccess ratio $\frac{\mathcal{C}_\imath}{1-p_\imath}$ and in the case of atomic OR game the optimal choice would be to pick an attack step with maximal utility-nonfailure ratio $\frac{\mathcal{U}(x_\imath)}{1-q_\imath}$.
The attacker actions may be represented as a single-player game, each state in which is represented by an attack tree with the corresponding Boolean function $\mathcal{F}$. The player picks a move $x_\imath$ (attack step) and pays a certain amount of expenses $\mathcal{C}_\imath$. The move succeeds with probability p and fails with probability q. The game ends when $\mathcal{F} \equiv 1$ meaning that the game is won by the player or $\mathcal{F} \equiv 0$ meaning the loss of the game. With probability $1-p-q$ the player is caught. In this case he has to pay a certain amount of penalty $\Pi_\imath$ and the game is over. Fig. 3.4 represents the game in a graphical form.

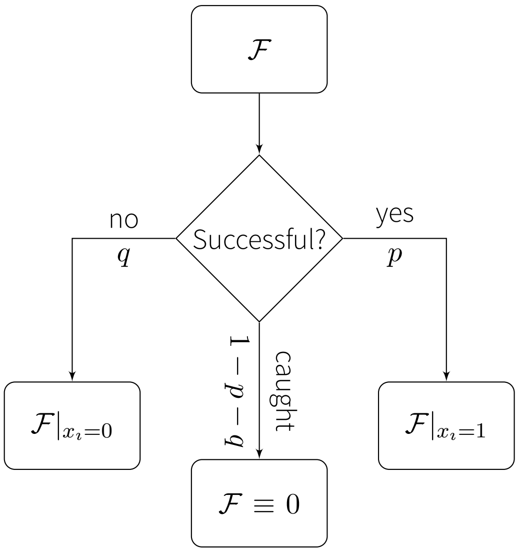


Figure 3.4: The attacker game in the fully adaptive model

3.5.1 Cost Reduction

In order to tackle the complexity of the precise computational method, the authors have elaborated two decomposition rules which propagate attacker utility from the leaves of the attack tree towards the root node:

$$\mathcal{U}(\mathcal{A}_1 \wedge \ldots \wedge \mathcal{A}_n) \leqslant \min\{\mathcal{U}(\mathcal{A}_1, \ldots, \mathcal{A}_n)\} \ ,$$
$$\mathcal{U}(\mathcal{A}_1 \vee \ldots \vee \mathcal{A}_n) \leqslant \mathcal{U}(\mathcal{A}_1) + \ldots + \mathcal{U}(\mathcal{A}_n) \ .$$

The propagation rules are applied to every node in the attack tree starting from the leaves up towards the root node of the tree. These rules are robust, but imprecise. Besides, the OR rule is valid only for the independent sub-trees – sub-trees containing no common atomic attacks the Boolean formulae of which have no common variables. For the general case the authors have suggested a technique called *cost reduction* the idea of which is to modify the sub-tree and reduce the costs $\mathcal{C}'_\jmath = \frac{\mathcal{C}_\jmath}{k_\jmath}$ as well as the penalties $\Pi'_\jmath = \frac{\Pi_\jmath}{k_\jmath}$ of common attack steps, where $k_\jmath$ is the number of times the common attack step $\mathcal{X}_\jmath$ is repeated in the Boolean formula of the sub-trees. Such a reduction of costs does not break the upper bound ideology – if the system is proven to be secure even if some of the attack steps are artificially made easier than they really are, this implies that the attacks against the real enterprise are unfeasible as well.

3.5.2 Shortcomings

Limited attacker capabilities. Providing an attacker with possibility to play fully adaptive strategies was a major achievement, but this freedom and flexibility came along with unnatural restrictions placed on the adversary. In the considered model the player was allowed to launch an attack only once (no move repetitions) which may not be the case in the real world. Additionally, the condition when the attacker is caught implied a game-over state. In the real world when the attacker gets detected he still may continue attacking and get profit. Thus, the fully adaptive model does not guarantee that there are no profitable ways to attack an enterprise.

3.6 Infinite Repetition Model

In order to tackle the shortcomings of the fully-adaptive model, the authors suggested another model called the infinite repetition model [6]. In this model the attacker is allowed to re-run failed attacks an arbitrary number of times. Due to the rules of the game after the move has failed the state of the game remains the same. This property is particularly interesting as in this case the optimal strategy which suggested to pick the move $x_\imath$ in the game state $\mathcal{S}_k$ will suggest to pick the same move in the state $\mathcal{S}_{k+1}$ provided that the move $x_\imath$ has been tried and failed in the state $\mathcal{S}_k$. This results in possibility for the player to launch failed attacks infinitely until they eventually succeed. Thus, the cost of infinitely repeated attack step becomes $\frac{\mathcal{C}}{p}$ and the probability of success is one. The model where the probabilities of attack steps are equal to one is called the failure-free model. The authors have shown that any repeatable atomic attack x with quantitative annotations $(p, q, \mathcal{C}, \Pi)$ is equivalent to a non-repeatable attack x' with parameters $(\frac{p}{1-q}, 0, \frac{\mathcal{C}}{1-q}, \Pi)$ and thus the infinite repetition model is equivalent to the failure-free model (see Fig. 3.5).

$$\begin{tikzcd}
1 & 1 & 1 & & & 1 \\
x_\imath \arrow[u, "p"] \arrow[r, "q"] \arrow[d, "1-p-q"'] & x_\imath \arrow[u, "p"] \arrow[r, "q"] \arrow[d, "1-p-q"'] & x_\imath \arrow[u, "p"] \arrow[r, "q"] \arrow[d, "1-p-q"'] & \cdots & \equiv & x'_\imath \arrow[u, "\frac{p}{1-q}"] \arrow[d, "1-\frac{p}{1-q}"'] \\
\perp & \perp & \perp & & & \perp
\end{tikzcd}$$

Figure 3.5: A repeatable atomic attack $\mathcal{X}$ with quantitative annotations $(p, q, \mathcal{C}, \Pi)$ is equivalent to a non-repeatable attack x' with parameters $(\frac{p}{1-q}, 0, \frac{\mathcal{C}}{1-q}, \Pi)$

An adversarial strategy in the infinite repetition model may be expressed as a BDD as shown in Fig. 3.6. The set of attacks in the figure $\langle x_1, x_2, \ldots, x_k \rangle$ is sorted by their cost-nonsuccess ratio $\frac{\mathcal{C}_i}{1-p_i}$ in ascending order and thus if the attack x_1 has the lowest cost-nonsuccess ratio it will be launched first.

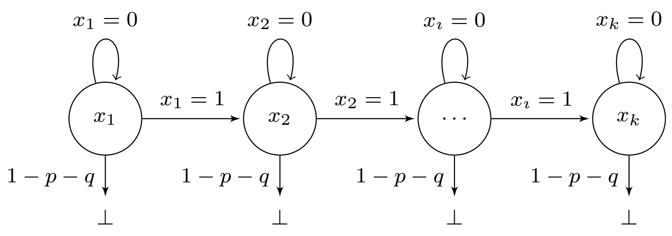


Figure 3.6: Adversarial strategy in the infinite repetition model

The exact utility in the failure-free model is computed using the formula

$$\mathcal{U}_\infty(\mathcal{A}) = \max\left\{ 0, \frac{-\mathcal{C}}{1-q} + \frac{p}{1-q} \cdot \mathcal{U}_\infty(\mathcal{A}_{x=1}), \mathcal{U}_\infty(\mathcal{A}_{x=0}) \right\} \quad ,$$

with initial conditions $\mathcal{U}_\infty(\mathbf{1}) = \mathcal{P}$ and $\mathcal{U}_\infty(\mathbf{0}) = 0$. The authors have shown that such non-adaptive strategies always exist in the set of optimal strategies for the game. The next best move to try suggested by an optimal strategy is an attack step with minimal cost-nonsuccess ratio $\frac{\mathcal{C}'}{1-p'} = \frac{\mathcal{C}}{1-q-p}$, where $p' = \frac{p}{1-q}$ and $c' = \frac{c}{1-q}$ are the transformed parameters.

3.6.1 Shortcomings

The shortcomings of the failure-free model are fairly similar to the shortcomings of the fully adaptive model. Namely, the condition when the attacker is caught implied a game-over state. In the real world when the attacker is detected he still may continue attacking and get profit.

3.7 Conclusions

The very first models were rather simplistic and left a fair amount of actions, available to the attacker in the real life, behind the scenes. As time went on, subsequent models tried to increase the granularity of the behavioral modeling of attackers and thus to bring the model more close to reality allowing attackers to perform more and more actions that they can do in real life.

Thus the multi-parameter model by Buldas *et al.* [4] as well as the parallel model by Jürgenson *et al.* [14] allowed an attacker only to decide if it is beneficial to attack or not. These models did not provide the attacker with flexibility to decide in which order to launch the attacks or when to withdraw. They did not consider that attackers may play adaptively and plan their next moves based on the experience obtained from the previous moves. Attackers were allowed to launch a single attack just once and if it failed it could not be launched again. The serial model by Jürgenson *et al.* [15] has made a step forward by introducing a temporal order to the attacker's decision making process and thus making the behavioral model of the attacker semi-adaptive. The linear order in which the attacker launched the attack steps was fixed in advance and the decision whether to continue attacking or to withdraw was completely determined by the successes and failures of the previously executed attack steps. The fully-adaptive model by Buldas *et al.* [6] was a substantial step forward in bringing the attacker model closer to reality. This was the first model which considered fully adaptive adversarial behavior (among all the models at that moment capable of doing multi-parameter quantitative risk analysis based on attack trees and relying on the propositional Boolean semantics to obtain the result). The fully-adaptive model did not consider that the attacker may re-run the failed attacks and this was fixed in the subsequent model called the infinite repetition model [6]. Up to this point every attempt to increase the granularity of modeling of adversarial behavior brought excessive complexity along with it and every subsequent model was more complex than the previous one. The infinite repetition and the failure-free models [6] have granted attacker with capabilities which are non-typical and can hardly be observed in real life – namely the possibility to re-run attacks infinitely. In real life, an attacker does not have such possibility as his resources are limited. In order to launch an attack the attacker has to invest money and in order to be able to run attacks infinitely the attacker's budget must be infinite. Another real-life constraint is time. Attacks cannot last infinitely long. Every subsequent launch of the attack increases the probability that the attack will get detected by the security monitoring solutions. Malicious activities will be discovered and the attacker may get caught. With each attempt to re-run failed attacks again the risk of detection increases and at some point a real-life attacker will decide to withdraw, but this is not the case in the infinite repetition and the failure-free models.

The models which appeared earlier than the the fully adaptive model tried to increase the modeling granularity and bring the attacker model closer to reality. This came at the expense of a huge increase in computational complexity. The fully adaptive model modeled reality with the greatest granularity compared to its predecessors and was hardly applicable for practical use due to complexity issues. It was designed mainly for scientific purposes – to derive heuristic methods which would lower the complexity of the computational methods and provide nevertheless an exact result. The failure-free model made the first step in the opposite direction, increasing the abilities of attackers even more and thus going further away from reality into the fictional reality where the attackers have supernatural abilities, infinite budgets and infinite time available for attacking without being detected and traced back and can perform actions impossible in the real life. Surprisingly, this has lowered the complexity of computational methods and made the models easier to handle and analyze. It turns out that the more close the model reflects real life capabilities and limitations the more complex the computational methods become. However if to go even further beyond reality into the fictional reality and to increase the capabilities of attackers even more, the complexity drops.

The drop in computational complexity is a fruitful goal as the easier the computational methods are, the more trustworthy are the results obtained by such methods. However in order to show that such a transition into the fictional reality is justified, we need to show that the analysis remains reliable and does not provide false-positive results. If the security assessment is based on the assumption that the attacker will not attack if it is not profitable for him, every model which tries to calculate the precise attacker outcome will always contain a margin for an error and thus the entire adversarial decision model fails. The reason for this is that in the case of the exact result possible errors may propagate in both directions – in the direction of lesser values as well as in the direction of the greater values. Such computational methods are capable of producing false-positive results if, for instance, the quantitative annotations are overlooked. Reliable computational methods do not need to calculate the precise result, but approach the result “from above” thus calculating its upper bound. The upper bound is reliable as its possible error margin extends only in one direction – in the direction of the lesser values and such an approach will not produce false-positive results. In a reliable model the security assessment is based on the upper bound analysis and the corresponding computational methods calculate the upper bounds as the result. It can be shown that the transition into the fictional reality implies that the corresponding computational methods produce the upper bounds as the result.

In such models attackers are granted with supernatural abilities and are capable of executing actions impossible in the real life. In the fictional reality the attackers are made stronger than they are in the real life and thus attacking the same target is easier for them. It is unlikely that the result obtained for the overpowered attackers will be less than the same result obtained for the real world attackers and thus if the attacking is not profitable for the overpowered attackers, it is unlikely that it will be profitable for the real world attackers. Thus the conclusions of the security assessment based on the upper bounds is reliable and free from false-positive results.

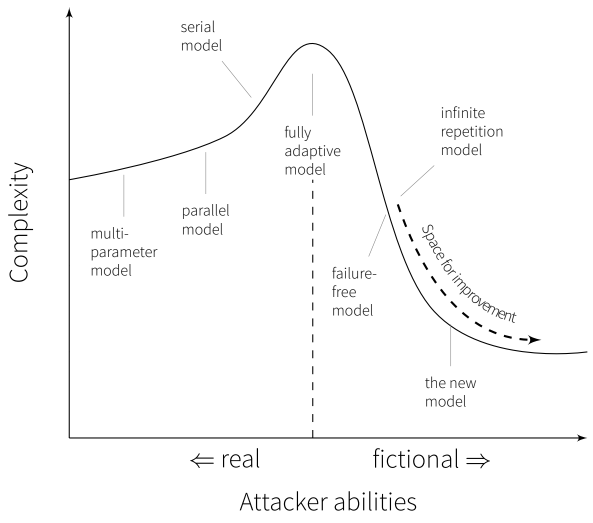


Figure 3.7: Computational complexity and modeling realism

It seems that by providing attackers with supernatural abilities we are still moving in the right direction – the computational methods remain reliable, trustworthy and do not introduce false positive results, at the same time the complexity decreases. The complexity dynamics of the computational methods in related models is illustrated in Fig. 3.7. It makes sense to make an assumption that by going even further into the fictional reality and providing even more abilities to the attackers it is reasonable to expect the continuous decrease in the complexity of the computational methods which become even more easy, reliable and robust. This gives a chance to come up with a reliable model and computational methods, that are not too complex and thus are applicable in practice.

How far does it make sense to go into the fictional reality? We suppose that the further we go the easier the computational methods become. At the same time, attackers become so overpowered that protecting the enterprise against such overpowered attackers will result in over-secured infrastructures. Over-securing is by no means a desirable goal as in the contemporary world security measures are costly and investments into security need to be justified. Security professionals have to explain to the management what will an organization benefit from invest-

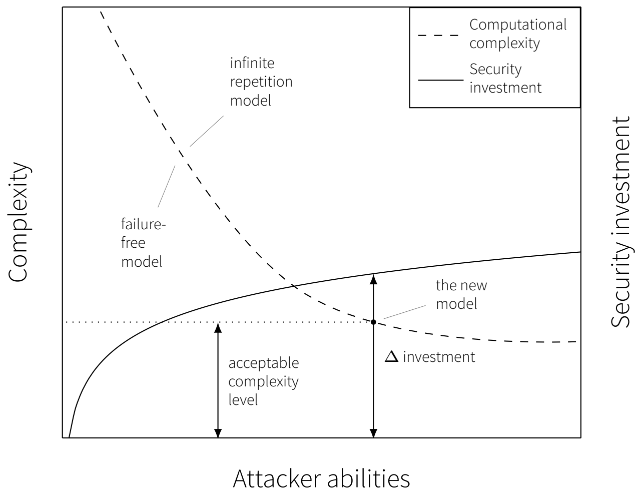


Figure 3.8: Computational complexity and over-securing investment

ing into such a costly protection. Obviously, meaningless over-securing has no justification. Too precise computational methods are complex and not applicable in practice, as they are unable to come up with the result in meaningful time. The more abilities we provide to the attackers in models the easier the computational methods become, but at the expense of over-securing the protected organizations. The necessity for over-securing is justified when it is balanced by a drop in the computational complexity, as shown in Fig. 3.8. We need to move into the fictional reality (and bring over-securing side-effect along with us) just far enough for the complexity of the computational methods to drop to the acceptable level when the analysis method can compute the result in meaningful time and thus be applicable in practice, as shown in Fig. 3.7.

The author was inspired by the failure-free model [4] by Buldas *et al.* The idea behind the presented research is to take the failure-free model as the baseline and move even further away from reality by eliminating limitations placed on the adversary in the failure-free model in assumption that the computational methods become more simple and the resulting analysis method will become more practice-oriented and applicable for real-life case studies.

Chapter 4

Improved Failure-Free Model

Even the fully-adaptive model [6] of Buldas *et al.* does not follow the upper bound ideology. The attacks are mostly associated with criminal behavior and hence in the attack tree models [4, 14, 15] elementary attacks are associated with penalties that the adversary has to pay if his actions are detected. The failure-free model introduces an additional restriction to the attacker model – the adversary has to discontinue attacking when detected and is not able to continue attacking after that. As this seems not to be true in all real-life cases, either the penalties in their model contain the potential future profits of the adversary (and hence are larger than they are in real life) or the model does not produce reliable upper bounds. Besides it seems that such a game-over assumption actually makes the computational methods more complex.

This chapter presents the improved failure-free model [5] that allows the adversary to repeat elementary attacks if they fail and to continue attacking when the launched attacks are detected. It turns out that such a model is somewhat easier to analyze compared to the failure-free model of Buldas *et al.* [6]. For example, in the case of a conjunctive decomposition $x_1 \wedge x_2 \wedge \ldots \wedge x_n$ the order in which the adversary launches attacks is irrelevant and due to this property the computational methods do not require to sort attacks by their cost-nonsuccess ratio $\frac{\mathcal{C}_i}{1-p_i}$ prior to analyzing, as was in the case of the original failure-free model. It is shown that in the suggested model non-adaptive strategies always exist in the set of optimal strategies which is good, as in the general case optimal strategies tend to be fully-adaptive. Such non-adaptive strategies are in the form of a directed single-branched BDD-s with self-loops and can be analyzed more efficiently compared to analyzing the fully-adaptive strategies. Additionally, it is shown that solving the attack game in the suggested model is equivalent to solving a Weighted Monotone Satisfiability (WMSAT) problem which we prove is NP-complete. This can be considered an achievement in reducing the complexity of the computational models, as the complexity of solving attack games in the general case tends to belong to PSPACE [25]. It is worth noticing that it is still unknown whether the similar problems in the existing models are NP-complete.

The new model contains several computational methods. The exact adversarial utility computation method is way too complex to be used in practice and is used for research purposes only as a reference to the exact result. The two propagation methods – the utility propagation and the expenses propagation – produce the precise upper bound estimation in the case of independent attack trees. In the case of dependent attack trees the propagation methods approximate the upper bound from above and thus bring the necessity for even greater excessive investments into security, but such investments are balanced by the linear complexity of the propagation methods.

4.1 Attacker Model

The attacker game in the improved failure-free model may be described as a single-player game played by the adversary. Every state in the game is represented by a monotone Boolean function $\mathcal{F}$ where each input variable $x_\imath$ in $\mathcal{F}$ is annotated with success probability $p_\imath$ and expenses $\mathcal{E}_\imath$ which is modeled as a random variable. Here we need to introduce a couple of terms necessary for understanding the explanation below.

Definition (Constant functions). *By* **1** *we mean a Boolean function that is identically true and by* **0** *we mean a Boolean function that is identically false.*

Thus when we say that some variable $x_\imath$ is assigned value 1 we mean that $x_\imath$ is set to true. Likewise, when we say that $x_\imath = 0$ we mean that $x_\imath$ is set to false.

Definition (Derived Boolean function). *If $\mathcal{F}(x_1, \ldots, x_m)$ is a Boolean function and $v \in \{0, 1\}$, then by the derived Boolean function $\mathcal{F}|_{x_\imath = v}$ we mean the function $\mathcal{F}(x_1, \ldots, x_{\imath-1}, v, x_{\imath+1}, \ldots, x_m)$ derived from $\mathcal{F}$ by the assignment $x_\imath := v$.*

For example if $\mathcal{F} = x_1 \vee x_2$ then $\mathcal{F}|_{x_1=1} = 1$. Similarly, if $\mathcal{G} = x_1 \wedge x_2$ then $\mathcal{G}|_{x_1=1} = x_2$.
In the initial state the game corresponds to the Boolean function $\mathcal{F}$ equivalent to the attack tree containing all possible moves of the adversary in the game. In this setting the adversarial goal is to satisfy $\mathcal{F}(x_1, \ldots, x_k)$ by picking variables $x_\imath$ one at a time and assigning $x_\imath = 1$. Actions that the adversary undertakes in every instance of the game can be described as follows:

1. The adversary executes a move $x_\imath$ and pays a certain amount of expenses $e_\imath \leftarrow \mathcal{E}_\imath$. This random choice is independent of other random choices from $\mathcal{E}_\imath$.
2. With a certain probability $p_\imath$, the move $x_\imath$ succeeds, and the function $\mathcal{F}$ representing the current game instance is transformed to its derived form $\mathcal{F}|_{x_\imath=1}$.
3. With probability $1 - p_\imath$ the game remains the same (the Boolean function

does not change: $\mathcal{F}|_{x_\imath=0} \equiv \mathcal{F}$) due to the ability of the adversary to re-run failed attacks again an infinite number of times.

The game goes on until one of the following conditions is satisfied:

- $\mathcal{F} \equiv 1$ – the adversary satisfies $\mathcal{F}$, wins the prize $\mathcal{P}$ and thus wins the game
- $\mathcal{F} \equiv 0$ – the adversary fails to satisfy $\mathcal{F}$ and thus loses the game
- when the adversary decides to discontinue playing the game

4.2 Strategies

This section will show that optimal strategies always exist in repeatable satisfiability games. It will also show that non-adaptive strategies in the form of a single-branched BDDs with self-loops always exist in the set of optimal strategies. Additionally, some of the properties of optimal strategies are studied in this section. Let us start with a few definitions required for understanding the explanations below.

4.2.1 Definitions

Definition (Line of a game). *By a line of a satisfiability game we mean a sequence of assignments $\lambda = \langle x_{\imath_1} = v_1, \ldots, x_{\imath_k} = v_k\rangle$ (where $v_\imath \in \{0,1\}$) that represent adversarial moves, and possibly some auxiliary information. We say that λ is a* ***winning line*** *if the Boolean formula $x_{\jmath_1} \wedge \ldots \wedge x_{\jmath_k} \Rightarrow \mathcal{F}(x_1, \ldots, x_n)$ is a tautology, where $\mathcal{F}$ is a Boolean function of the satisfiability game.*

Definition (Strategy). *By a strategy $\mathcal{S}$ for a game $\mathcal{G}$ we mean a rule that for any line λ of $\mathcal{G}$ either suggests the next move $x_{\imath_{k+1}}$ or decides to give up.*

Definition (Line of a strategy). *A line of a strategy $\mathcal{S}$ for a game $\mathcal{G}$ is the smallest set $\mathcal{L}$ of lines of $\mathcal{G}$ such that (1) $\langle\rangle \in \mathcal{L}$ and (2) if $\lambda \in \mathcal{L}$, and $\mathcal{S}$ suggests $x_\imath$ as the next move to try, then $\langle \lambda, x_\imath = 0\rangle \in \mathcal{L}$ and $\langle \lambda, x_\imath = 1\rangle \in \mathcal{L}$.*

Definition (Branch). *A branch β of a strategy $\mathcal{S}$ for a game $\mathcal{G}$ is a line λ of $\mathcal{S}$ for which $\mathcal{S}$ does not suggest the next move. By $\mathcal{B}_\mathcal{S}$ we denote the set of all branches of $\mathcal{S}$.*

For example, all winning lines of $\mathcal{S}$ are branches.

Definition (Expenses of a branch). *If $\beta = \langle x_{\imath_1} = v_1, \ldots, x_{\imath_k} = v_k\rangle$ is a branch of a strategy $\mathcal{S}$ for $\mathcal{G}$, then by expenses $\epsilon_\mathcal{G}(\mathcal{S}, \beta)$ of β we mean the sum $\overline{\mathcal{E}}_{\imath_1} + \ldots + \overline{\mathcal{E}}_{\imath_k}$ where by $\overline{\mathcal{E}}_{\imath_\jmath}$ we mean the mathematical expectation of $\mathcal{E}_{\imath_\jmath}$.*

Definition (Prize of a branch). *The prize $\mathcal{P}_\mathcal{G}(\mathcal{S}, \beta)$ of a branch β of a strategy $\mathcal{S}$ is $\mathcal{P}$ if β is a winning branch, and 0 otherwise.*

Definition (Utility of a strategy). *By the utility of a strategy $\mathcal{S}$ in a game $\mathcal{G}$ we mean the sum:*

$$\mathcal{U}\left(\mathcal{G},\mathcal{S}\right) = \sum_{\beta\in\mathcal{B}_{\mathcal{S}}} Pr\left(\beta\right)\cdot\left[\mathcal{P}_{\mathcal{G}}\left(\mathcal{S},\beta\right) - \epsilon_{\mathcal{G}}\left(\mathcal{S},\beta\right)\right] \quad .$$

For the empty strategy $\mathcal{U}\left(\mathcal{G},\emptyset\right) = 0$.

Definition (Prize of a strategy). *By the prize $\mathcal{P}\left(\mathcal{G},\mathcal{S}\right)$ of a strategy $\mathcal{S}$ we mean the sum $\sum\limits_{\beta\in\mathcal{B}_{\mathcal{S}}} Pr\left(\beta\right)\cdot\mathcal{P}_{\mathcal{G}}\left(\mathcal{S},\beta\right)$.*

Definition (Expenses of a strategy). *By the expenses $\mathcal{E}\left(\mathcal{G},\mathcal{S}\right)$ of a strategy $\mathcal{S}$ we mean the sum $\sum\limits_{\beta\in\mathcal{B}_{\mathcal{S}}} Pr\left(\beta\right)\cdot\epsilon_{\mathcal{G}}\left(\mathcal{S},\beta\right)$.*

It is easy to see that $\mathcal{U}\left(\mathcal{G},\mathcal{S}\right) = \mathcal{P}\left(\mathcal{G},\mathcal{S}\right) - \mathcal{E}\left(\mathcal{G},\mathcal{S}\right)$.

Theorem 4.2.1. *The utility of the satisfiability game is the difference between the prize and expenses of the game:*

$$\mathcal{U}(\mathcal{G},\mathcal{S}) = \mathcal{P}(\mathcal{G},\mathcal{S}) - \mathcal{E}(\mathcal{G},\mathcal{S}) \quad . \tag{4.1}$$

Proof. Let $\mathcal{W}_{\mathcal{S}}$ be the set of all winning branches and $\mathcal{L}_{\mathcal{S}}$ be the set of all non-winning branches of the strategy $\mathcal{S}$. Then:

$$\begin{aligned}\mathcal{U}(\mathcal{G},\mathcal{S}) &= \sum_{\beta\in\mathcal{W}_{\mathcal{S}}} \mathrm{Pr}(\beta)\left[\mathcal{P}_{\mathcal{G}}(\mathcal{S},\beta) - \epsilon_{\mathcal{G}}(\mathcal{S},\beta) - \sum_{\beta\in\mathcal{L}_{\mathcal{S}}} \mathrm{Pr}(\beta)\epsilon_{\mathcal{G}}(\mathcal{S},\beta)\right] \\ &= \sum_{\beta\in\mathcal{W}_{\mathcal{S}}} \mathrm{Pr}(\beta)\mathcal{P}_{\mathcal{G}}(\mathcal{S},\beta) - \sum_{\beta}\epsilon_{\mathcal{G}}(\mathcal{S},\beta) \\ &= \mathrm{Pr}(\mathcal{W}_{\mathcal{S}})\mathcal{P} - \mathcal{E}(\mathcal{G},\mathcal{S}) = \mathcal{P}(\mathcal{G},\mathcal{S}) - \mathcal{E}(\mathcal{G},\mathcal{S}) \quad .\end{aligned}$$

□

Definition (Expenses of a satisfiability game). *By the expenses $\mathcal{E}(\mathcal{G})$ of a satisfiability game $\mathcal{G}$ we mean the limit*

$$\mathcal{E}(\mathcal{G}) = \inf_{\mathcal{S}} \mathcal{E}(\mathcal{G},\mathcal{S}) \quad ,$$

where $\mathcal{S}$ varies over all possible winning strategies.

Theorem 4.2.2. *For any satisfiability game $\mathcal{G}$:*

$$\mathcal{U}(\mathcal{G}) = \max\left\{0, \mathcal{P} - \mathcal{E}(\mathcal{G})\right\} \quad .$$

Proof. Let $\mathcal{S} \neq \emptyset$ be an optimal strategy for $\mathcal{G}$. This means that $\mathcal{S}$ is a winning strategy, and hence $\mathcal{U}(\mathcal{G},\mathcal{S}) = \mathcal{P} - \mathcal{E}(\mathcal{G},\mathcal{S})$ holds. This brings us to the following feature:

$$\mathcal{U}(\mathcal{G}) = \sup_{\mathcal{S}} \mathcal{U}(\mathcal{G},\mathcal{S}) = \mathcal{P} - \inf_{\mathcal{S}} \mathcal{E}(\mathcal{G},\mathcal{S}) = \mathcal{P} - \mathcal{E}(\mathcal{G}) \ .$$

If $\mathcal{S} = \emptyset$ then obviously $\mathcal{U}(\mathcal{G}) = \mathcal{U}(\mathcal{G},\emptyset) = 0$. □

Theorem 4.2.2 justifies the expenses propagation method, which was introduced in Section 4.4.4. As $\mathcal{P}$ is a constant, the utility upper bound can be obtained by computing the lower bound of expenses and subtracting it from $\mathcal{P}$.

Definition (Utility of a satisfiability game). *The utility of a satisfiability game $\mathcal{G}$ is the limit $\mathcal{U}(\mathcal{G}) = \sup_{\mathcal{S}} \mathcal{U}(\mathcal{G},\mathcal{S})$ that exists due to the bound $\mathcal{U}(\mathcal{G},\mathcal{S}) \leqslant \mathcal{P}$.*

Definition (Optimal strategy). *By an optimal strategy for a game $\mathcal{G}$ we mean a strategy $\mathcal{S}$ for which $\mathcal{U}(\mathcal{G}) = \mathcal{U}(\mathcal{G},\mathcal{S})$.*

Prior to showing that in repeatable satisfiability games optimal strategies always exist, we need to introduce the two lemmas below.

Lemma 4.2.3. *For every repeatable satisfiability game $\mathcal{G}$ with $\mathcal{U}(\mathcal{G}) > 0$ there is $x_\imath$ such that*

$$\sup_{\mathcal{S}\in\mathbb{S}_{x_\imath}} \mathcal{U}(\mathcal{G},\mathcal{S}) = \mathcal{U}(\mathcal{G}) \ ,$$

where $\mathbb{S}_{x_\imath}$ is the set of all non-empty strategies with $x_\imath$ as the first move.

Proof. As every $\mathcal{S} \neq \emptyset$ has the first move $x_\imath$, we have:

$$\mathcal{U}(\mathcal{G}) = \sup_{\mathcal{S}} \mathcal{U}(\mathcal{G},\mathcal{S}) = \max_{\jmath} \sup_{\mathcal{S}\in\mathbb{S}_{x_\jmath}} \mathcal{U}(\mathcal{G},\mathcal{S}) \ ,$$

and hence there is $x_\imath$ such that $\mathcal{U}(\mathcal{G}) = \sup_{\mathcal{S}\in\mathbb{S}_{x_\imath}} \mathcal{U}(\mathcal{G},\mathcal{S})$. □

Lemma 4.2.4. *For every repeatable satisfiability game G and for every atomic variable $x_\imath$:*

$$\sup_{\mathcal{S}\in\mathbb{S}_{x_\imath}} \mathcal{U}(\mathcal{G},\mathcal{S}) = -\mathcal{E}_\imath + p_\imath \cdot \mathcal{U}(\mathcal{G}|_{x_\imath=1}) + (1-p_\imath)\cdot \mathcal{U}(\mathcal{G}) \ .$$

Proof. This is because the part $\mathcal{S}'$ of $\mathcal{S}$ for playing $\mathcal{G}|_{x_\imath=1}$ and the part $\mathcal{S}''$ of $\mathcal{S}$ for playing $\mathcal{G}$ after an unsuccessful trial of $x_\imath$ can be chosen independently. □

Now we are ready to show that for every repeatable satisfiability game optimal strategies always exist.

Theorem 4.2.5. *Repeatable satisfiability games have optimal strategies.*

Proof. If $\mathcal{U}(\mathcal{G}) = 0$, then $\mathcal{S} = \emptyset$ is optimal. For the case $\mathcal{U}(\mathcal{G}) > 0$ we use induction on the number m of atomic variables. If $m = 0$, there are no moves and $\emptyset$ is the only possible strategy – it is optimal by definition. In case $m > 0$ and supposing that every repeatable satisfiability game with $m - 1$ atomic variables has an optimal strategy, by *Lemma 4.2.3*, there is $x_\imath$ such that

$$\mathcal{U}(\mathcal{G}) = \sup_{\mathcal{S}\in\mathbb{S}_{x_\imath}} \mathcal{U}(\mathcal{G}, \mathcal{S}) \ .$$

Let $\mathcal{S}_0$ be the strategy that repeats $x_\imath$ until $x_\imath$ succeeds and then behaves like an optimal strategy for $\mathcal{G}|_{x_\imath=1}$ (a game with $m - 1$ atomic variables). Utility of $\mathcal{S}_0$ is

$$\mathcal{U}(\mathcal{G}, \mathcal{S}_0) = -\frac{\mathcal{E}_\imath}{p_\imath} + \mathcal{U}(\mathcal{G}|_{x_\imath=1}) \ .$$

On the other hand, by *Lemma 4.2.4* we have

$$\mathcal{U}(\mathcal{G}) = \sup_{\mathcal{S}\in\mathbb{S}_{x_\imath}} \mathcal{U}(\mathcal{G}, \mathcal{S}) = -\mathcal{E}_\imath + p_\imath \mathcal{U}(\mathcal{G}|_{x_\imath=1}) + (1 - p_\imath)\mathcal{U}(\mathcal{G}) \ ,$$

that implies

$$\mathcal{U}(\mathcal{G}) = -\frac{\mathcal{E}_\imath}{p_\imath} + \mathcal{U}(\mathcal{G}|_{x_\imath=1}) = \mathcal{U}(\mathcal{G}, \mathcal{S}_0) \ ,$$

and hence $\mathcal{S}_0$ is optimal. □

Corollary 4.2.6. *In every repeatable satisfiability game there exist optimal strategies in the form of directed single-branched* BDDs *with self-loops.*

Proof. Let $\mathcal{S}$ be an optimal strategy for $\mathcal{G}$ and $x_{\imath_1}$ be the first move suggested by $\mathcal{S}$. In case of a failure, $x_{\imath_1}$ remains the best move and hence $\mathcal{S}$ has a self-loop at $x_{\imath_1}$. In case of success, the Boolean function of the game reduces to $\mathcal{F}|_{x_{\imath_1}=1}$. Let $x_{\imath_2}$ be the next move suggested by $\mathcal{S}$. Similarly, we conclude that there is a self-loop at $x_{\imath_2}$ in case of a failure, and so on. This leads to the BDD in Fig. 4.1. □

$$\begin{tikzcd}
x_{\imath_1} \arrow[loop above, "x_{\imath_1} = 0"] \arrow[r, "x_{\imath_1} = 1"] & x_{\imath_2} \arrow[loop above, "x_{\imath_2} = 0"] \arrow[r, "x_{\imath_2} = 1"] & \cdots \arrow[loop above, "x_{\imath_i} = 0"] \arrow[r, "x_{\imath_{k-1}} = 1"] & x_{\imath_k} \arrow[loop above, "x_{\imath_k} = 0"]
\end{tikzcd}$$

Figure 4.1: A strategy in the form of a directed single-branched BDD with self-loops

Theorem 4.2.7. *If $\mathcal{S}$ is a strategy in the form of a self-looped* BDD *(Fig. 4.1), then*

$$\mathcal{E}(\mathcal{G}, \mathcal{S}) = \frac{\overline{\mathcal{E}}_{\imath_1}}{p_{\imath_1}} + \ldots + \frac{\overline{\mathcal{E}}_{\imath_k}}{p_{\imath_k}} \ .$$

Proof. The probability that a move succeeds at the n-th try is $p(1-p)^{n-1}$ and the average expenses are $\overline{\mathcal{E}}(1-p)^{n-1}$ and hence the total success probability is

$$p \cdot \sum_{n=1}^{\infty}(1-p)^{n-1} = 1 \ ,$$

and the average expenses are

$$\overline{\mathcal{E}} \cdot \sum_{n=1}^{\infty}(1-p)^{n-1} = \frac{\overline{\mathcal{E}}}{p} \ .$$

□

It is obvious that in the new model $\mathcal{P}(\mathcal{G},\mathcal{S}) \in \{0,\mathcal{P}\}$.

4.3 Complexity

We show that the problem of finding the optimal strategy in the improved failure-free model is equivalent to solving a *Weighted Monotone Satisfiability* (WMSAT) problem [5].

Definition (Weighted Monotone Satisfiability problem). *Given a threshold value $\mathcal{P}$ and a monotone Boolean function $\mathcal{F}(x_1,\ldots,x_m)$ with corresponding weights $w(x_\imath) = w_\imath$ decide whether there is a satisfying assignment $\mathcal{A}$ with a total weight $w(\mathcal{A}) < \mathcal{P}$.*

Theorem 4.3.1. *In the improved failure-free model the problem of deciding whether $\mathcal{U}(\mathcal{G}) > 0$ is equivalent to solving the weighted monotone satisfiability problem with the same Boolean function as in $\mathcal{G}$ and with weights of the input variables $x_\imath$ defined by $w_\imath = \frac{\overline{\mathcal{E}}_\imath}{p_\imath}$ with threshold value $\mathcal{P}$.*

Proof. If $\mathcal{S}$ is an optimal strategy in the improved failure-free model and is a single-branched BDD with self-loops with nodes $x_{\imath_1},\ldots,x_{\imath_k}$, then the assignment $\mathcal{A} = \langle x_{\imath_1} = \ldots = x_{\imath_k} = 1\rangle$ satisfies the Boolean function of the game and its total weight is $w(\mathcal{A}) = \frac{\overline{\mathcal{E}}_{\imath_1}}{p_{\imath_1}} + \ldots + \frac{\overline{\mathcal{E}}_{\imath_k}}{p_{\imath_k}}$. If $\mathcal{U}(\mathcal{G}) > 0$, then

$$0 < \mathcal{U}(\mathcal{G}) = \mathcal{U}(\mathcal{G},\mathcal{S}) = \mathcal{P} - \frac{\overline{\mathcal{E}}_{\imath_1}}{p_{\imath_1}} - \ldots - \frac{\overline{\mathcal{E}}_{\imath_k}}{p_{\imath_k}} = \mathcal{P} - w(\mathcal{A}) \ .$$

Thus, $w(\mathcal{A}) < \mathcal{P}$. If there is an assignment $\mathcal{A} = \langle x_{\imath_1} = \ldots = x_{\imath_k} = 1\rangle$ in the WMSAT model with a total weight $w(\mathcal{A}) = \frac{\overline{\mathcal{E}}_{\imath_1}}{p_{\imath_1}} + \ldots + \frac{\overline{\mathcal{E}}_{\imath_k}}{p_{\imath_k}} < \mathcal{P}$, then the strategy shown in Fig. 4.1 has the utility

$$\mathcal{U}(\mathcal{G},\mathcal{S}) = \mathcal{P} - \frac{\overline{\mathcal{E}}_{\imath_1}}{p_{\imath_1}} - \ldots - \frac{\overline{\mathcal{E}}_{\imath_k}}{p_{\imath_k}} > 0 \ , \tag{4.2}$$

and hence $\mathcal{U}(\mathcal{G}) \geqslant \mathcal{U}(\mathcal{G},\mathcal{S}) > 0$. □

Theorem 4.3.2. *The Weighted Monotone Satisfiability Problem is* NP*-complete.*

Proof. We will show that the vertex cover problem can be polynomially reduced to the WMSAT problem. Let $\mathcal{G}$ be the graph with a vertex set $\{v_1, \ldots, v_m\}$. We define a Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ as follows. For each edge $(v_\imath, v_\jmath)$ of $\mathcal{G}$ we define the clause $\mathcal{C}_{\imath\jmath} = x_\imath \vee x_\jmath$. The Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ is defined as the conjunction of all $\mathcal{C}_{\imath\jmath}$ such that $(v_\imath, v_\jmath)$ is an edge of $\mathcal{G}$. Let the weight $w_\imath$ of each $x_\imath$ be equal to 1. It is obvious that $\mathcal{G}$ has a vertex cover $\mathcal{S}$ of size $|\mathcal{S}| \leqslant \mathcal{P}$ iff the monotone Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ has a satisfying assignment with a total weight less than or equal to $\mathcal{P}$. $\square$

4.4 Computational Methods

This section describes three computational methods: the exact method, the utility propagation method, and the expenses propagation method. The exact method computes the exact adversarial utility, but it is way too complex to be used in practice and is used for research purposes only as a reference to the real exact utility. The two propagation methods – the utility propagation and the expenses propagation – produce the precise upper bound estimation in the case of independent attack trees. In the case of dependent attack trees the propagation methods approximate the upper bound from above and thus bring the necessity for even greater excessive investments into security, but such investments are balanced by the linear complexity of the propagation methods.

4.4.1 Utility Computation

The function which computes the exact adversarial utility can be expressed in the form of the recurrent relation (4.3):

$$\mathcal{U}(\mathcal{G}) = \max\left\{ 0,\ -\frac{\overline{\mathcal{E}}_\imath}{p_\imath} + \mathcal{U}(\mathcal{G}|_{x_\imath=1}),\ \mathcal{U}(\mathcal{G}|_{x_\imath=0}) \right\}\ , \tag{4.3}$$

with the seed values $\mathcal{U}(\mathbf{1}) = \mathcal{P}$ and $\mathcal{U}(\mathbf{0}) = 0$, and where x_i is any possible valid move in $\mathcal{G}$. Algorithm 4.4.1 allows to compute the exact adversarial utility in time exponential in the size of the game, the worst-case time complexity of the algorithm is $\mathcal{O}(2^n)$.

4.4.2 Expenses Computation

Following Theorem. 4.2.2:

$$\mathcal{U}(\mathcal{G}) = \max\left\{ 0,\ \mathcal{P} - \mathcal{E}(\mathcal{G}) \right\}\ , \tag{4.4}$$

ALGORITHM 4.4.1: Exact algorithm to compute adversarial utility

```
Input: Satisfiability game instance $\mathcal{G}$
Output: Adversarial utility (real number)
Function Utility ($\mathcal{G}$)
if $\mathcal{G} \equiv 1$ then
    return $\mathcal{P}$
else if $\mathcal{G} \equiv 0$ then
    return $0$
else
    return max $\left\{ 0,\ -\frac{\overline{\mathcal{E}}_i}{p_i} + \mathcal{U}\left(\mathcal{G}|_{x_i=1}\right),\ \mathcal{U}\left(\mathcal{G}|_{x_i=0}\right) \right\}$
```

thus it is possible to compute the exact adversarial utility by first computing the adversarial expenses using equation (4.5) and then subtracting it from the prize of the game using equation (4.4):

$$\mathcal{E}\left(\mathcal{G}\right) = \min\left\{ \mathcal{E}(x_\imath) + \mathcal{E}\left(\mathcal{G}|_{x_\imath=1}\right),\ \mathcal{E}\left(\mathcal{G}|_{x_\imath=0}\right) \right\}, \tag{4.5}$$

with seed values $\mathcal{E}(1) = \frac{\mathcal{C}_\imath}{p_\imath}$ and $\mathcal{E}(0) = 0$. Algorithm 4.4.2 allows us to compute the exact adversarial utility by calculating the adversarial expenses, in time exponential in the size of the game, the worst-case time complexity of the algorithm is $\mathcal{O}(2^n)$.

ALGORITHM 4.4.2: Exact algorithm to compute adversarial utility

```
Input: Satisfiability game instance $\mathcal{G}$
Output: Adversarial utility (real number)
Function Utility ($\mathcal{G}$)
return $\mathcal{P} -$ Expenses($\mathcal{G}$)

Input: Satisfiability game instance $\mathcal{G}$
Output: Adversarial expenses (real number)
Function Expenses ($\mathcal{G}$)
if $\mathcal{G} \equiv 1$ then
    return $\frac{\mathcal{C}_\imath}{p_\imath}$
else if $\mathcal{G} \equiv 0$ then
    return $0$
else
    return min $\left\{ \mathcal{E}(x_\imath) + \mathcal{E}\left(\mathcal{G}|_{x_\imath=1}\right),\ \mathcal{E}\left(\mathcal{G}|_{x_\imath=0}\right) \right\}$
```

4.4.3 Utility Propagation

The failure-free model [6] introduced the following relations:

$$\begin{aligned} \mathcal{U}\left(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k\right) &\leqslant \min\Big\{\mathcal{U}\left(\mathcal{G}_1\right), \ldots, \mathcal{U}\left(\mathcal{G}_k\right)\Big\} \ , \\ \mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) &\leqslant \mathcal{U}\left(\mathcal{G}_1\right) + \ldots + \mathcal{U}\left(\mathcal{G}_k\right) \ . \end{aligned} \tag{4.6}$$

The precision of inequality (4.6) can be improved by substituting it with equation:

$$\mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) = \max\Big\{\mathcal{U}\left(\mathcal{G}_1\right), \ldots, \mathcal{U}\left(\mathcal{G}_k\right)\Big\} \ . \tag{4.7}$$

It turns out that equation (4.7) is valid not only for the improved failure-free model, but for the failure-free model [6] as well. Besides, the expenses propagation method for the improved failure-free model introduced in Section 4.4.4 is even more precise.

The bottom-up utility propagation rule can be used to propagate adversarial utility from the leaves towards the root node in an attack tree using the following improved formulae:

$$\begin{aligned} \mathcal{U}\left(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k\right) &\leqslant \min\Big\{\mathcal{U}\left(\mathcal{G}_1\right), \ldots, \mathcal{U}\left(\mathcal{G}_k\right)\Big\} \ , \\ \mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) &= \max\Big\{\mathcal{U}\left(\mathcal{G}_1\right), \ldots, \mathcal{U}\left(\mathcal{G}_k\right)\Big\} \ . \end{aligned}$$

This way it is possible to obtain the adversarial utility upper bounds. Prior to continuing with the discussion it is necessary to introduce a couple of definitions.

Definition (Min-term). *By a min-term of a Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ we mean a conjunction of variables $x_{\imath_1} \wedge x_{\imath_2} \wedge \ldots \wedge x_{\imath_k}$ such that*

$$x_{\imath_1} \wedge x_{\imath_2} \wedge \ldots \wedge x_{\imath_k} \Rightarrow \mathcal{F}\left(x_1, \ldots, x_m\right)$$

is a tautology.

Definition (Critical min-term). *A min-term $x_1 \wedge \ldots \wedge x_k$ of $\mathcal{F}$ is critical if none of the sub-terms $x_1, \ldots, x_{\imath-1}, x_{\imath+1}, \ldots, x_m$ is a min-term of $\mathcal{F}$.*

Theorem 4.4.1. *Utility of the conjunctive game is less or equal to the minimum utility among its sub-games:*

$$\mathcal{U}\left(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k\right) \leqslant \min\Big\{\mathcal{U}\left(\mathcal{G}_1\right), \ldots, \mathcal{U}\left(\mathcal{G}_k\right)\Big\} \ . \tag{4.8}$$

Proof. Let $\mathcal{S}$ be an optimal strategy for the game $\mathcal{G} = (\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k)$. According to *Thm. 8* in [6], this strategy is equivalent to a non-adaptive strategy $x_{\imath_1} \wedge \ldots \wedge x_{\imath_m}$, where $x_{\imath_1} \wedge \ldots \wedge x_{\imath_m}$ is a critical min-term of the Boolean function that represents the current game instance, such that

$$x_{\imath_1} \wedge \ldots \wedge x_{\imath_m} \Rightarrow \mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k \Rightarrow \mathcal{G}_\imath$$

are tautologies. According to *Lemma 1* in [6] the utility upper bound of this game is less or equal to the utilities of its sub-games: $\mathcal{U}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) \leqslant \mathcal{U}(\mathcal{G}_\imath)$. As this holds for any $\imath$ this implies 4.8. $\square$

The proof is based on the fact that optimal strategies can be represented by critical min-terms of $\mathcal{F}$ and relies on *Lemma 1* and *Theorem 8* in [6]. *Lemma 1* states that if $\mathcal{B}$ is a sub-tree of $\mathcal{A}$ and $\mathcal{F}_{\mathcal{A}}(x_1, \ldots, x_k) \Rightarrow \mathcal{F}_{\mathcal{B}}(x_1, \ldots, x_k)$ is a tautology then $\mathcal{U}(\mathcal{A}) \leqslant \mathcal{U}(\mathcal{B})$. *Theorem 8* states that in the failure-free model attack trees $\mathcal{A}$ have non-adaptive optimal strategies, and a fixed ordering of moves $(x_{\imath_1}, \ldots, x_{\imath_k})$ that the optimal strategy follows is a critical min-term of the Boolean formula $\mathcal{F}$ of $\mathcal{A}$.

Theorem 4.4.2. *Utility of the disjunctive game equals to the maximum utility among its sub-games:*

$$\mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) = \max\Big\{\mathcal{U}\left(\mathcal{G}_1\right), \ldots, \mathcal{U}\left(\mathcal{G}_k\right)\Big\} \quad . \tag{4.9}$$

Proof. Let $\mathcal{S}$ be the optimal strategy for the game $\mathcal{G} = \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$. According to *Thm. 8* in [6], $\mathcal{S}$ can be represented as a critical min-term $x_{\imath_1} \wedge \ldots \wedge x_{\imath_m}$ of the Boolean function of $\mathcal{G} = \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$, this means that there exists such $\imath$ that $x_{\imath_1} \wedge \ldots \wedge x_{\imath_m}$ is a min-term of $\mathcal{G}_\imath$. Hence $x_{\imath_1} \wedge \ldots \wedge x_{\imath_m} \Rightarrow \mathcal{G}_\imath$ is a tautology. This means that

$$\begin{aligned}\mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) &= \mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\, , \mathcal{S}\right) \\ &\leqslant \mathcal{U}\left(\mathcal{G}_i\right) \leqslant \max\Big\{\mathcal{U}\left(\mathcal{G}_1\right), \ldots, \mathcal{U}\left(\mathcal{G}_k\right)\Big\} \quad .\end{aligned} \tag{4.10}$$

On the other hand, for any $\imath$ let $\mathcal{S}_\imath$ be the optimal strategy of $\mathcal{G}_\imath$. As

$$\mathcal{G}_\imath \Rightarrow \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$$

is a tautology, then

$$\mathcal{U}\left(\mathcal{G}_\imath\right) = \mathcal{U}\left(\mathcal{G}_\imath\, , \mathcal{S}_\imath\right) \leqslant \mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) \quad .$$

As $\imath$ was arbitrary, this implies

$$\max\Big\{\mathcal{U}\left(\mathcal{G}_1\right), \ldots, \mathcal{U}\left(\mathcal{G}_k\right)\Big\} \leqslant \mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) \quad . \tag{4.11}$$

Combining inequalities (4.10) and (4.11) we reach equation (4.9). $\square$

The disjunctive adversarial utility propagation rule produces more precise results compared to the similar rule in the Buldas *et al.* model [6]:

$$\mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) \leqslant \mathcal{U}\left(\mathcal{G}_1\right) + \ldots + \mathcal{U}\left(\mathcal{G}_k\right) \quad .$$

Algorithm 4.4.3 utilizes the conjunctive and disjunctive bottom-up adversarial utility propagation rules in every game instance starting from the elementary moves and ending up in the root instance of the game. The algorithm allows to compute the adversarial utility upper bound in time linear in the size of the game in the worst case.

Algorithm 4.4.3: Iterated utility propagation in the improved failure-free game

```
Input: Satisfiability game instance $\mathcal{G}$
Output: Utility upper bound (real number)
Function ComputeUtilityUpperBound ($\mathcal{G}$)
if $\mathcal{G}$ is an instance of a conjunctive game then
    /* $\mathcal{G}_1, \dots, \mathcal{G}_k$ are the sub-games of game $\mathcal{G}$ */
    return min $\left\{ \mathcal{U}(\mathcal{G}_1), \dots, \mathcal{U}(\mathcal{G}_k) \right\}$
else if $\mathcal{G}$ is an instance of a disjunctive game then
    /* $\mathcal{G}_1, \dots, \mathcal{G}_k$ are the sub-games of game $\mathcal{G}$ */
    return max $\left\{ \mathcal{U}(\mathcal{G}_1), \dots, \mathcal{U}(\mathcal{G}_k) \right\}$
else
    /* $\mathcal{G}$ is leaf */
    return $\mathcal{U}(\mathcal{G})$
```

4.4.4 Expenses Propagation

As the current implementation of the improved failure-free model does not consider any intermediate pay-offs, the prize $\mathcal{P}$ is fixed and the adversary gets it once he wins the game. This makes it possible to obtain the adversarial utility upper bounds using the expenses propagation method. Propagating expenses we obtain the lower bound for the adversarial expenses in the game $\mathcal{E}(\mathcal{G})$. The utility upper bound can be obtained according to Theorem 4.2.2 using formula $\mathcal{U}(\mathcal{G}) = \mathcal{P} - \mathcal{E}(\mathcal{G})$. The expenses lower bounds in the case of independent games can be computed using equations (4.12):

$$\begin{aligned} \mathcal{E}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k) &= \min \left\{ \mathcal{E}(\mathcal{G}_1), \ldots, \mathcal{E}(\mathcal{G}_k) \right\} \ , \\ \mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) &= \mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k) \ . \end{aligned} \tag{4.12}$$

The rule for the disjunctive game holds in the case of independent, as well as in the case of dependent games. However, the rule for the conjunctive game holds only in the case of independent games. In order to obtain the expenses lower bounds using the expenses propagation method, in the case of a disjunctive game the expenses of the cheapest sub-game are propagated, and in the case of a conjunctive game the sum of the expenses of all the sub-games in propagated towards the root node.

Theorem 4.4.3. *Strategy $\mathcal{S}_{opt}$ that is optimal for the game $\mathcal{G} = \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$ contains only one single sub-game $\mathcal{G}_\imath$ of game $\mathcal{G}$ with minimal expenses $\mathcal{E}(\mathcal{G}_\imath)$.*

Proof. Let $\mathcal{S}$ be the optimal strategy for the game $\mathcal{G} = \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$. According to *Thm. 8* in [6] it is equivalent to a non-adaptive strategy $x_{\imath_1} \wedge \ldots \wedge x_{\imath_m}$ where

$x_{\imath_1} \wedge \ldots \wedge x_{\imath_m}$ is a critical min-term of the Boolean function that represents the current game $\mathcal{G} = \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$, such that

$$x_{\imath_1} \wedge \ldots \wedge x_{\imath_m} \Rightarrow \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k \Rightarrow \mathcal{G}_\imath$$

is a tautology. This means that we are playing one single sub-game from the set of sub-games of the current game. For the game with k sub-games there are at least k strategies. Playing one single sub-game that is the cheapest one we get the maximal possible utility value for the parent game, thus this is the optimal strategy. □

Theorem 4.4.4. *The expenses of the disjunctive game are equal to the minimal expenses among its sub-games:*

$$\mathcal{E}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) = \min\left\{\mathcal{E}\left(\mathcal{G}_1\right), \ldots, \mathcal{E}\left(\mathcal{G}_k\right)\right\} \ .$$

Proof. According to *Thm. 4.2.1*:$\mathcal{U}(\mathcal{G}) = \mathcal{P} - \mathcal{E}(\mathcal{G})$. As $\mathcal{P}$ is a constant $\mathcal{U}(\mathcal{G})$ can be maximized by minimizing $\mathcal{E}(\mathcal{G})$. According to *Thm. 8* in [6]:

$$\mathcal{U}\left(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k\right) \leqslant \mathcal{U}\left(\mathcal{G}_1\right) + \ldots + \mathcal{U}\left(\mathcal{G}_k\right) \ .$$

Thus,

$$\max\left\{\mathcal{U}(\mathcal{G}_1), \ldots, \mathcal{U}(\mathcal{G}_k)\right\} = \mathcal{P} - \mathcal{E}(\mathcal{G}_1, \ldots, \mathcal{G}_k) = \mathcal{P} - \min\left\{\mathcal{E}(\mathcal{G}_1), \ldots, \mathcal{E}(\mathcal{G}_k)\right\} \ .$$

□

Thus the utility of a disjunctive game can be computed as for the elementary case:

$$\mathcal{U}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k) = \mathcal{P} - \mathcal{E}(\mathcal{G}_\imath) \ ,$$

where $\mathcal{G}_\imath$ is the cheapest sub-game of $\mathcal{G}$. It may seem natural that the expenses of a conjunctive game should be equal to the sum of expenses of its sub-games:

$$\mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) = \sum_{\imath=1}^{k} \mathcal{E}(\mathcal{G}_\imath) \ .$$

It turns out that this statement is true only for the case of the so-called *independent games*.

Definition (Independent satisfiability games). *Independent satisfiability games are games containing no common moves, their corresponding Boolean formulae contain no common variables. On the other hand, dependent satisfiability games are games which contain common moves, meaning that a single move can be a part of several scenarios.*

If $\mathcal{G}$ contains common moves or even common sub-games $\mathcal{E}(\mathcal{G})$ will be less than the sum of expenses of its sub-games:

$$\mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) < \sum_{\imath=1}^{k} \mathcal{E}(\mathcal{G}_\imath) \enspace .$$

Theorem 4.4.5. $\mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) \leqslant \mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k)$.

Proof. Let $\mathcal{G} = \mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k$ and let $\mathcal{S}_1, \ldots, \mathcal{S}_k$ be optimal strategies for the sub-games $\mathcal{G}_1, \ldots, \mathcal{G}_k$, respectively. Let $\mathcal{S}$ be a strategy which plays all the sub-games simultaneously using strategies $\mathcal{S}_{\imath \in k}$. The expenses of $\mathcal{S}$ are $\mathcal{E}(\mathcal{G}, \mathcal{S}) = \mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k)$. Therefore:

$$\mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) \leqslant \mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k) \enspace .$$

□

Thus:

$$\max\Big\{\mathcal{E}(\mathcal{G}_1), \ldots, \mathcal{E}(\mathcal{G}_k)\Big\} \leqslant \mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) \leqslant \mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k) \enspace .$$

Let $\mathcal{G} = \mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k$ be an independent conjunctive game. For a conjunctive game to succeed all its games need to be played and succeed. Thus, the expenses value of the independent conjunctive game is the sum of the expenses of all its sub-games.

Theorem 4.4.6. *The expenses of a conjunction of independent games is equal to the sum of expenses of its sub-games:*

$$\mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) = \mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k) \enspace . \tag{4.13}$$

Proof. When sub-games $\mathcal{G}_1, \ldots, \mathcal{G}_k$ are independent, every move $x_\imath$ changes the result of only one of the sub-games of game $\mathcal{G}$. The expenses that the player needs to pay in order to win all sub-games are greater or equal to the sum of expenses of the sub-games of the game. This means that

$$\mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) \leqslant \mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k) \enspace .$$

Combining with Thm. 4.4.5 we reach an equation (4.13). □

Algorithm 4.4.4 utilizes the conjunctive and disjunctive bottom-up adversarial expenses propagation rules to calculate the lower bound of adversarial expenses. The utility upper bound is then obtained by the subtraction of the expenses from the prize of the game. The algorithm allows to compute the adversarial utility upper bound in time linear in the size of the game in the worst case.

ALGORITHM 4.4.4: Iterated expenses propagation in the improved failure-free game, assuming that all conjunctive nodes have independent sub-trees

```
    Input: Satisfiability game instance $\mathcal{G}$
    Output: Utility upper bound (real number)
1   Procedure ComputeUtilityUpperBound ($\mathcal{G}$)
2   return $\mathcal{P}$ − ComputeExpensesLowerBound($\mathcal{G}$)
3
    Input: Satisfiability game instance $\mathcal{G}$
    Output: Expenses lower bound (real number)
4   Procedure ComputeExpensesLowerBound ($\mathcal{G}$)
5   if $\mathcal{G}$ is a conjunction of independent games then
6       /* $\mathcal{G}_1, \ldots, \mathcal{G}_k$ are the sub-games of game $\mathcal{G}$ */
7       return $\mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k)$
8   else if $\mathcal{G}$ is an instance of a disjunctive game then
9       /* $\mathcal{G}_1, \ldots, \mathcal{G}_k$ are the sub-games of game $\mathcal{G}$ */
10      return $\min\{\mathcal{E}(\mathcal{G}_1), \ldots, \mathcal{E}(\mathcal{G}_k)\}$
11  else
12      /* $\mathcal{G}$ is leaf */
13      return $\mathcal{E}(\mathcal{G})$
```

4.5 EXPENSES REDUCTION

This section outlines the problem associated with the dependent games and suggests a solution to it. Dependent games are games having common moves like is shown in Fig. 3.2.

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be sub-games of game $\mathcal{G}$. These sub-games may in turn contain the conjunctive as well as disjunctive sub-games alternately with no evidence if these sub-games contain no common moves. It is reasonable to assume that some of the sub-games may contain common moves and that the optimal strategy might suggest to launch them. Thus these common moves multiply their corresponding investments into the expenses parameter that these nodes propagate. Let $\mathcal{G} = \mathcal{F}_1(x, x_1, \ldots, x_m) \wedge \mathcal{F}_2(x, x_1, \ldots, x_n)$ be a Boolean formula with a common variable x. Let us call the number of times that x is present in $\mathcal{G}$ the rank of x and denote it by $r(x)$. The idea behind cost reduction is the substitution of every occurrence of x with independent variable $x'_\imath$ such that $\sum\limits_{\imath=1}^{r(x)} x'_\imath = w(x)$. Although by reducing the expenses of the common moves we make them easier to play, the idea behind this is that if the system can be proven to be secure even if some of the attacks are artificially made easier than they really are, this implies that the attacks against the real organization are infeasible. Let us assume that we have an attack tree which is really a tree the leaves of which are attacks except that two leaves may represent the same attack. For example, $\mathcal{T} = (x_1 \wedge x_2) \vee (x_2 \wedge x_3)$, where x_2 is used twice. We say that the rank $r(x_2) = 2$. When we use expenses reduction, we can replace attack x_2 with independent attacks x_2' and x_2'' and reduce their costs so that $\mathcal{C}_{x_2'} + \mathcal{C}_{x''_2} = \mathcal{C}_x$. After such change the cost propagation method is exact. If we replace all the instances of $\mathcal{E}_\imath$ with $\frac{1}{2}\mathcal{E}_\imath$, then the result is no more than twice less than the original value. This leads to the following theorem:

Theorem 4.5.1. *Let N be the maximal rank of an elementary attack. Then the expenses reduction method gives the expenses lower bound that is not less than N times the exact value of the expenses.*

Proof. As the adversarial expenses can be expressed in the form of a linear function $f(a, b) = \alpha a + \beta b$:

$$f(\frac{1}{N}a, b) = \alpha\frac{a}{N} + \beta b \geqslant \alpha\frac{a}{N} + \beta\frac{b}{N} = \frac{1}{N}(\alpha a + \beta b) = \frac{1}{N}f(a, b) \ .$$
$$f(\frac{a}{N}, \frac{b}{N}) = \alpha\frac{a}{N} + \beta\frac{b}{N} = \frac{1}{N}f(\alpha a + \beta b) = \frac{1}{N}f(a, b) \ .$$

□

Theorem 4.5.2. *Let $\mathcal{F}(x_1, \ldots, x_n)$ be a monotone Boolean formula. Let $\mathcal{G}$ be a sub-formula in $\mathcal{F}$ that is used at least two times, i.e. if $\mathcal{F}$ is represented as a Boolean circuit we can imagine $\mathcal{G}$ as a formula with output followed by a fan-out component. Let*

$$\mathcal{F}(x_1, \ldots, x_n) = \tilde{\mathcal{F}}(x_1, \ldots, x_n, \mathcal{G}(x_1, \ldots, x_n)) \ ,$$

where $\tilde{\mathcal{F}}$ is a formula with less number of fan-outs compared to $\mathcal{F}$. Then in terms of Boolean functions:

$$\mathcal{F}(x_1, \ldots, x_n) \equiv \tilde{\mathcal{F}}(x_1, \ldots, x_n, 0) \vee \Big(\mathcal{G}(x_1, \ldots, x_n) \wedge \tilde{\mathcal{F}}(x_1, \ldots, x_n, 1)\Big) \ . \quad (4.14)$$

Proof. Indeed, let $\mathcal{A} : x_\imath = a_\imath$ be a satisfying assignment to $\mathcal{F}$, such that $\mathcal{F}(a_1, \ldots, a_n) = 1$. If $\mathcal{G}(a_1, \ldots, a_n) = 0$ then $\mathcal{A}$ is also a satisfying assignment to $\tilde{\mathcal{F}}(x_1, \ldots, x_n, 0)$ and hence, also of the right hand side of (4.14). If $\mathcal{G}(a_1, \ldots, a_n) = 1$ then $\mathcal{A}$ is a satisfying assignment to both $\mathcal{G}(x_1, \ldots, x_n)$ and $\tilde{\mathcal{F}}(x_1, \ldots, x_n, 1)$ and hence also of the right hand side of (4.14). Let now $\mathcal{A} : x_\imath = a_\imath$ be a satisfying assignment to the right hand side of (4.14). Then either $\mathcal{A}$ satisfies $\tilde{\mathcal{F}}(x_1, \ldots, x_n, 0)$ or $\mathcal{A}$ satisfies both $\mathcal{G}(x_1, \ldots, x_n)$ and $\tilde{\mathcal{F}}(x_1, \ldots, x_n, 1)$. In the former case due to monotonicity $\mathcal{A}$ also satisfies

$$\mathcal{F}(x_1, \ldots, x_n) = \tilde{\mathcal{F}}(x_1, \ldots, x_n, \mathcal{G}(x_1, \ldots, x_n)) \ .$$

In the latter case $\mathcal{A}$ satisfies the conjunction $\mathcal{G}(x_1, \ldots, x_n) \wedge \tilde{\mathcal{F}}(x_1, \ldots, x_n, 1)$ and hence also $\mathcal{F}(x_1, \ldots, x_n)$. □

Hence, using the Shannon expansion (4.14) we can reduce an attack tree with fan-outs to two instances $\tilde{\mathcal{F}}(x_1,\ldots,x_n,0)$ and $\mathcal{G}(x_1,\ldots,x_n) \wedge \tilde{\mathcal{F}}(x_1,\ldots,x_n,1)$ of attack trees with smaller number of fan-outs. Iteratively eliminating fan-out components we eventually reach the state in which the transformed attack tree will have no fan-outs and therefore will be completely independent. This way, non-atomic fan-outs can be handled with in a similar way compared to atomic fan-outs. It can be shown that this way of fan-out elimination converges.

Theorem 4.5.3. *Iterative application of Shannon expansion to eliminate fan-outs in an attack tree eventually results in an attack tree without fan-out components.*

Proof. Let $\phi(\mathcal{F})$ be the number of fan-outs of function $\mathcal{F}(x_1,\ldots,x_n)$ and $\phi(\tilde{\mathcal{F}})$ be the number of fan-outs of function $\tilde{\mathcal{F}}(x_1,\ldots,x_n,\mathcal{G}(x_1,\ldots,x_n))$. Here we assume that $\mathcal{G}(x_1,\ldots,x_n)$ is not a projector $\mathcal{G} \equiv x_\imath$ does not hold – is more or less complex function. Then $\phi(\mathcal{F}) = \phi(\tilde{\mathcal{F}}) + \phi(\mathcal{G}+1)$. $\tilde{\mathcal{F}}(x_1,\ldots,x_n,0)$ contains $\phi(\tilde{\mathcal{F}})$ fan-outs which is less than $\phi(\mathcal{F})$. $\tilde{\mathcal{F}}(x_1,\ldots,x_n,1) \wedge \mathcal{G}(x_1,\ldots,x_n)$ contain $\phi(\tilde{\mathcal{F}}) + \phi(\mathcal{G})$ number of fan-outs which is as well less than $\phi(\mathcal{F})$. This shows that method converges as with each subsequent iteration of such transformation the number of fan-outs reduces, thus the overall amount of fan-outs reduces and eventually the number of fan-outs reduces to 0. $\square$

Lets assume we have only one variable $x_\imath$ and $r(x_\imath) = 100$. In this case it would be not good to use expenses reduction as it would give an upper bound estimate that can be 100 times higher than the real value. What we can do in this case is to use a single step of branching first, i.e. using the formula

$$\mathcal{E}\left(\mathcal{G}\right) = \min\left\{\ \mathcal{E}(x_\imath) + \mathcal{E}\left(\mathcal{G}|_{x_\imath=1}\right),\ \mathcal{E}\left(\mathcal{G}|_{x_\imath=0}\right)\ \right\} \quad ,$$

but instead of continuing the recursion we compute $\mathcal{E}\left(\mathcal{G}|_{x_\imath=1}\right)$ and $\mathcal{E}\left(\mathcal{G}|_{x_\imath=0}\right)$ with expenses propagation. Note that as $x_\imath$ was the only “branching” elementary attack, the values we get are exact. This leads to the following theorem:

Theorem 4.5.4. *If a monotone Boolean function $\mathcal{F}$ has m variables x with rank $r(x) > 1$ then $\mathcal{E}(\mathcal{G})$ can be computed in time $\mathcal{O}(|\mathcal{F}|)\cdot 2^m$.*

Proof. Let $\mathcal{F}(x_1,\ldots,x_m,x_{m+1},\ldots,x_n)$ be a monotone Boolean function where variables $x_1,\ldots,x_m$ have rank $r(x_\imath) > 1$, and variables $x_{m+1},\ldots,x_n$ have rank 1. Let $v_1,\ldots,v_m \in \{0,1\}$ and let $(v_1,\ldots,v_m)$ denote the assignment $x_1 = v_1$, $\ldots, x_m = v_m$. By using the Shannon expansion, we have:

$$\mathcal{E}(\mathcal{F}) = \min_{(v_1,\ldots,v_m)} \Big\{(v_1\cdot w_1 + \ldots + v_m\cdot w_m) + \mathcal{E}(\mathcal{F}(v_1,\ldots,v_m,x_{m+1},\ldots,x_n))\Big\} \quad ,$$

where the minimum is computed over all possible 2^m assignments. As $\mathcal{E}(\mathcal{F}(v_1,\ldots,v_m,x_{m+1},\ldots,x_n))$ can be computed in time $\mathcal{O}(|\mathcal{G}|)$, the overall complexity of finding $\mathcal{E}(\mathcal{F})$ is $\mathcal{O}(|\mathcal{G}|)\cdot 2^m$. $\square$

The following theorem shows that cost reduction does not lead to false-positive results by not making the attack process harder for the attacker.

Theorem 4.5.5. *Let*

$$\mathcal{F}(x, x_1, \ldots, x_n) = \mathcal{F}_1(x, x_1, \ldots, x_n) \wedge \mathcal{F}_2(x, x_1, \ldots, x_m)$$

be a monotone Boolean function, and

$$\mathcal{F}'(x', x'', x_1, \ldots, x_m) = \mathcal{F}_1(x', x_1, \ldots, x_m) \wedge \mathcal{F}_2(x'', x_1, \ldots, x_m)$$

be another Boolean function derived from $\mathcal{F}$ by using substitutions $x \mapsto x'$ and $x \mapsto x''$. The weights are related in the following way: $w(x') + w(x'') = w(x)$. Let μ be a min-term of $\mathcal{F}$ and μ' be a min-term of $\mathcal{F}'$. Then for any min-term μ of $\mathcal{F}$ there exists a min-term m' of $\mathcal{F}'$ such that $w(\mu') \leqslant w(\mu)$.

Proof. Let $\mu = x_{\imath_1} \wedge \ldots \wedge x_{\imath_k}$. By assumptions, $\mu \Rightarrow \mathcal{F}(x, x_1, \ldots, x_m)$ is a tautology, and hence

$$\begin{aligned} \mu &\Rightarrow \mathcal{F}_1(x, x_1, \ldots, x_m)\ , \\ \mu &\Rightarrow \mathcal{F}_2(x, x_1, \ldots, x_m) \end{aligned}$$

are also tautologies. First, let us assume that $x \notin \{x_{\imath_1}, \ldots, x_{\imath_k}\}$. Considering that tautologies will be preserved under any variable substitution $x_\imath \mapsto x_\jmath$ and therefore we can substitute x with x' and x'' so that

$$\begin{aligned} \mu &\Rightarrow \mathcal{F}_1(x', x_1, \ldots, x_m)\ , \\ \mu &\Rightarrow \mathcal{F}_2(x'', x_1, \ldots, x_m) \end{aligned}$$

are tautologies. This means that

$$\mu \Rightarrow \mathcal{F}_1(x', x_1, \ldots, x_m) \wedge \mathcal{F}_2(x'', x_1, \ldots, x_m) = \mathcal{F}'(x', x'', x_1, \ldots, x_m)$$

is also a tautology. Hence, μ is also a min-term of $\mathcal{F}'$.
Now consider the case when μ contains the common attack x and similarly to the previous case let $\mu = x \wedge x_{\imath_1} \wedge \ldots \wedge x_{\imath_k}$ be a min-term of $\mathcal{F}$. This means that

$$\begin{aligned} x \wedge x_{\imath_1} \wedge \ldots \wedge x_{\imath_k} &\Rightarrow \mathcal{F}_1(x, x_1, \ldots, x_m)\ , \\ x \wedge x_{\imath_1} \wedge \ldots \wedge x_{\imath_k} &\Rightarrow \mathcal{F}_2(x, x_1, \ldots, x_m) \end{aligned}$$

are tautologies. Hence, also

$$\begin{aligned} x' \wedge x_{\imath_1} \wedge \ldots \wedge x_{\imath_k} &\Rightarrow \mathcal{F}_1(x', x_1, \ldots, x_m)\ , \\ x'' \wedge x_{\imath_1} \wedge \ldots \wedge x_{\imath_k} &\Rightarrow \mathcal{F}_2(x'', x_1, \ldots, x_m) \end{aligned}$$

are tautologies, which means that

$$\begin{aligned} \mu' &= x' \wedge x'' \wedge x_{\imath_1} \wedge \ldots \wedge x_{\imath_k} \\ &\Rightarrow \mathcal{F}_1(x', x_1, \ldots, x_m) \wedge \mathcal{F}(x'', x_1, \ldots, x_m) = \mathcal{F}'(x', x'', x_1, \ldots, x_m) \end{aligned}$$

is also a tautology. As $w(x') + w(x'') = w(x)$ we have $w(\mu) = w(\mu')$. □

4.6 Conclusions and Future Research

This chapter introduced the improved failure-free model in which adversaries are not limited in any way and may repeat failed attacks again an arbitrary number of times. It was shown that in the new model optimal strategies always exist, and there are non-adaptive optimal strategies in the form of a single-branched BDDs with self-loops. Due to the infinite adversarial budget assumption the order in which the attacker launches attacks is irrelevant. The problem of finding an optimal strategy in such a repeatable satisfiability game – the weighted monotone satisfiability problem – was shown to be NP-complete. Several computational methods have been developed. The two methods to compute precise outcome are quite complex and can hardly be used to analyze attack scenarios of practical sizes. It turned out that removing adversarial limitations from the failure-free model [6] allowed to make the model and computational methods easier and allowed to come up with efficient propagation methods, which calculate reliable upper bounds of the adversarial utility and can come up with the result in time linear in the size of an attack tree. This makes the propagation methods very efficient, however they are limited to analyzing the so-called independent attack trees – trees having no common moves.

Using Shannon expansion it is possible to eliminate fan-outs in the tree and make it independent. The expansion can be used to eliminate fan-outs from the leaves as well as from the intermediate nodes in an attack tree. Indeed the Shannon expansion can make a tree independent, but each application of it increases the search space. The subsequent research focuses on determining the optimal combinations of the propagation, expenses reduction algorithm and Shannon expansion to obtain as exact result as we can (this reduces the need for excessive investments into security), at the same time remain robust, so that the computational methods can be used in practice.

Chapter 5

Improved Failure-Free Model with Limited Budget

The improved failure-free model [5] assumes unlimited adversarial budget. This model provides reliable upper bounds however this may result in over-secured systems. It has not been studied how much extra cost the upper bound oriented methods cause. It is natural to assume that the adversarial budget is limited and such an assumption would allow to model the adversarial decision making more close to the one that may happen in real life. Introducing such a natural limitation increases the modeling granularity and thus it may be considered as a step in the direction to the real world. Such a step is expected to bring extra complexity along but it might not be so big as was the case with the existing models and would be justified if the method which considers limited budget of the adversary produced more precise or reliable results compared to the improved failure-free model.

This chapter focuses on fully-adaptive adversarial strategies assuming that the adversarial budget is limited and presents the limited improved failure-free model [22] in which the limitation placed on the adversarial budget is the only limitation applied to the adversary compared to the failure-free model. The following three cases are analyzed: the single attack case, the elementary conjunction, and the elementary disjunction. Based on the three elementary cases the effect of limiting adversarial budget in the fully-adaptive strategies is studied and analyzed. The reliability and precision aspects of the limited model are also studied and compared to reliability and precision of the improved failure-free model. It is shown that the single attack case as well as the elementary disjunction does not produce more precise or reliable results whatsoever compared to the improved failure-free model. The elementary conjunction can produce more precise result iff the adversarial reward is estimated with the required precision which in real-life scenarios may be less than 1€. If the analysts fail to do that, the results of the analysis are unreliable. In practice it is doubtful that analysts would be able to come up with such precise estimations. Even if such precise estimations existed the analysis still would not produce reliable results as there is still a margin for human error, e.g.

in the case when quantitative annotations on attacks such as cost of an attack or success probability are overlooked. On the contrary the improved failure-free model with unlimited budget provides reliable upper bounds despite the fact that such an upper-bound oriented approach may result in over-secured systems.

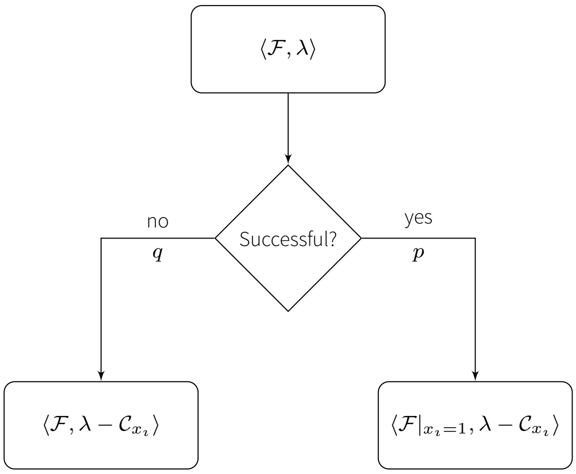
It seems that limiting adversarial budget has made the computational methods much more complex compared to the improved failure-free model which assumed unlimited budgets. For instance, optimal strategies that were shown to be non-adaptive in the improved failure-free model are almost always fully-adaptive in the limited budget model. Besides, the fully adaptive strategies are more complex and require bigger computational effort to analyze them. The best move to undertake in certain states of the game changes bouncing between the attacks. Even the elementary cases studied in this research become quite complex if to consider budget limitations compared to the improved failure-free model. It is doubtful that in the general case there exists a graceful and easy solution to derive optimal strategies. Considering the requirement to be able to estimate the adversarial reward with precision not achievable in practice it is hardly reasonable to face the complexity of the calculations of the adaptive strategies considering limited adversarial budget.

5.1 Limited Failure-Free Satisfiability Game

When we introduce a limitation on the adversarial budget to the improved failure-free model, the adversaries still behave in a fully-adaptive way and are allowed to launch failed attacks again in any order until the budget gets so small that no attacks can be launched. When the budget decreases by a considerable amount, the budget limitation starts affecting possible strategic choices of the adversary. The set of possible choices reduces – the adversary may launch only some subset of the set of attacks available in the beginning of the game – and eventually this subset becomes an empty set. It turns out that the optimal strategy depends on the amount of monetary resource available to the adversary.

In the improved failure-free model the state of the game is represented by the Boolean function $\mathcal{F}$. If the attack failed the adversary found himself in the very same instance of the game $\mathcal{F}$ because of the unlimited budget assumption. Due to this non-adaptive strategies always exist in the set of optimal strategies of the game. This is not always the case when we consider budget limitations – in general, optimal strategies are fully-adaptive, except for some certain sets of quantitative annotations in the case of which optimal strategies are non-adaptive. When we consider budget limitations, the state of the game is represented by the Boolean function $\mathcal{F}$ and the budget λ denoted by $\langle \mathcal{F}, \lambda \rangle$. Every move undertaken by the adversary in the game results in transition in the state of the game. Thus if the adversary launches a move $x_\imath$ and it succeeds, the new state of the game is denoted by $\langle \mathcal{F}|_{x_\imath=1}, \lambda - \mathcal{C}_{x_\imath} \rangle$ and if $x_\imath$ fails the new state of the game is

$\langle \mathcal{F}|_{x_\imath=0}, \lambda - \mathcal{C}_{x_\imath}\rangle \equiv \langle \mathcal{F}, \lambda - \mathcal{C}_{x_\imath}\rangle$, where $\mathcal{C}_{x_\imath}$ is the cost of move $x_\imath$. After the move has been executed the new iteration of the game starts. Possible state transitions in the game can be graphically described as shown in Fig. 5.1.



$\langle \mathcal{F}, \lambda\rangle$
Successful?
no
q
yes
p
$\langle \mathcal{F}, \lambda - \mathcal{C}_{x_\imath}\rangle$
$\langle \mathcal{F}|_{x_\imath=1}, \lambda - \mathcal{C}_{x_\imath}\rangle$

Figure 5.1: The attacker game in the limited improved fully adaptive model

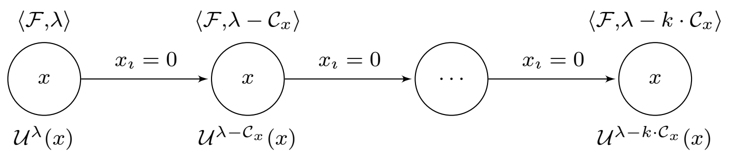
Let us denote the adversarial utility in a state $\langle \mathcal{F}, \lambda\rangle$ by $\mathcal{U}^\lambda(\mathcal{F})$. The relation between the adversarial utility upper bound $\mathcal{U}^\infty(\mathcal{F})$ in the limited failure-free model [5] and the utility $\mathcal{U}^\lambda(\mathcal{F})$ in the limited improved failure-free model is the following:

$$\mathcal{U}^\infty(\mathcal{F}) = \lim_{\lambda\to\infty} \mathcal{U}^\lambda(\mathcal{F}) \ .$$

Indeed in some cases optimal strategies are non-adaptive, but in the general case optimal strategies in the limited improved failure-free model are adaptive which makes computations reasonably complex. With gradual increase in adversarial budget his utility increases as well and in the case of infinite growth of adversarial budget it approaches the utility upper bound in the improved failure-free model. It turns out that in the case of a reasonably big budget the complexity added by the budget limitation does not add any value nor give any additional benefits, as the difference between the utility upper bound in the limited improved failure-free model and the utility upper bound in the improved failure-free model becomes negligible.

5.2 Single Elementary Attack Case

Let us consider the simplest possible case – when the adversary has just one single choice and can choose either to run attack x or not. If the adversary can choose from only one single choice, he will continue launching the attack corresponding to this choice until it succeeds, or as long as the budget is sufficient for re-running it. Such a strategy may be represented in the form of a single-branched BDD as in Fig. 5.2.



$\langle \mathcal{F},\lambda \rangle$
$\langle \mathcal{F},\lambda - \mathcal{C}_x \rangle$
$\langle \mathcal{F},\lambda - k\cdot\mathcal{C}_x \rangle$
x
$x_\imath = 0$
x
$x_\imath = 0$
$\cdots$
$x_\imath = 0$
x
$\mathcal{U}^{\lambda}(x)$
$\mathcal{U}^{\lambda-\mathcal{C}_x}(x)$
$\mathcal{U}^{\lambda-k\cdot\mathcal{C}_x}(x)$

Figure 5.2: An adaptive strategy suggesting to iterate attack x until it succeeds or as long as the adversarial budget allows to launch the attack

In accordance with the strategy the adversary launches the attack x annotated with cost $\mathcal{C}_x$ and success probability p_x. If x succeeds the adversary has accomplished the attack and has won the game. If x fails, the adversary finds himself in another state of the game – $\langle x, \lambda - \mathcal{C}_x \rangle$. Thus adversarial utility may be expressed in the form of relation (5.1):

$$\mathcal{U}^{\lambda}(x) = \max\Big\{0, \mathcal{U}(x) + (1-p_x)\cdot\mathcal{U}^{\lambda-\mathcal{C}_x}(x)\Big\} \ . \tag{5.1}$$

Thus, when the adversary launches attack x just a single time his utility is $\mathcal{U}(x)$. If the same attack is launched two times, utility is $\mathcal{U}(x) + (1-p_x)\mathcal{U}(x)$. If the attacker launches attack x n times, the corresponding utility is:

$$\begin{aligned}
\mathcal{U}^{\lambda}(x) &= \mathcal{U}(x) + (1-p_x)\cdot\Big[\mathcal{U}_x + (1-p_x)\cdot\Big[\mathcal{U}(x) + \ldots + (1-p_x)\mathcal{U}(x) \\
&= \mathcal{U}(x) + (1-p_x)\cdot\mathcal{U}(x) + \ldots + (1-p_x)^{n-1}\cdot\mathcal{U}(x) \\
&= \mathcal{U}(x)\cdot\Big[1 + (1-p_x) + (1-p_x)^2 + \ldots + (1-p_x)^{n-1}\Big] \ .
\end{aligned}$$

The series $\Big[1 + (1-p_x) + (1-p_x)^2 + \ldots + (1-p_x)^{n-1}\Big]$ is a geometric sum which is:

$$\sum_{k=0}^{n-1}(1-p_x)^k = \frac{1-(1-p_x)^n}{p_x} \ , \tag{5.2}$$

we reach the following equation:

$$\mathcal{U}^{\lambda}(x) = \frac{\mathcal{U}(x)}{p_x}\cdot\Bigg[1-(1-p_x)^n\Bigg] \ .$$

It can be seen as well that

$$\mathcal{U}^{\lambda}(x) = \mathcal{U}(x)\cdot\sum_{k=0}^{\infty}(1-p_x)^k = \frac{\mathcal{U}(x)}{p_x} = \Big[\mathcal{P} - \frac{\mathcal{C}(x)}{p_x}\Big] = \mathcal{U}^{\infty}(x) \ .$$

Considering that x was launched n times: $\lambda = n \cdot \mathcal{C}_x$ and thus $n = \left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor$. Therefore:

$$\mathcal{U}^\lambda(x) = \underbrace{\left[\mathcal{P} - \frac{\mathcal{C}(x)}{p_x}\right]}_{\mathcal{U}^\infty(x)} \cdot \left[1 - (1 - p_x)^{\left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor}\right] \quad . \tag{5.3}$$

Equation (5.3) sets the relation between the limited budget model and the improved failure-free model and basically says that the utility in the limited budget model is $1 - (1 - p_x)^{\left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor}$ times smaller than the utility upper bound. As $1 - (1 - p_x)^{\left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor}$ is a progression which grows from 0 towards 1 with increase in $\left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor$ it is obvious that in the case of infinite λ the value $1 - (1 - p_x)^{\left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor}$ approaches 1 and the utility in the limited budget model approaches the utility upper bound in the improved failure-free model.

It can be seen in Fig. 5.3 that the adversarial utility changes in the points where the budget is a multiple of the cost of the considered attack. If the adversarial budget is less than the cost of the attack the budget is insufficient to launch the attack even a single time and thus the utility is 0. The optimal strategy in such a case is empty – the attacker will be better off even not trying to attack and doing nothing. In case the adversarial budget exceeds the cost of an attack, the utility grows with each subsequent trial to launch an attack, as every subsequent trial increases the probability that the attack will succeed. Thus the bigger the budget is, the more times the adversary can re-run the same attack, the greater is the adversarial utility and eventually it approaches the utility upper bound in the model without budget limitations, as shown in Fig. 5.3.

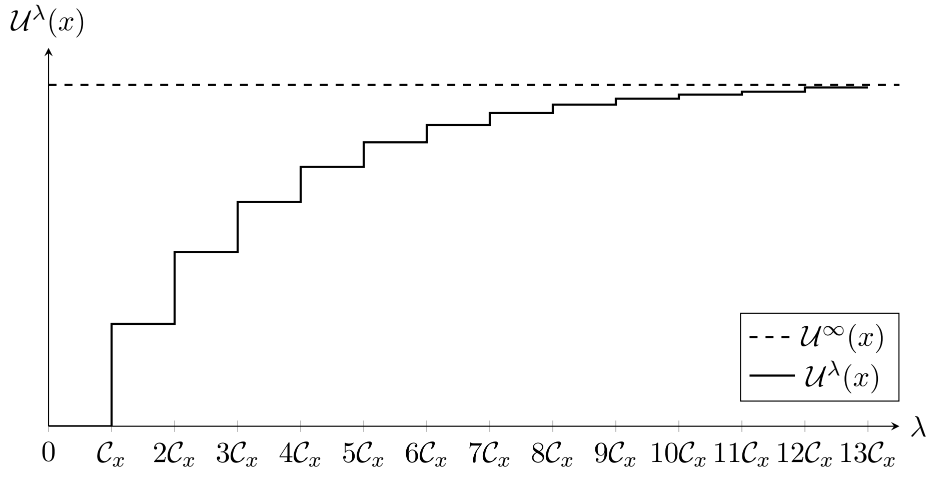


Figure 5.3: Adversarial utility in the single attack case

5.2.1 Comparison with the Improved Failure-Free Model

We are particularly interested if the limited budget assumption can produce results different from the results in the improved failure-free model – e.g. when the improved failure-free model analysis result says that the analyzed infrastructure is *insecure* while the limited budget model analysis result says that the analyzed infrastructure is *secure*.

Theorem 5.2.1. *If an elementary single attack game is unprofitable in the improved failure-free model with limited budget, it will be unprofitable in the improved failure-free model with unlimited budget.*

Proof. In the case of a single attack the adversarial utility upper bound in the improved failure free model is

$$\mathcal{U}^{\infty}(x) = \mathcal{P} - \frac{\mathcal{C}_x}{p_x} \ ,$$

and in the limited budget model the adversarial utility in the same case given budget λ will be

$$\mathcal{U}^{\lambda}(x) = \left[\mathcal{P} - \frac{\mathcal{C}_x}{p_x}\right] \cdot \left[1 - (1 - p_x)^{\left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor}\right] \ .$$

The result produced by the limited budget model will differ from the result obtained in the improved failure-free model when the following set of conditions is satisfied:

$$\begin{cases} \mathcal{U}^{\infty}(x) = \mathcal{P} - \frac{\mathcal{C}_x}{p_x} > 0 \ , \\ \mathcal{U}^{\lambda}(x) = \left[\mathcal{P} - \frac{\mathcal{C}_x}{p_x}\right] \cdot \left[1 - (1 - p_x)^{\left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor}\right] \leqslant 0 \ . \end{cases} \tag{5.4}$$

As $1 - (1 - p_x)^{\left\lfloor \frac{\lambda}{\mathcal{C}_x} \right\rfloor}$ is non-negative series which grows from 0 towards 1, it can be seen that the condition (5.4) can be satisfied only in the case when the adversary has no resources to attack – when $\lambda < \mathcal{C}_x$. $\square$

Thus limiting adversarial budget does not provide more trustworthy nor more reliable results compared to the improved failure-free model in the case of a single attack. If in the case of some positive budget λ the adversarial utility is positive, it can be less or equal to zero in the model with budget limitations iff $\lambda < \mathcal{C}_x$. In other words, if the system is insecure in the improved failure-free model it will also be insecure in the model with budget limitations for any adversarial budget sufficient to launch the attack at least once.

Two Attacks

Let us now consider the case when the adversary can launch two attacks $x_\imath$ and $x_\jmath$ annotated with corresponding costs $\mathcal{C}_\imath$ and $\mathcal{C}_\jmath$ and success probabilities $p_{x_\imath}$ and $p_{x_\jmath}$. The adversarial utility changes in the so-called lattice points which are the projections of points $(n\mathcal{C}_{x_\imath}, m\mathcal{C}_{x_\jmath})$ in a two-dimensional Euclidean space into one-dimensional space using the formula $\mathcal{L}_\imath = n\mathcal{C}_{x_\imath} + m\mathcal{C}_{x_\jmath}$, where:

$$n \in \left\{1, 2, \ldots, \left\lfloor \frac{\lambda}{\mathcal{C}_{x_\imath}} \right\rfloor \right\}, \; m = \left\{1, 2, \ldots, \left\lfloor \frac{\lambda}{\mathcal{C}_{x_\jmath}} \right\rfloor \right\}, \; \forall \imath : \; \mathcal{L}_\imath \leqslant \lambda \ ,$$

as shown in Fig. 5.4.

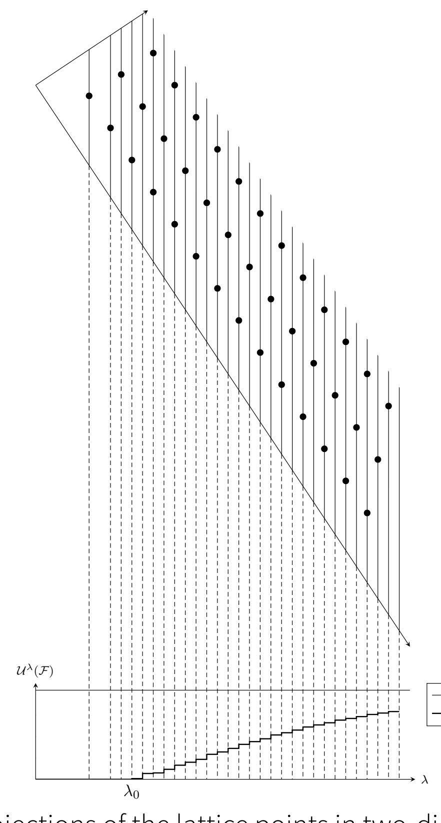

$\mathcal{U}^\lambda(\mathcal{F})$
$\mathcal{U}^\infty(\mathcal{F})$
$\mathcal{U}^\lambda(\mathcal{F})$
λ_0
λ

Figure 5.4: Projections of the lattice points in two-dimensional space into one-dimensional space

In the case of the three elementary attacks $x_\imath, x_\jmath, x_k$ annotated with corresponding costs $\mathcal{C}_\imath, \mathcal{C}_\jmath, \mathcal{C}_k$ and success probabilities $p_{x_\imath}, p_{x_\jmath}, p_k$, the adversarial utility changes in the projections of points $(n\mathcal{C}_{x_\imath}, m\mathcal{C}_{x_\jmath}, r\mathcal{C}_{x_k})$ in a three-dimensional space into one-dimensional space. Thus with the increase in the number of possible attacks the lattice argument space becomes more complex. It can be shown that the distance between the two adjacent lattice points has a lower bound.

Theorem 5.2.2. *If the ratio of attack costs $\mathcal{C}_{x_\imath}$ and $\mathcal{C}_{x_\jmath}$ is a rational number, i.e. $\frac{\mathcal{C}_{x_\imath}}{\mathcal{C}_{x_\jmath}} = \frac{p}{q}$, where* $\gcd(p, q) = 1$*, then the distance between the two adjacent lattice points $\mathcal{L}_\imath$ and $\mathcal{L}_{\imath+1}$ will be not less than $\frac{\mathcal{C}_{x_\jmath}}{q}$.*

Proof. The distance δ between the two adjacent lattice points $\mathcal{L}_\imath$ and $\mathcal{L}_{\imath+1}$ may be expressed as

$$\begin{aligned}\delta &= |(n - n') \cdot \mathcal{C}_{x_\imath} + (m - m') \cdot \mathcal{C}_{x_\jmath}| \\ &= |(n - n') \cdot p + (m - m') \cdot q| \cdot \frac{\mathcal{C}_{x_\jmath}}{q} \\ &= \begin{cases} 0, & \text{if } \alpha = 0 \ , \\ \geqslant \frac{\mathcal{C}_{x_\jmath}}{q}, & \text{if } \alpha \neq 0 \ . \end{cases}\end{aligned}$$

□

If the ratio of the costs of attacks is irrational, lattice points appear with increasing frequency eventually positioning infinitely close to each other. In the real life we can expect the costs to be rational (it would be non-trivial for an analyst to estimate an irrational value for the cost annotation) and for this reason the above mentioned bound exists in the practical cases.

5.3 Elementary Disjunctive Game

In the case of an elementary disjunctive game in order to win it is sufficient that any of the two attacks $\{x_\imath, x_\jmath\}$ succeeds. The initial state of the game is denoted as $\langle x_\imath \vee x_\jmath, \lambda\rangle$ and the subset of available attacks to launch (assuming that $\lambda \geqslant \mathcal{C}_{x_\imath}$ and $\lambda \geqslant \mathcal{C}_{x_\jmath}$) is $\mathcal{X} = \{x_\imath, x_\jmath\}$. In every state of the game the adversary may choose any attack from the subset of available attacks and launch it or to discontinue playing. If the adversary launches attack $x_k \in \mathcal{X}$ and it succeeds the game transits into the state $\langle \mathbf{1}, \lambda - \mathcal{C}_k\rangle$, where $\mathcal{C}_k$ is the cost of the launched attack, and the adversary has won the game. If x_k failed the game transits into the state $\langle x_\imath \vee x_\jmath, \lambda - \mathcal{C}_k\rangle$ and the game goes on while $\mathcal{C}_k \leqslant \lambda$. At some point the current λ will reduce the set of possible attacks to just one single (cheapest) attack $x_r = \mathcal{X} \setminus x_k$ and eventually the set of possible attacks becomes an empty set. Upon reaching the game state in which $\mathcal{C}_k > \lambda$ and if by this time the adversary failed to satisfy the Boolean function of the game and the adversary has no valid moves left (the remaining λ is insufficient to launch even a single attack once) – the adversary has lost the game. The adversarial utility may be expressed in the form of the recurrent relation (5.5):

$$\mathcal{U}^\lambda(x_\imath \vee x_\jmath) = \max \begin{cases} 0, \\ \mathcal{U}(x_\imath) + (1 - p_{x_\imath}) \cdot \mathcal{U}^{\lambda - \mathcal{C}_{x_\imath}}(x_\imath \vee x_\jmath) \ , \\ \mathcal{U}(x_\jmath) + (1 - p_{x_\jmath}) \cdot \mathcal{U}^{\lambda - \mathcal{C}_{x_\jmath}}(x_\imath \vee x_\jmath) \ . \end{cases} \tag{5.5}$$

In certain cases under certain conditions the optimal strategy in the elementary disjunctive game is non-adaptive and suggests to repeat one of the attacks independently of the current state of the game. Let us bring an example of such a case.

Theorem 5.3.1. *If the costs of the attacks are equal the attack with greater success probability will be the best choice in every state of the game.*

Proof. Assume that $\mathcal{C}_{x_\imath} = \mathcal{C}_{x_\jmath} = \mathcal{C}$. Given this assumption the utility of the game may be expressed in the form of equation (5.6).

$$\mathcal{U}^\lambda(x_\imath \vee x_\jmath) = \max \begin{cases} 0, \\ \mathcal{U}(x_\imath) + (1 - p_{x_\imath}) \cdot \mathcal{U}^{\lambda - \mathcal{C}}(x_\imath \vee x_\jmath) \ , \\ \mathcal{U}(x_\jmath) + (1 - p_{x_\jmath}) \cdot \mathcal{U}^{\lambda - \mathcal{C}}(x_\imath \vee x_\jmath) \ . \end{cases} \tag{5.6}$$

The attack $x_\imath$ will be the best choice in every state of the game if

$$\mathcal{U}(x_\imath) + (1 - p_{x_\imath}) \cdot \mathcal{U}^{\lambda - \mathcal{C}}(x_\imath \vee x_\jmath) > \mathcal{U}(x_\jmath) + (1 - p_{x_\jmath}) \cdot \mathcal{U}^{\lambda - \mathcal{C}}(x_\imath \vee x_\jmath) \ . \tag{5.7}$$

Solving inequality (5.7) we reach condition $p_{x_\imath} > p_{x_\jmath}$. □

Algorithm 5.3.1 outlines the recursive procedure to calculate maximal adversarial utility in the elementary disjunctive game given budget λ according to (5.5).

ALGORITHM 5.3.1: Adversarial utility in the elementary disjunctive game with given budget

```
   Input: Attack $x_\imath$ cost $\mathcal{C}_\imath$
   Input: Attack $x_\imath$ probability $p_\imath$
   Input: Attack $x_\jmath$ cost $\mathcal{C}_\jmath$
   Input: Attack $x_\jmath$ probability $p_\jmath$
   Input: Prize of the game $\mathcal{P}$
   Input: Budget $\lambda$
   Output: Adversarial utility (a real number)
1  Function DisjunctiveUtility ($\mathcal{C}_\imath, p_\imath, \mathcal{C}_\jmath, p_\jmath, \mathcal{P}, \lambda$)
2  if $\lambda$ is less than $\mathcal{C}_\imath$ and $\mathcal{C}_\jmath$ then
3      return (0)
4  $\mathcal{U}(x_\imath)$ := -$\mathcal{C}_\imath + p_\imath \cdot \mathcal{P}$
5  $\mathcal{U}(x_\jmath)$ := -$\mathcal{C}_\jmath + p_\jmath \cdot \mathcal{P}$
6  if $\lambda$ is greater than $\mathcal{C}_\imath$ then
7      $\mathcal{U}_\imath = \mathcal{U}(x_\imath) + (1 - p_\imath)\cdot$ DisjunctiveUtility ($\mathcal{C}_\imath, p_\imath, \mathcal{C}_\jmath, p_\jmath, \mathcal{P}, \lambda - \mathcal{C}_\imath$)
8  if $\lambda$ is greater than $\mathcal{C}_\jmath$ then
9      $\mathcal{U}_\jmath = \mathcal{U}(x_\jmath) + (1 - p_\jmath)\cdot$ DisjunctiveUtility ($\mathcal{C}_\imath, p_\imath, \mathcal{C}_\jmath, p_\jmath, \mathcal{P}, \lambda - \mathcal{C}_\jmath$)
10 return max $\left\{ 0, \mathcal{U}_\imath, \mathcal{U}_\jmath \right\}$
```

5.3.1 Optimal Strategies in Disjunctive Games

It is best to demonstrate the variety of optimal strategies in elementary disjunctive games and in particular the dynamics of the best move changing depending on the current budget λ by several examples. Let us start with an example which demonstrates the case when the best move bounces between the two attacks $x_\imath$ and $x_\jmath$ in the case when the budget is rather small and sticks to one single attack later on thus forming a clear adversarial preference of one attack over another (Fig. 5.5). The symbol $\emptyset$ denotes the case when the best possible choice of the adversary would be not to start playing the game.

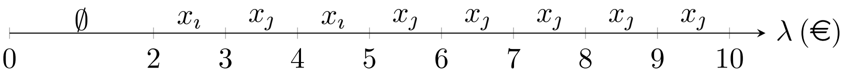


Figure 5.5: Optimal strategy in elementary disjunctive game with with quantitative annotations:
$\mathcal{C}_{x_\imath} = 2€, p_{x_\imath} = 0.3, \mathcal{C}_{x_\jmath} = 3€, p_{x_\jmath} = 0.48, \mathcal{P} = 30€$

The next example (Fig. 5.6) demonstrates the case when both of the attacks are equally good for the adversary while the budget is rather small and thus there is no difference for the adversary whether to launch attack $x_\imath$ or $x_\jmath$. However when the budget increases the adversary has a clear distinguishable preference for one attack over the other. The case when the adversary has no clear preference of one attack over another is denoted by the $=$ symbol.

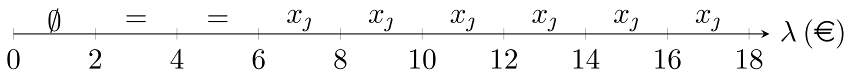


Figure 5.6: Optimal strategy in elementary disjunctive game with with quantitative annotations:
$\mathcal{C}_{x_\imath} = 2€, p_{x_\imath} = 0.05, \mathcal{C}_{x_\jmath} = 6€, p_{x_\jmath} = 0.9, \mathcal{P} = 30€$

The next example (Fig. 5.7) demonstrates the case when the costs of the attacks are irrational but their ratio is rational. It can be seen that the best move to undertake in a certain state of the game bounces between attacks $x_\imath$ and $x_\jmath$.

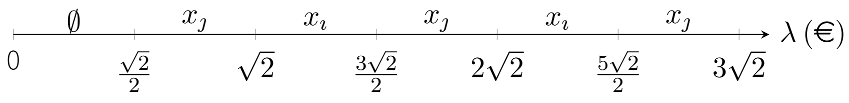


Figure 5.7: Optimal strategy in elementary disjunctive game with with quantitative annotations:
$\mathcal{C}_{x_\imath} = \sqrt{2}€, p_{x_\imath} = 0.8, \mathcal{C}_{x_\jmath} = \frac{\sqrt{2}}{2}€, p_{x_\imath} = 0.45, \mathcal{P} = 30€$

The last example (Fig. 5.8) demonstrates that there are cases when the optimal strategy is non-adaptive and iterates just one single attack, e.g. $x_\jmath$ disregarding the current state of the game.

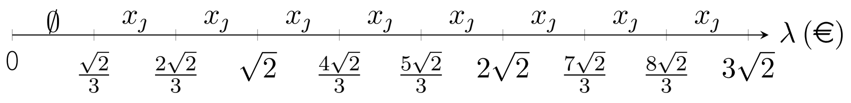


Figure 5.8: Optimal strategy in elementary disjunctive game with with quantitative annotations:
$\mathcal{C}_{x_\imath} = \sqrt{2}, p_{x_\imath} = 0.1, \mathcal{C}_{x_\jmath} = \frac{\sqrt{2}}{3}€, p_{x_\jmath} = 0.38, \mathcal{P} = 30€$

5.3.2 Comparison with the Improved Failure-Free Model

It can be shown that the case when the improved failure-free model analysis concludes that the analyzed enterprise is insecure while the budgeted model result states that the considered enterprise is secure – is impossible.

Theorem 5.3.2. *If an elementary disjunctive game is unprofitable in the improved failure-free model with limited budget, it will be unprofitable in the improved failure-free model with unlimited budget.*

Proof. Let us consider adversarial budget $\mathcal{I}$ for which the following inequalities hold:

$$\begin{aligned} &\mathcal{U}^{\mathcal{I}}(x_\imath \vee x_\jmath) > 0 \ , \\ &\mathcal{U}^{\mathcal{I}-\mathcal{C}}(x_\imath \vee x_\jmath) \leqslant 0 \ , \end{aligned} \tag{5.8}$$

where $\mathcal{C}$ is the corresponding cost of any of the attacks. Assuming $\mathcal{I}$ is greater than $\mathcal{C}_{x_\imath}$ and $\mathcal{C}_{x_\jmath}$:

$$\mathcal{U}^{\mathcal{C}}(x_\imath \vee x_\jmath) \leqslant 0 \ . \tag{5.9}$$

Let x_k with cost $\mathcal{C}$ and success probability p be the best move in the considered state of the game. In this case:

$$\mathcal{U}^{\mathcal{I}}(x_\imath \vee x_\jmath) = \mathcal{U}^{\mathcal{C}}(x_\imath \vee x_\jmath) + (1-p) \cdot \mathcal{U}^{\mathcal{I}-\mathcal{C}}(x_\imath \vee x_\jmath) \ .$$

As $\mathcal{U}^{\mathcal{I}-\mathcal{C}}(x_\imath \vee x_\jmath) \leqslant 0$ by (5.8) and $\mathcal{U}^{\mathcal{C}}(x_\imath \vee x_\jmath) \leqslant 0$ by (5.9) it contradicts with the initial assumption $\mathcal{U}^{\mathcal{I}}(x_\imath \vee x_\jmath) > 0$. ☐

Thus it seems that there is no point in limiting adversarial budget in the elementary disjunctive games.

5.4 Elementary Conjunctive Game

In the case of an elementary conjunctive game in the initial state of the game the adversary may choose either to launch attack $x_\imath$ or to launch $x_\jmath$ or not to start playing. If the adversary has chosen to launch attack $x_\imath$ and it has failed, the game transits into the state $\langle x_\imath \wedge x_\jmath, \lambda - \mathcal{C}_{x_\imath}\rangle$. If $x_\imath$ succeeded, the game transits into the state $(x_\imath \wedge x_\jmath)|_{x_\imath=1}, \lambda - \mathcal{C}_{x_\imath}$ which is identical to $x_\jmath, \lambda - \mathcal{C}_{x_\imath}$. In this state the adversary faces the following choices: either to launch the remaining attack $x_\jmath$ (while λ is sufficient to launch it) or to discontinue playing the game. If $x_\jmath$ succeeds, the game transits into the state $\langle \mathbf{1}, \lambda - \mathcal{C}_{x_\imath} - \mathcal{C}_{x_\jmath} \rangle$ and thus the adversary has won the game. In case $x_\jmath$ fails, the game continues until λ becomes insufficient to continue playing. Following the discussion above, the adversarial utility may be expressed in the form of the relation (5.10):

$$\mathcal{U}^\lambda(x_\imath \wedge x_\jmath) = \max \begin{cases} 0, \\ -\mathcal{C}_{x_\imath} + p_{x_\imath}\mathcal{U}^{\lambda-\mathcal{C}_{x_\imath}}(x_\jmath) + (1-p_{x_\imath}) \cdot \mathcal{U}^{\lambda-\mathcal{C}_{x_\imath}}(x_\imath \wedge x_\jmath) \ , \\ -\mathcal{C}_{x_\jmath} + p_{x_\jmath}\mathcal{U}^{\lambda-\mathcal{C}_{x_\jmath}}(x_\imath) + (1-p_{x_\jmath}) \cdot \mathcal{U}^{\lambda-\mathcal{C}_{x_\jmath}}(x_\imath \wedge x_\jmath) \ . \end{cases} \tag{5.10}$$

In the elementary conjunctive games the positive utility may not be achieved immediately by the adversary, as shown in Fig. 5.9. Let us call the minimal value of the adversarial budget, sufficient to generate positive utility, the *lower bound of adversarial budget* and denote it by λ_0.

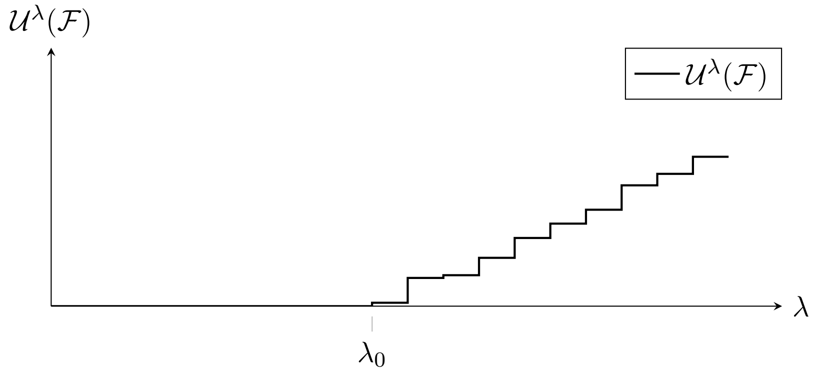


Figure 5.9: Lower bound of adversarial budget sufficient to generate positive utility

Considering that we are looking for λ in the case of which the adversarial utility makes an initial step climb from zero towards some positive value we can disregard the previous state of the game, as the utility in this state is zero by definition and thus the equation (5.10) may be re-written as follows:

$$\mathcal{U}^\lambda(x_\imath \wedge x_\jmath) = \max \begin{cases} 0, \\ -\mathcal{C}_{x_\imath} + p_{x_\imath}\mathcal{U}^{\lambda-\mathcal{C}_{x_\imath}}(x_\jmath) + \underbrace{(1-p_{x_\imath}) \cdot \mathcal{U}^{\lambda-\mathcal{C}_{x_\imath}}(x_\imath \wedge x_\jmath)}_{0} \ , \\ -\mathcal{C}_{x_\jmath} + p_{x_\jmath}\mathcal{U}^{\lambda-\mathcal{C}_{x_\jmath}}(x_\imath) + \underbrace{(1-p_{x_\jmath}) \cdot \mathcal{U}^{\lambda-\mathcal{C}_{x_\jmath}}(x_\imath \wedge x_\jmath)}_{0} \ . \end{cases}$$

$$= \max \begin{cases} 0, \\ -\mathcal{C}_{x_\imath} + p_{x_\imath}\mathcal{U}^{\lambda-\mathcal{C}_{x_\imath}}(x_\jmath) \ , \\ -\mathcal{C}_{x_\jmath} + p_{x_\jmath}\mathcal{U}^{\lambda-\mathcal{C}_{x_\jmath}}(x_\imath) \ . \end{cases} \tag{5.11}$$

The equation (5.3) in Section 5.2 states that:

$$\mathcal{U}^{\lambda}(x_\imath) = \mathcal{U}^{\infty}(x_\imath) \cdot \left[1 - (1-p_\imath)^{\left\lfloor \frac{\lambda}{\mathcal{C}_{x_\imath}} \right\rfloor}\right] \quad ,$$

and therefore:

$$\mathcal{U}^{\lambda - \mathcal{C}_{x_\imath}}(x_\jmath) = \mathcal{U}^{\infty}(x_\jmath) \cdot \left[1 - (1-p_{x_\jmath})^{\left\lfloor \frac{\lambda - \mathcal{C}_{x_\imath}}{\mathcal{C}_{x_\jmath}} \right\rfloor}\right] \quad ,$$

$$\mathcal{U}^{\lambda - \mathcal{C}_{x_\jmath}}(x_\imath) = \mathcal{U}^{\infty}(x_\imath) \cdot \left[1 - (1-p_{x_\imath})^{\left\lfloor \frac{\lambda - \mathcal{C}_{x_\jmath}}{\mathcal{C}_{x_\imath}} \right\rfloor}\right] \quad .$$

Now it is possible to re-write equation (5.11) in the following way:

$$\mathcal{U}^{\lambda}(x_\imath \wedge x_\jmath) = \max \begin{cases} 0, \\ -\mathcal{C}_{x_\imath} + p_{x_\imath} \cdot \left[\mathcal{U}^{\infty}(x_\jmath)\right] \cdot \left[1 - (1-p_{x_\jmath})^{\left\lfloor \frac{\lambda - \mathcal{C}_{x_\imath}}{\mathcal{C}_{x_\jmath}} \right\rfloor}\right] \quad , \\ -\mathcal{C}_{x_\jmath} + p_{x_\jmath} \cdot \left[\mathcal{U}^{\infty}(x_\imath)\right] \cdot \left[1 - (1-p_{x_\imath})^{\left\lfloor \frac{\lambda - \mathcal{C}_{x_\jmath}}{\mathcal{C}_{x_\imath}} \right\rfloor}\right] \quad . \end{cases}$$

We are settled to find λ in the case of which $\mathcal{U}^{\lambda}(x_\imath \wedge x_\jmath)$ is positive – therefore the following system of inequalities must hold:

$$-\mathcal{C}_{x_\imath} + p_{x_\imath} \cdot \mathcal{U}^{\infty}(x_\jmath) \cdot \left[1 - (1-p_{x_\jmath})^{\left\lfloor \frac{\lambda - \mathcal{C}_{x_\imath}}{\mathcal{C}_{x_\jmath}} \right\rfloor}\right] > 0$$

$$-\mathcal{C}_{x_\jmath} + p_{x_\jmath} \cdot \mathcal{U}^{\infty}(x_\imath) \cdot \left[1 - (1-p_{x_\imath})^{\left\lfloor \frac{\lambda - \mathcal{C}_{x_\jmath}}{\mathcal{C}_{x_\imath}} \right\rfloor}\right] > 0$$

Solving the inequalities for λ we get:

$$\frac{\lambda - \mathcal{C}_{x_\imath}}{\mathcal{C}_{x_\jmath}} < \log_{(1-p_{x_\jmath})} \left[1 - \frac{\mathcal{C}_{x_\imath}}{p_{x_\imath} \cdot \mathcal{U}^{\infty}(x_\jmath)}\right]$$

$$\frac{\lambda - \mathcal{C}_{x_\jmath}}{\mathcal{C}_{x_\imath}} < \log_{(1-p_{x_\imath})} \left[1 - \frac{\mathcal{C}_{x_\jmath}}{p_{x_\jmath} \cdot \mathcal{U}^{\infty}(x_\imath)}\right]$$

And therefore:

$$\lambda_0 = \min \begin{cases} \left\lfloor \log_{(1-p_{x_\jmath})} \left[1 - \frac{\mathcal{C}_{x_\imath}}{p_{x_\imath} \cdot \mathcal{U}^{\infty}(x_\jmath)}\right] \right\rfloor \cdot \mathcal{C}_{x_\jmath} + \mathcal{C}_{x_\imath} \quad , \\ \left\lfloor \log_{(1-p_{x_\imath})} \left[1 - \frac{\mathcal{C}_{x_\jmath}}{p_{x_\jmath} \cdot \mathcal{U}^{\infty}(x_\imath)}\right] \right\rfloor \cdot \mathcal{X}_{x_\imath} + \mathcal{C}_{x_\jmath} \quad . \end{cases}$$

Algorithm 5.4.1 outlines the recursive procedure to calculate adversarial utility in the elementary conjunctive game given budget λ according to (5.10).

ALGORITHM 5.4.1: Adversarial utility in the elementary conjunctive game with the given budget

```
  Input: Attack $x_\imath$ cost $\mathcal{C}_{x_\imath}$
  Input: Attack $x_\imath$ probability $x_\imath$
  Input: Attack $x_\jmath$ cost $\mathcal{C}_{x_\jmath}$
  Input: Attack $x_\jmath$ probability $x_\jmath$
  Input: Prize of the game $\mathcal{P}$
  Input: Budget $\lambda$
  Output: Adversarial utility (a real number)
1 Function ConjunctiveUtility $(\mathcal{C}_{x_\imath}, p_\imath, \mathcal{C}_{x_\jmath}, p_\jmath, \mathcal{P}, \lambda)$
2 if $\lambda < \mathcal{C}_{x_\imath} + \mathcal{C}_{x_\jmath}$ then
3     return (0)
4 aux$_\imath := \left[\mathcal{P} - \frac{\mathcal{C}_{x_\imath}}{p_\imath}\right] \cdot \left[1 - (1 - p_\jmath)^{\left\lfloor \frac{\lambda - \mathcal{C}_{x_\imath}}{\mathcal{C}_{x_\jmath}} \right\rfloor}\right]$
5 aux$_\jmath := \left[\mathcal{P} - \frac{\mathcal{C}_{x_\jmath}}{p_\jmath}\right] \cdot \left[1 - (1 - p_\imath)^{\left\lfloor \frac{\lambda - \mathcal{C}_{x_\jmath}}{\mathcal{C}_{x_\imath}} \right\rfloor}\right]$
6 $\mathcal{U}_\imath = -\mathcal{C}_{x_\imath} + p_\imath \cdot$ aux$_\jmath + (1 - p_\imath) \cdot$ ConjunctiveUtility
  $(\mathcal{C}_{x_\imath}, p_\imath, \mathcal{C}_{x_\jmath}, p_\jmath, \mathcal{P}, \lambda - \mathcal{C}_{x_\imath})$
7 $\mathcal{U}_\jmath = -\mathcal{C}_{x_\jmath} + p_\jmath \cdot$ aux$_\imath + (1 - p_\jmath) \cdot$ ConjunctiveUtility
  $(\mathcal{C}_{x_\imath}, p_\imath, \mathcal{C}_{x_\jmath}, p_\jmath, \mathcal{P}, \lambda - \mathcal{C}_{x_\jmath})$
8 return max $\left\{ 0, \mathcal{U}_\imath, \mathcal{U}_\jmath \right\}$
```

5.4.1 Optimal Strategies in Conjunctive Games

It is best to demonstrate the variety of optimal strategies in elementary conjunctive games and in particular the dynamics of the best move changing depending on the current budget λ by several examples. Let us start with an example which shows that there are certain sets of quantitative annotations on the attacks which render the adversary indifferent in whether to launch attack $x_\imath$ or $x_\jmath$ in every state of the game (Fig. 5.10).

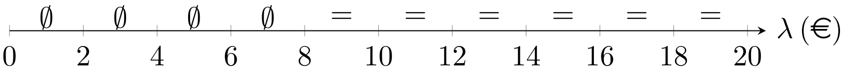


Figure 5.10: Optimal strategy in elementary conjunctive game with quantitative annotations $\mathcal{C}_{x_\imath} = 2€, p_{x_\imath} = 0.05, \mathcal{C}_{x_\jmath} = 6€, p_{x_\jmath} = 0.9, \mathcal{P} = 30€$

The second example (Fig. 5.11) demonstrates the case when the best move bounces between attacks $x_\imath$ and $x_\jmath$. In some states of the game both of the attacks are equally optimal to try.

The next example (Fig. 5.12) demonstrates the case when the costs of corresponding attacks are irrational, but their relation may be expressed in terms of a fraction

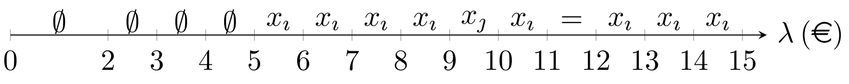

∅ ∅ ∅ ∅ $x_\imath$ $x_\imath$ $x_\imath$ $x_\imath$ $x_\jmath$ $x_\imath$ = $x_\imath$ $x_\imath$ $x_\imath$
λ (€)
0 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Figure 5.11: Optimal strategy in elementary conjunctive game with quantitative annotations $\mathcal{C}_{x_\imath} = 2€, p_{x_\imath} = 0.3, \mathcal{C}_{x_\jmath} = 3€, p_{x_\jmath} = 0.48, \mathcal{P} = 30€$

of rational numbers. It can be seen that with given quantitative annotations the optimal strategy is non-adaptive and will suggest to iterate attack $x_\jmath$.

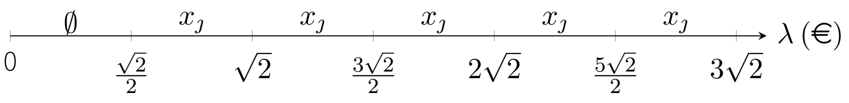

∅ $x_\jmath$ $x_\jmath$ $x_\jmath$ $x_\jmath$ $x_\jmath$
λ (€)
0 $\frac{\sqrt{2}}{2}$ $\sqrt{2}$ $\frac{3\sqrt{2}}{2}$ $2\sqrt{2}$ $\frac{5\sqrt{2}}{2}$ $3\sqrt{2}$

Figure 5.12: Optimal strategy in elementary conjunctive game with quantitative annotations
$\mathcal{C}_{x_\imath} = \sqrt{2}€, p_{x_\imath} = 0.8, \mathcal{C}_{x_\jmath} = \frac{\sqrt{2}}{2}€, p_{x_\jmath} = 0.45, \mathcal{P} = 30€$

The last example (Fig. 5.13) demonstrates the case when the optimal strategy is fully-adaptive and the best move to undertake is $x_\imath$ or $x_\jmath$ depending on the state of the game.

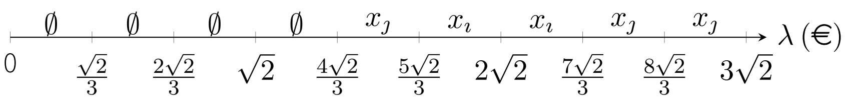

∅ ∅ ∅ ∅ $x_\jmath$ $x_\imath$ $x_\imath$ $x_\jmath$ $x_\jmath$
λ (€)
0 $\frac{\sqrt{2}}{3}$ $\frac{2\sqrt{2}}{3}$ $\sqrt{2}$ $\frac{4\sqrt{2}}{3}$ $\frac{5\sqrt{2}}{3}$ $2\sqrt{2}$ $\frac{7\sqrt{2}}{3}$ $\frac{8\sqrt{2}}{3}$ $3\sqrt{2}$

Figure 5.13: Optimal strategy in elementary conjunctive game with quantitative annotations
$\mathcal{C}_{x_\imath} = \sqrt{2}€, p_{x_\imath} = 0.1, \mathcal{C}_{p_{x_\jmath}} = \frac{\sqrt{2}}{3}€, p_{p_{x_\jmath}} = 0.38, \mathcal{P} = 30€$

5.4.2 Comparison with the Improved Failure-Free Model

Similarly to the elementary disjunctive game it is desirable to investigate if the computational methods assuming limited adversarial budget can produce results different from the result in the improved failure-free model. A different result would be produced in the case when, for instance, the improved failure-free model analysis result states that the considered enterprise is insecure while the results of analysis assuming limited adversarial budget states that the analyzed enterprise in secure.

Assume the case when in order to achieve his goal an adversary has to succeed in attacks $x_\imath$ and $x_\jmath$ annotated with corresponding costs $\mathcal{C}_{x_\imath}$ and $\mathcal{C}_{x_\jmath}$ and success probabilities $p_{x_\imath}$ and $p_{x_\jmath}$. According to the improved failure-free model the upper bound of the utility is $\mathcal{U}^\infty(x_\imath \wedge x_\jmath) = \mathcal{P} - \frac{\mathcal{C}_{x_\imath}}{p_{x_\imath}} - \frac{\mathcal{C}_{x_\jmath}}{p_{x_\jmath}}$. The considered enterprise is secure if $\mathcal{P} \leqslant \frac{\mathcal{C}_{x_\imath}}{p_{x_\imath}} + \frac{\mathcal{C}_{x_\jmath}}{p_{x_\jmath}}$ and insecure if $\mathcal{P} > \frac{\mathcal{C}_{x_\imath}}{p_{x_\imath}} + \frac{\mathcal{C}_{x_\jmath}}{p_{x_\jmath}}$.

For the sake of simplicity, let us assume the case in which $\mathcal{C}_{x_\imath} = \mathcal{C}_{x_\jmath} = \mathcal{C}$ and $p_{x_\imath} = p_{x_\jmath} = p$. Let the adversarial budget λ suffice to launch m attacks ($\lambda = m \cdot \mathcal{C}$). Let us assume a strategy which suggests to try attack $x_\imath$ first while λ is sufficient to launch it or until $x_\imath$ succeeds. After $x_\imath$ succeeded if there is still budget left to launch $x_\jmath$ it is launched until success or until λ becomes insufficient and there will be no valid moves left. The adversarial decision making process in elementary conjunctive games is shown in Fig. 5.14.

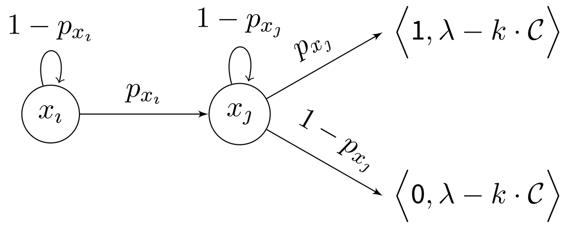

$1 - p_{x_\imath}$
$1 - p_{x_\jmath}$
$p_{x_\jmath}$
$\left\langle 1, \lambda - k \cdot \mathcal{C} \right\rangle$
$x_\imath$
$p_{x_\imath}$
$x_\jmath$
$1 - p_{x_\jmath}$
$\left\langle 0, \lambda - k \cdot \mathcal{C} \right\rangle$

Figure 5.14: Adversarial decision making process in the case of the elementary conjunctive game, where $k \in \{2, \ldots, m\}$

The adversarial utility in the model considering limited adversarial budget is:

$$\begin{aligned}
\mathcal{U}^{m \cdot \mathcal{C}}(x_\imath \wedge x_\jmath) = -\mathcal{C} &+ p \cdot \mathcal{U}^\infty(x_\jmath) \cdot \Big[1 - (1-p)^{m-1}\Big] \\
&+ (1-p) \cdot \Bigg[-\mathcal{C} + p \cdot \mathcal{U}^\infty(x_\jmath) \cdot \Big[1 - (1-p)^{m-1}\Big]\Bigg] \\
&+ (1-p)^2 \cdot \Bigg[-\mathcal{C} + p \cdot \mathcal{U}^\infty(x_\jmath) \cdot \Big[1 - (1-p)^{m-2}\Big]\Bigg] \\
&+ \ldots \\
&+ (1-p)^{m-3} \cdot \Bigg[-\mathcal{C} + p \cdot \mathcal{U}^\infty(x_\jmath) \cdot \Big[1 - (1-p)^{2}\Big]\Bigg] \\
&+ (1-p)^{m-2} \cdot \Bigg[-\mathcal{C} + p \cdot \mathcal{U}^\infty(x_\jmath) \cdot \Big[1 - (1-p)^{1}\Big]\Bigg] \ .
\end{aligned}$$

And therefore:

$$\begin{aligned}
\mathcal{U}^{m \cdot \mathcal{C}}(x_\imath \wedge x_\jmath) = \Big[-\mathcal{C} + p \cdot \mathcal{U}^\infty(x_\jmath)\Big] \cdot \Big[1 + (1-p) + \ldots + (1-p)^{m-2}\Big] \\
- (m-1)(1-p)^{m-1} \cdot \Big[p \cdot \mathcal{U}^\infty(x_\jmath)\Big] \ .
\end{aligned}$$

According to (5.2):

$$\sum_{k=0}^{m-2} (1 - p_{x_\imath})^k = \frac{1}{p} \cdot \Big[1 - (1-p)^{m-1}\Big] \ ,$$

and thus:

$$\begin{aligned}\mathcal{U}^{m\cdot\mathcal{C}}(x_\imath \wedge x_\jmath) &= \frac{1}{p}\cdot\Big[-\mathcal{C}+p\cdot\mathcal{U}^{\infty}(x_\jmath)\Big]\cdot\Big[1-(1-p)^{m-1}\Big]\\ &\quad-(m-1)(1-p)^{m-1}\cdot\Big[p\cdot\mathcal{U}^{\infty}(x_\jmath)\Big]\\ &=\left[\mathcal{P}-\frac{2\mathcal{C}}{p}\right]\cdot\Big[1-(1-p)^{m-1}\Big]\\ &\quad-(m-1)(1-p)^{m-1}\cdot\Big[p\mathcal{P}-\mathcal{C}\Big]\ .\end{aligned}$$

The model considering limited adversarial budget will produce results different from the results in the improved failure-free model iff the following inequalities hold:

$$\begin{aligned}&\mathcal{P}-\frac{\mathcal{C}_{x_\imath}}{p_{x_\imath}}+\frac{\mathcal{C}_{x_\jmath}}{p_{x_\jmath}}>0\\ &\left[\mathcal{P}-\frac{2\mathcal{C}}{p}\right]\Big[1-(1-p)^{m-1}\Big]-(m-1)(1-p)^{m-1}\Big[p\mathcal{P}-\mathcal{C}\Big]\leqslant 0\ .\end{aligned}$$

Solving for $\mathcal{P}$ we get:

$$\frac{2\mathcal{C}}{p}\leqslant\mathcal{P}\leqslant\frac{2\mathcal{C}}{p}\cdot\frac{1-\Big[1+\mathcal{C}(m-1)\Big]\cdot(1-p)^{m-1}}{1-\Big[1+p(m-1)\Big]\cdot(1-p)^{m-1}}\ . \tag{5.12}$$

Inequality (5.12) shows the interval for the value of prize within which the result of the model considering limited adversarial budget and the result of the improved failure-free model differ. In a broader sense the accuracy bounds for profit are as shown in Fig. 5.15. It can be seen that if the prize is less than $\frac{2\mathcal{C}}{p}$ then the con-

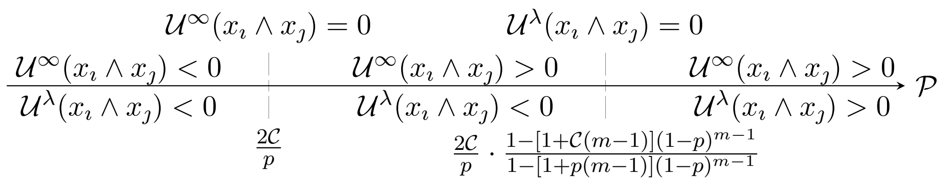

$\mathcal{U}^{\infty}(x_\imath \wedge x_\jmath)=0$
$\mathcal{U}^{\lambda}(x_\imath \wedge x_\jmath)=0$
$\mathcal{U}^{\infty}(x_\imath \wedge x_\jmath)<0$
$\mathcal{U}^{\lambda}(x_\imath \wedge x_\jmath)<0$
$\mathcal{U}^{\infty}(x_\imath \wedge x_\jmath)>0$
$\mathcal{U}^{\lambda}(x_\imath \wedge x_\jmath)<0$
$\mathcal{U}^{\infty}(x_\imath \wedge x_\jmath)>0$
$\mathcal{U}^{\lambda}(x_\imath \wedge x_\jmath)>0$
$\mathcal{P}$
$\frac{2\mathcal{C}}{p}$
$\frac{2\mathcal{C}}{p}\cdot\frac{1-[1+\mathcal{C}(m-1)](1-p)^{m-1}}{1-[1+p(m-1)](1-p)^{m-1}}$

Figure 5.15: Comparison of the improved failure-free model to the limited budget model – prize accuracy bounds

sidered enterprise is secure according to both models. If the prize is greater than $\frac{2\mathcal{C}}{p}\cdot\frac{1-[1+\mathcal{C}(m-1)](1-p)^{m-1}}{1-[1+p(m-1)](1-p)^{m-1}}$ then the system is insecure according to both models. However when the prize is between $\frac{2\mathcal{C}}{p}$ and $\frac{2\mathcal{C}}{p}\cdot\frac{1-[1+\mathcal{C}(m-1)](1-p)^{m-1}}{1-[1+p(m-1)](1-p)^{m-1}}$ the model which considers limited adversarial budget may produce results different from the results in the improved failure-free model.

Table 5.1: *Quantitative annotations: Prize: 30€ Cost: 2€ Probability: 0.3*

Lambda (#)	$\mathcal{P}$ Domain (€)	Span (€)	Deviation (€)	Precision (%)
2	$(13.333, 28.888]$	15.555	± 7.777	0.519
3	$(13.333, 22.407]$	9.074	± 4.537	0.302
4	$(13.333, 19.242]$	5.909	± 2.954	0.197
5	$(13.333, 17.405]$	4.071	± 2.036	0.136
6	$(13.333, 16.232]$	2.899	± 1.449	0.097
7	$(13.333, 15.439]$	2.105	± 1.052	0.071
8	$(13.333, 14.882]$	1.548	± 0.774	0.052
9	$(13.333, 14.481]$	1.147	± 0.574	0.038

Experiments with various possible quantitative annotations for the attacks $x_\imath$ and $x_\jmath$ have shown that the interval (5.15) becomes negligibly small – less than 1€. It is nearly impossible in practice to estimate the value of an asset with precision less than 1€. For this reason in practice the model which considers the limited adversarial budget may produce unreliable results in case the analysts were unable to estimate the prize annotation with the required precision. At the same time the improved failure-free model produces reliable upper bounds. It can be seen in Table 5.1 that already with rather small increase in budget (approximately three times greater than the corresponding costs of the attacks) the prize must be estimated with precision less than €1 in order to ensure reliability of the results. The first column in a table describes the monetary budget of the adversary. The second column describes the interval for possible prize values, the column named span shows the length of such an interval. Precision is the length of uncertainty interval divided by mean value.

5.5 Open Questions

One of the open questions is to figure out how well the most optimal strategy from the subset of non-adaptive strategies $\mathcal{U}^\lambda_{na}(\mathcal{G})$ might approximate the optimal strategy from the set of all possible strategies $\mathcal{U}^\lambda(\mathcal{G})$. If $\mathcal{U}^\lambda_{na}(\mathcal{G})$ provides pretty good approximation to $\mathcal{U}^\lambda(\mathcal{G})$, then there exists infinitely small α such that:

$$\mathcal{U}^\lambda_{na}(\mathcal{G}) \leqslant \mathcal{U}^\lambda(\mathcal{G}) \leqslant \alpha \cdot \mathcal{U}^\lambda_{na}(\mathcal{G}) \leqslant \mathcal{U}^\infty(\mathcal{G}) \ .$$

If this holds, it might enable calculation of acceptably precise result without facing the complexity and the computational overhead introduced by the precise utility calculation routines.

Secondly, it would be interesting to see when the optimal move in certain states of the game changes by bouncing between the two possible moves thus following some pattern. Additionally, to verify the hypothesis that this might happen in the theoretical case when the ratio of the costs of the corresponding attacks is irrational.

The bigger the adversarial budget λ is, the more adversarial utility approaches the utility upper bound in the improved failure-free model. Optimal strategies in the improved failure-free model are non-adaptive and do not depend on the ordering of the attacks. In the case of big λ optimal strategies are likely to behave non-adaptively as well in the limited budget model. This means that optimal move in certain states of the game is likely to bounce changing from one attack to another, but with increase in λ the optimal move remains the same. It also means that the utility of various strategies, beginning with different moves, become closer to each other with the increase in λ and there should exist arbitrarily small δ such that

$$|\mathcal{U}^{\lambda}(\mathcal{S}_{\imath}) - \mathcal{U}^{\lambda}(\mathcal{S}_{\jmath})| \leqslant \delta \enspace , \tag{5.13}$$

where $\mathcal{S}_{\imath}$ and $\mathcal{S}_{\jmath}$ are the two strategies from the set of all strategies of the game.

5.6 Conclusions

Three kinds of elementary satisfiability games have been analyzed – the single attack game, the elementary disjunctive game and the elementary conjunctive game, assuming that the adversarial budget is limited. It turned out that limiting adversarial budget makes computational procedures reasonably complex that makes it doubtful that such an approach is applicable for real-life case analysis. Additionally, in the case of elementary conjunctive game the *prize* parameter needs to be estimated quite precisely – if analysts fail to do that the analysis results will be unreliable. In practice it is very hard to estimate the cost of an asset or information with the desired precision and it is hardly reasonable to face the complexities of budget limitations and its possibility of a false positive result which might happen in the case of elementary conjunctive games.

The improved failure-free model is on the contrary less complex and provides reliable upper bounds. Due to the fact that when the move fails the player finds himself in the very same instance of the game results in the existence of non-adaptive strategies in the set of optimal strategies of the game and the ordering of the attacks in non-adaptive optimal strategies is irrelevant. In the model with budget limitations the subset of non-adaptive strategies exists in the set of all strategies, however non-adaptive optimal strategies tend to be present in some specific cases only. Non-adaptive strategies are relatively easy to derive and compute.

Chapter 6

Attacker Profiling

The security risk is comprised of the two independent components – the threat and vulnerability. Therefore it is possible to distinguish between

- the so-called *threat landscape* that contains all the human-made intentional threats which could potentially harm the considered enterprise and cause losses or damage to the protected assets
- the so-called *vulnerability landscape* formed by the vulnerabilities of the considered organization – weaknesses in any aspect of the infrastructure which enable possibilities for threat materialization and render it or some portion of it susceptible to damage and compromise.

The threat landscape is a characteristic of the environment in which the considered enterprise operates or where some product or technological solution is deployed. It is formed by the potential threats and the corresponding threat agents specific to some particular groups of people operating cross-border and sharing common ideology. It may be formed by the threats and threat agents specific to some country, region, urban area, or even be concentrated the vicinity of some specific geolocation. It is obvious that the risks of an enterprise operating in Germany would be different from the risks of the very same enterprise if it operated in Iraq. The threat agents have varying sets of properties – available resources, skills, experience, incentives, motivations, risk tolerance levels, views and expectations of the target enterprise. The threat agents may be devices, hardware or software components, organized criminal groups, individuals or even national security agencies or governments. Threat agents may be motivated by profit, fame, curiosity, beliefs, religion, patriotism, or may be just terrorists. The properties of the threat agents to some extent determine their strategic preferences and eventually their behavior. The threat landscape is constantly changing, every day, every hour, and never remains the same. The security solutions which yesterday provided optimal protection against the surrounding threats may totally fail today because the threat landscape has changed since then. It may change due to various reasons: threat agent behavior (an alteration in the resources available to the threat agents) as well as some events, both intentional and unintentional, not directly related to the threat agents such as organizational culture, strategy and vision of the enterprise, new emerging products and technology, new markets, new allies and competitors, cultural and political environment in the country, riots, strikes, conflicts, accidental tactless remarks about some target groups or interested parties, politically or religiously intolerant statements accidentally made by the representatives of an enterprise and misinterpreted by the target audience as rude unpolite, harming their religious beliefs or concerning the interests of other parties, socially or ecologically irresponsible actions of an enterprise, even some enterprise-internal events such as the election of the new CEO may trigger an immediate reaction in the surrounding threat landscape.

The vulnerability landscape is a characteristic of the considered enterprise, its physical, technical and organizational infrastructure – its employees, assets, policies and procedures, business processes, technological solutions, etc. The vulnerability infrastructure changes not so frequently as the threat infrastructure, but nevertheless it may change due to infrastructure updates (e.g. patching, component replacement, awareness training, deployment of security measures and procedures, etc.) as well as unintentional events like human errors. Even if no patching is undertaken or no infrastructure components are replaced, the vulnerability landscape still changes on its own with time, as the technology solutions become older, new weaknesses and zero-day vulnerabilities are discovered, potential threat agents collect knowledge about the internal processes of the organization and thus gain some insider knowledge which they might use for their advantage. If the infrastructure remains the same for considerably long time the employees in the lack of regular checks and security audits become careless and ignorant towards the established rules and procedures and thus facilitate the emergence of new vulnerabilities.

Providing meaningful metrics for operational security risk is hard [38]. There are different reasons for this: ever-changing threat landscape, difficulties in validation and updating of risk estimations, inability of performing comprehensive quantification of the threat environment,etc. Quantitative security analysis relies on quantitative annotations assigned to a single attack steps in a complex multi-step attack, such as likelihood of success of an attack, time required to prepare and launch an attack, and the difficulty of executing an attack. When the threat environment changes these annotations have to be updated as well. However the quantitative metrics of these annotations is jointly influenced by various sets of underlying components in both threat- as well as vulnerability landscapes. Thus it is rather difficult to provide a trustworthy and reliable quantitative estimations for such annotations as it is practically impossible to estimate the cumulative effect of several underlying factors altogether. In practice such kind of estimations tend to be imprecise and contain a reasonable degree of uncertainty.

For example, it is almost impossible to provide a meaningful estimation for the time annotation, as the time required to prepare and launch an attack depends on specific properties of the threat agent like the attacker skill level, capabilities, available resources, experience, as well as it depends on some infrastructure properties such as the difficulty of executing the considered attack or the related component strength. Likewise, the likelihood of success depends on attacker skill, difficulty of executing an attack and time invested into attacking. The more skillful and experienced the attacker is, the more resources are available to the attacker – the more likely he is to succeed in the considered attack. Similar reasoning may be applied to the skill parameter – the more skillful and experienced the attacker is the less difficult is the attack process for him, the less time it will take to succeed in an attack. Less skilled attackers given sufficient amount of time may be as efficient in terms of the success likelihood as more skilled attackers who have been given less time for executing the same attack. Similar logic may be extended to other quantitative annotations as well.

Despite that the analysis has to deal somehow with the ever changing nature of the threat and vulnerability landscapes the analysts have to update or re-assess the estimations of the corresponding quantitative annotations in a timely manner, which is a realistic scenario in dynamic organizations. It is unclear how to update such joint estimations when some of its underlying components change while the others remain unchanged or on the contrary when all the underlying components change. For example, the analysts may have assigned a 0.2 success likelihood assuming a malicious individual as attacker. How should this value be updated if one faces a national security agency instead and the infrastructure of the considered enterprise has been patched in the meantime and the employees have undergone an awareness training?

Attacker profiling [28] is the concept of separation of the infrastructure properties from the properties of the threat agents and may be seen as a step forward in dealing with the challenges and complexity of operational security risk metrics by analyzing multi-step attacks in the context of complex socio-technical systems. Such a separation is justified by the fact the the threat landscape is independent of the vulnerability landscape and reducing any factor of risk – either reducing potential losses or reducing the probability of vulnerability exploitation – we are reducing the risk. The separation of the infrastructure properties from the properties of the threat agents adds flexibility to quantitative security analysis enabling the assessment of operational security risks using different combinations of attacker profiles and infrastructure properties resulting in comprehensive risk analysis and providing much deeper insight on the surrounding risk landscape. If either threat or vulnerability landscape changes, this can be efficiently reflected by either updating the corresponding attacker profile(s) or updating the properties of the considered infrastructure without the need to update the joint estimations like time, cost, or success likelihood. Attacker profiles that are independent of the properties of the target infrastructure become useful re-usable components for analyzing risks. The considered organization may be analyzed to compare risks in the case of varying attacker profiles. Likewise, the same attacker profiles may be used for analyzing different enterprises. This leads to the idea that the attacker profiles that are potentially shareable with the others and thus forming the shared library of attacker profiles. The attacker profiles from the attacker profile library can be effectively re-used thus reducing the modeling effort and workload of the analysts. Besides, it turns out that attacker profiling increases the reliability of the analysis results and the separation of infrastructure properties from the properties of potential attackers allows to update these values in a timely manner independently from each other and reflect the ever changing nature of the risk landscape in security analysis in a more reliable way.

The attacker profiling concept is not new. Back in 1998, Philips *et al.* [27] outlined the importance of the attacker characteristics in attack graphs for network vulnerability analysis. Several research projects have focused on attacker profiling using honeypot in "Know Your Enemies" series [18, 19, 20] which outlined the range of techniques and tools that were used by attackers for reconnaissance and also shed light on the motivations of the blackhat community. Several researchers proposed the concept of attacker personas which was related to goal, motivations, attitudes, and skills [3, 7, 8, 9]. Faily *et al.* highlighted insider threat motivations and properties, as well as the use of attacker personas for threat identification. Pardue *et al.* [26] mentioned the importance of attacker properties and also the complexity of the attacks assessing risks of an e-voting system. The authors argue that the likelihood of attacks can be referred to as cost of an attacker, which can be estimated on various scales and measured in various units, such as dollars, number of attackers, time invested into attacking, and effort. In addition, Sallhammar *et al.* [30] demonstrate the process of deriving the probability of the expected attacker behavior in assumption that the attacker has complete information about the vulnerabilities of the target infrastructure. Tipton *et al.* [37] argue that risk aversion, degree of difficulty, discoverability, ease of access, effectiveness of controls, effort, incentive, interest, skill level, motivation, resources required, risk of detection, and special equipment needed are the factors that can be included in attacker profiling. There are some common parameters that are most often used in research projects to define an attacker profile – these values are more feasible for quantitative analysis and give clear understanding of the related attacker properties. In the scope of atomic threats some standards already acknowledge the distinction between the properties of attackers and properties of the target infrastructure. For example, the Factor Analysis of Information Security Risk (FAIR) taxonomy distinguishes between the *threat capability* and *control strength* to determine the likelihood of success. However, FAIR does not consider changing conditions in environments with multi-step attacks.

6.1 Attacker Profiles

There are two possible approaches to applying adversarial profiling to quantitative security risk analysis based on attack trees – the so-called constraint based approach and the approach which uses Item Response theory [29] to derive estimations for such annotations like success likelihood from the values of underlying parameters which jointly influence the considered annotations. Both of the approaches will be outlined in this section. First let us introduce some definitions first.

Definition (Attacker Profile). *An attacker profile $t = (t_1, \dots, t_n)$ is an n-tuple of adversarial properties t_k, each of which belongs to an ordered domain T_k.*

An attacker profile expresses adversarial capabilities and limitations.

Definition (Attack profile). *An attack profile $s = (s_1, \dots, s_m)$ is an m-tuple of attack properties $s_\imath$, each of which belongs to an ordered domain $S_\imath$.*

An attack profile expresses the resources required for attacking. An attack profile is an attack-specific characteristic and is part of the quantitative annotations on the leaves in an attack tree. In example, one may consider an attack profile $s = (d, t)$, where d is the component strength or in other words difficulty of attacking the considered component, while t is the minimal time required to successfully attack the considered component.

Definition (Attacking profile). *An attacking profile $p = (f_1, \dots, f_k)$ is a k-tuple of functions where every function $f_\imath(t, s)$, given as input a pair of an attacker profile t and attack profile s, returns a value from an ordered domain $L_\jmath$.*

Attacking profile characterizes the ability of attackers described by the attacker profile to execute an attack of the given profile. The examples of such characteristics are the time required for attacking, success likelihood, etc.

The domains $T_k, S_\imath, L_\jmath$ may represent continuous values such as reals, or may reflect the magnitude and may be measured in levels (e.g. low, medium, high). In example, one may consider an attacker profile $t = (b, s, t)$, where b is the adversarial budget measured in €, s corresponds to the skill level measured on an ordinal scale in terms of levels low, medium, high, and t is the time budget available to the adversary measured on an ordinal scale as well and expressed in the following levels: seconds, hours, minutes, or days.

6.1.1 The Constraint based Approach

The somewhat simplistic constraint based approach emerged from the observation that constructing an attack tree analysts include all possible attack scenarios related to the considered primary threat in an attack tree. Some of the attacks are more realistic than the others, considering the environment in which the analyzed enterprise operates. The fact that some attacks seem not so realistic does not mean that they are impossible or that they will never happen. Therefore a typical attack tree analysis assumes an overpowered adversary who is capable of launching every possible attack, included in the attack tree, against the target enterprise or infrastructure. Applying an attacker profile to an attack tree invalidates certain nodes and eventually entire sub-trees in the attack tree thus enabling the independent analysis of the derived attack scenarios containing only attacks feasible for the considered class of threat agents. Depending on the severity of the adversarial limitations set in the attacker profile the derived attack scenario may be much smaller than the original attack tree containing all possible attacks and thus much easier to analyze.

An attacker may be unable to execute an attack[1] when the attacker does not have all the necessary equipment, knowledge and skills to execute an attack, or when executing an attack takes more resources or effort than the attacker can invest into attacking. The time that an attacker can invest into attacking is bounded by various factors like individual risk tolerance levels of attackers, the strength of the deployed security control mechanisms like intrusion detection/prevention systems, surveillance tracking solutions, or by the adversarial risk aversion strategy. For instance the attack “Decipher an RSA-encrypted network traffic” requires certain knowledge in cryptography and mathematics and besides, the time required for brute-forcing the RSA encryption definitely exceeds the time that an attacker can invest into attacking. Such an attack might be feasible for a national security agency, but not for a script-kiddie.

It can be seen that if the skill level of the attacker is insufficient to execute a certain attack, the success likelihood of the attacker in the considered attack is close to zero. Likewise, in the case when the minimal time required to execute an attack exceeds the available time budget of the attacker, his success likelihood in the considered attack will be zero. Attacker profiles can help to eliminate the attacks which the considered class of attackers cannot execute and thus eliminate certain nodes and entire sub-trees from the attack tree. The resulting attack tree is smaller and easier to analyze.

It is possible to match the strength of infrastructure component (infrastructure characteristic) against the attacker skill level (threat agent characteristic) and likewise to match the minimal time required to successfully execute an attack (infrastructure characteristic) against the attacker time budget (threat agent characteristic) as shown in Table 6.1.

In the considered case the attacker profile is represented by a triplet of adversarial characteristics (s, t, b) where s is the attacker skill level and t is the amount of time that the attacker can invest into attacking, and b is the budget available

[1]being unable to execute an attack should be differentiated from being unwilling to execute an attack – an attacker may wish to execute an attack but still be unable to execute it

Table 6.1: Infrastructure properties and related adversarial characteristics

Property of the infrastructure component	Quantitative annotation on the attack tree leaf	Characteristic of the threat agent
component strength	difficulty	skill level
minimal time required to exploit a vulnerability	minimal time required to successfully execute an attack	available time budget

to the attacker. The attack profile is a triplet of attack properties (c, d, a), where c corresponds to the preparation costs in order to launch the attack, d is the difficulty of attacking and a is the minimal time required to succeed in the attack. The attacking profile is represented by the parametric function $f_1(d, s, a, t, b, c)$ described by equation (6.1.1) determines the feasibility of executing the considered attack by a given class of attackers and returns true iff an attacker described by an attacker profile can execute the considered attack and false otherwise.

$$f_1(d, s, a, t, b, c) = \begin{cases} 1, & \text{iff} \begin{cases} d \leqslant s & , \\ a \leqslant t & , \\ c \leqslant b & , \end{cases} \\ 0, & \text{otherwise,} \end{cases}$$

Let us consider an attack tree shown in Fig. 6.1 and an attacker profile describing a skillful attacker with little money so that the adversarial budget is 100 €, the skill level is high, and the available time budget is several days. It can be seen that the attack titled "*Exploit vulnerability*" is annotated with cost 200 € and therefore the considered class of attackers cannot launch this attack step. This attack step cannot be launched by the adversaries described by the attacker profile also for the reason that the difficulty of the attack (very high) exceeds the attacker skills (high). Thus, the attack "*Exploit vulnerability*" is eliminated from the attack tree. The attack tree obtained after applying attacker profile to it is shown in Fig. 6.2.

6.1.2 Item Response Theory based Approach

The drawback of the constraint-based approach introduced in the previous section is that if the attacker profile does not match the specified constraints it is assumed that he is unable to execute the considered attack – his likelihood of success is zero and therefore the corresponding nodes and entire sub-trees can be eliminated from the attack tree. This is not true in all cases. Even if the difficulty of exploitation is very high and the attacker skills are low, there is still a residual chance that the attacker will accidentally succeed in attacking and the constraint-based approach does not take this into account. Another problem is that if attackers corresponding to a certain profile can execute a certain attack, the resulting success likelihood is determined by the infrastructure properties as well the indi-

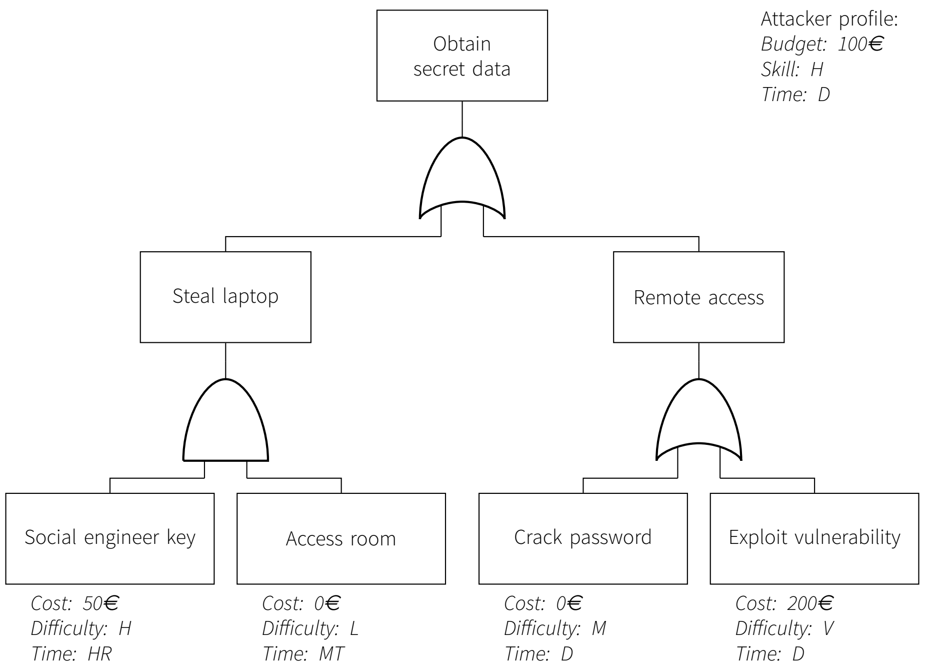


Figure 6.1: An example attack tree to steal the secret data with quantitative annotations on the leafs

vidual characteristics of the considered class of attackers. The problems outlined above can be tackled by applying the elementary Item Response theory [29] to define the relations between the underlying components of quantitative security risk metrics more precisely.

Different functions may be used to denote relations between the quantitative security risk annotations and the underlying components which altogether determine the considered annotation with varying degree of granularity. Let us start with a simplified case in which we assume that the success likelihood of a particular attacker in executing a particular attack depends only on the infrastructure component strength δ. The more difficult is it to exploit the vulnerability in the considered infrastructure component, the less probable will the threat exploit this vulnerability, materialize and cause damage to the affected asset. We say that let the likelihood of success be 0.5 in case the difficulty is medium. Thus we have a fixed point of reference and we need to come up with a continuous function following the following considerations:

- the value domain of the function should be bound by the probabilistic domain – $[0, 1]$.
- it might not be desirable that the function may return zero likelihood in result, rather is should asymptotically approach the zero-level.
- the function should be a monotone descending function within its domain,

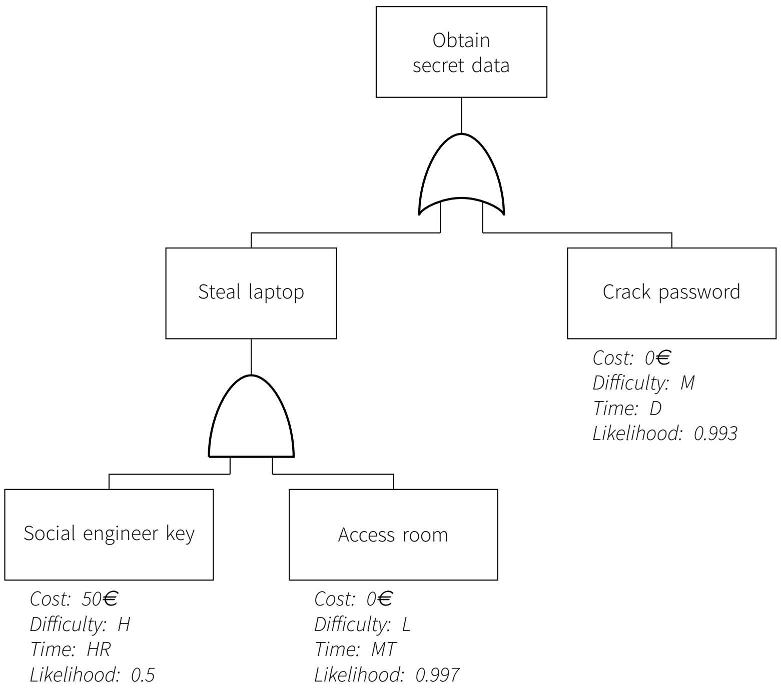


Figure 6.2: An example attack tree after applying the attacker profile

as the greater the component strength is the less is the resulting likelihood.

For example it is possible to use the logistic function as the one shown in Fig. 6.3 indicating that the likelihood will be 0.5 when the component strength is medium to derive the success likelihood in assumption that it depends on the component strength only. It is worth noting that the component strength may still be estimated on the ordinal scale, provided that the experts agree on the mapping between the qualitative annotations like low, medium, high and their corresponding quantitative values.

If to consider that the success likelihood depends not only on the component strength δ but on the adversarial skill level β as well, we need to come up with a function $f(\beta, \delta)$ such that it is bounded with the probabilistic domain $[0, 1]$ and increases whenever β increases and decreases whenever δ increases – the more skillful the attacker is and the less difficult it is to exploit the vulnerability in the considered infrastructure component the more likely is the attacker to succeed in the considered attack step. Similarly to the approach outlined in [29] let the success likelihood be 0.5 when the attacker skill level is equal to the component strength. We get to the logistic function in the form of $p = \frac{e^{(\beta-\delta)}}{1+e^{(\beta-\delta)}}$ as shown in Fig. 6.4.

As it was stated earlier in this chapter the likelihood of success depends at least on the attacker skill level β, component strength δ, and time invested into attacking γ and thus $p = f(\beta, \delta, \gamma)$. This function cannot be expressed graphically as

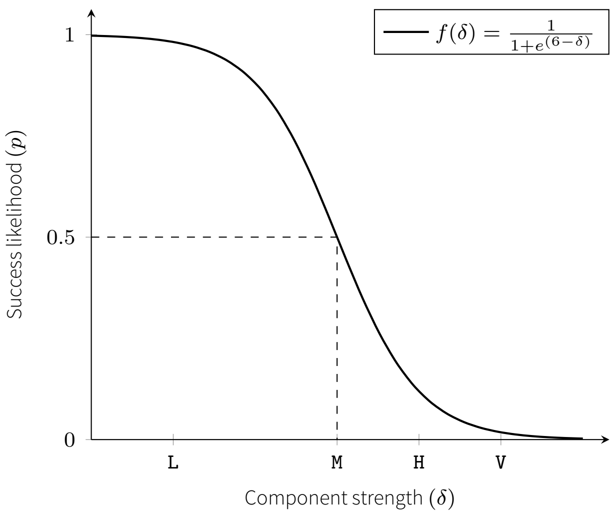


Figure 6.3: Success likelihood p as a function of component strength δ

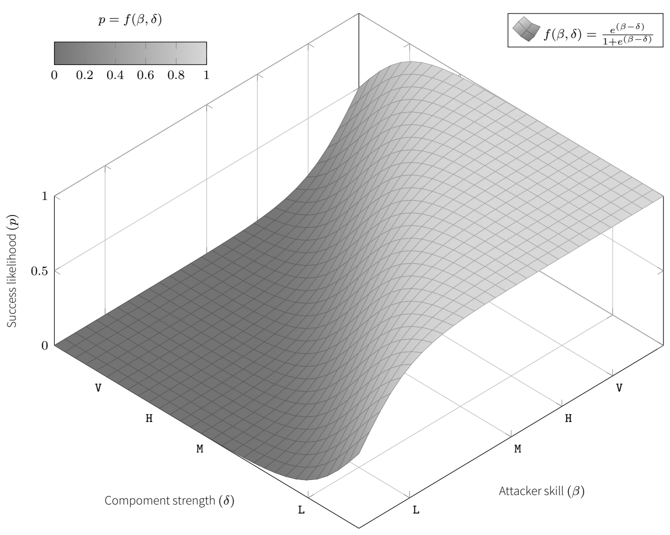


Figure 6.4: Success likelihood p as a function of attacker skill level β and component strength δ

any function in a four-dimensional domain, and it is yet to discover. As an alternative one can derive the likelihood of success from attacker skill level, infrastructure component strength and the time invested into attacking. The ongoing related research focuses on such extensions based on probability distributions [1]. Similar logic may be applied to the annotation time meaning minimal time required for successfully attacking an infrastructure component. It in turn depends (at least) on the attacker skill level and the component strength (the difficulty of attacking). The more skillful the attacker is, the less time it takes for him to accomplish an attack. The more difficult is to attack the more time it takes to accomplish an attack. Some interesting relations may be observed, like less skilled attacker given more time may be as efficient in terms of likelihood of success as more skilled attacker who is given less time for attacking. For the moment there is no specific solution to suggest and the discovery of such a function is left for future research.

6.2 Conclusions

The attacker profiling technique introduced in this chapter is a way to separate infrastructure properties from the properties of the threat agents. This kind of separation allows to estimate and update these properties independently from one another in a timely manner. It becomes possible to derive meaningful estimations of the quantitative annotations of operational security risks from the underlying infrastructure and threat agent characteristics instead of providing imprecise estimations containing a reasonable degree of uncertainty to these annotations directly. One can more precisely estimate how would a complex quantitative annotation such as success likelihood change in case some of its underlying components change or when all of its underlying components change. It is possible to define several typical attacker profiles, e.g. the fame- or curiosity oriented script kiddie, profit-oriented malicious individual, organized crime group, or a national security agency. Such profiles may be crowd-sourced and shared among the interested parties thus forming the public libraries of attacker profiles. As attacker profiles may contain potentially sensitive information it is possible to think about various degrees of sharing like enterprise-internal sharing, trusted sharing among collaborative partners, and publishing. Attacker profiles add flexibility to risk analysis as the risks of the considered enterprise may be studied for various sets of attacker profiles. Likewise, same attacker profiles may be re-used and analyzed with varying enterprise profiles. The attacker profiling approach provides a broader and more detailed overview of the surrounding risk landscape in a timely manner following constant changes in the risk landscape. It allows to undertake informed decisions in assessing the cost-effectiveness of the defensive measures and enable the prediction, prioritization and prevention of emerging attacks in nearly semi-automated way.

CHAPTER 7

APPROXTREE+

The computational methods of the parallel model [14] had exponential complexity and for this reason could analyze attack trees having no more than 30 nodes. In order to improve the performance of the computational method the same authors suggested a new computational method named ApproxTree [16] which relied on genetic algorithm to provide approximated estimations of the adversarial utility in a relatively robust way. This method was not limited to analyzing the attack trees having no common moves and could analyze attack trees in the form of a propositional directed acyclic graphs (PDAGS) in reasonable time. As this computational method was promising for possible practical use it was desirable to enhance it by introducing limited adversarial budget as well as attacker profiling considerations into it. This chapter focuses on introducing the new computational method named ApproxTree+ [21] which is the modification of the ApproxTree method. The enhancements include the ability to take limited adversarial budget into account as well as integration of attacker profiling into it. One of the objectives of integrating adversarial profiling into the computational method was to determine if integration of attacker profiling into existing quantitative risk assessment methods brings any performance overhead along that might render these methods inapplicable for practical use.

The related models and computational methods were described in the previous chapters and therefore for the description of the parallel model the reader is referred to Section 3.2, and for the genetic approximations of the parallel model (the ApproxTree method) the reader is referred to Section 3.4 of Chapter 3.

7.1 THE APPROXTREE+ METHOD

The analysis method can be described by the following rules:

1. The attacker constructs an attack tree and evaluates the quantitative annotations attached to each leaf in the attack tree following these considera-

tions:

  - In order to prepare and launch an attack $x_\imath$ the attacker has to invest a certain amount of resources $\mathcal{C}_\imath$.
  - The attack $x_\imath$ may succeed with probability $p_\imath$ and fail with probability $1 - p_\imath$.
  - Depending on the deployed security controls the attacker sometimes has to carry additional costs after failing or succeeding with the attack. The sum of all the preparation and additional costs is denoted by the $Expenses_\imath$ parameter.
  - Additionally, there is a global parameter *prize* denoted by $\mathcal{P}$ for the whole attack tree that describes the benefit of the attacker in case he succeeds in materializing the primary threat.

2. The attacker considers all potential attack suites – subsets $\sigma \subseteq \mathcal{X}$, where $\mathcal{X} = \{x_1, \dots, x_n\}$ is the set of all attacks considered in the attack scenario. Some of the attack suites satisfy the Boolean function $\mathcal{F}$ of the attack tree, some do not. For the satisfying attack suites the attacker computes the outcome value $Outcome_\sigma$.
3. Finally the attacker chooses the most profitable attack suite and launches attack from the suite simultaneously.

The computational method introduced in [16] aimes at maximizing the expression

$$Outcome_\sigma = p_\sigma \cdot \mathcal{P} - \sum_{x_\imath \in \sigma} Expenses_\imath$$

over all the assignments $\sigma \subseteq \mathcal{X}$ that satisfy the monotone Boolean function $\mathcal{F}$. The success probability of the primary threat p_σ can be computed in time linear in the size of the elementary attacks n using (7.1)

$$p_\sigma = \sum_{\substack{\mathcal{R} \subseteq \sigma \\ \mathcal{F}(\mathcal{R} := true) = true}} \prod_{x_\imath \in \mathcal{R}} p_\imath \prod_{x_\jmath \in \sigma \setminus \mathcal{R}} (1 - p_\jmath) \ . \qquad (7.1)$$

7.2 Adversarial Budget Limitations

For the same reasons as the ones outlined in Chapter 5 it is desirable to investigate the case when the adversarial budget is limited as it seems to be a natural assumption which increases the granularity of attacker models and brings the analysis more close to reality. Due to the stochastic nature of the genetic approximation approach the possible increase in complexity should be negligible, if any at all due to the fact the attacker profiling results in search space reduction and a considerable drop in complexity resulting from this.

A monotone Boolean function $\mathcal{F}(x_1, \ldots, x_n)$ corresponding to the considered attack tree where every variable $x_\imath \in \mathcal{X}$ is annotated with cost $\mathcal{C}_\imath$, success probability $p_\imath$, difficulty $d_\imath$ and minimal time required to exploit the related vulnerability $t_\imath$ is an input to the ApproxTree+ method. The computational method considers all possible attack suites which are defined as follows:

Definition (Attack Suite). *Attack suite $\sigma \subseteq \mathcal{X}$ is a set of elementary attacks which have been chosen by the attacker to be launched and used to try to materialize the primary threat.*

The adversarial utility is optimized over the entire set of the so-called satisfying attack suites and the most profitable attack suite together with the related utility is returned in result. A satisfying attack suite is defined as follows:

Definition (Satisfying attack suite). *A satisfying attack suite σ evaluates $\mathcal{F}$ to true when all the elementary attacks $x_\imath \in \sigma$ have been evaluated to true.*

The adversarial characteristics are described by the following set of properties:

1. *Budget* $t_b \in \mathbb{R}$ – the monetary resource of the attacker, measured in currency units.
2. *Skill* $t_s \in \langle L, M, H \rangle$ – the skill level of the attacker, measured on an ordinal scale (Low/Medium/High).
3. *Time* $t_a \in \langle S, MT, HR, D \rangle$ – the available time resource of the attacker, measured on an ordinal scale (Seconds/Minutes/Hours/Days).

The attacker-, attack-, and attacking profiles are defined as follows:

Definition (Attacker profile). *An attacker profile is a triplet of adversarial characteristics (s, t, b) where $s \in$ ⟨low,medium,high,very high⟩ is the attacker skill level and $t \in$ ⟨seconds,minutes,hours,days⟩ is the amount of time that the attacker can invest into attacking, and $b \in \mathbb{R}$ is the budget available to the attacker.*

Definition (Attack profile). *An attack profile is a triplet of attack properties (c, d, a), where $c \in \mathbb{R}$ denotes the preparation costs that the attacker has to pay in order to prepare and launch an attack, $d \in$ ⟨low,medium,high,very high⟩ is the difficulty of attacking and $a \in$ ⟨seconds,minutes,hours,days⟩ is the minimal time required to succeed in the attack.*

Definition (Attacking profile). *An attacking profile is a pair of functions $\langle f_1, f_2 \rangle$, where the function $f_1(d, s, a, t, b, c)$ described by equation* (7.2) *determines the feasibility of executing the considered attack by a given class of attackers and returns true iff an attacker described by an attacker profile can execute the considered attack and false otherwise. The function $f_2(\sigma, b)$ described by equation* (7.2) *determines the feasibility of executing an attack suite by the attacker described by a given attacker profile.*

As attack suites σ are generated only during the analysis phase, the function $f_2(\sigma, b)$ is applied only during the run time of the computation algorithm, while function $f_1(d, s, a, t, b, c)$ may be applied before the actual analysis to effectively reduce the search space.

$$f_1(d, s, a, t, b, c) = \begin{cases} 1, & \text{iff} \begin{cases} d \leqslant s & , \\ a \leqslant t & , \\ c \leqslant b & , \end{cases} \\ 0, & \text{otherwise.} \end{cases} \tag{7.2}$$

$$f_2(\sigma, b) = \begin{cases} 1, & \text{iff } b \geqslant \sum_{x_\imath \in \sigma} \mathcal{C}_{x_\imath} \quad , \\ 0 & \text{otherwise} \quad . \end{cases}$$

Now the term profile satisfying attack suite can defined as follows:

Definition. *A profile satisfying attack suite σ is a satisfying attack suite such that:*

$$\forall f_\imath \in L_\jmath : \begin{cases} f_1(d, s, a, t, b, c) \equiv 1 & , \\ f_2(\sigma, b) \equiv 1 & . \end{cases}$$

Before the actual analysis starts the ApproxTree+ launches a single round of profiling on the input attack tree eliminating nodes and sub-trees infeasible for the considered class of attackers – the ones for which the function $f_1(d, s, a, t, b, c)$ returns false. The resulting tree which is already smaller than the initial tree is provided to the genetic analysis method as input. The analysis method considers all potential attack suites σ and runs the second round of profiling on the considered attack suites. During the second round of profiling the computational method disregards the attack suites the execution of which is infeasible for the adversary due to the budget limitations – the ones for which the function $f_2(\sigma, b)$ returns false. The necessity for two separate rounds of profiling can be easily explained – the cost of executing an attack suite capable of materializing the primary threat is unknown prior to the analysis phase which considers all the potential attack suites σ and only then the cost of an attack suite can be matched against the adversarial budget.

7.3 Genetic Approximations

In order to tackle the potential exponential amount of computations in (7.1) a genetic algorithm is used to facilitate the usage of the computational method for large attack trees:

1. Create the first generation of n individuals (profile-satisfying attack suites,

not all of them are necessarily distinct).

2. All the individuals in the initial population are crossed with every other individual producing $\binom{n}{2}$ new individuals.
3. Each individual is mutated with probability p.
4. The mutated population is joined with the initial population
5. Finally, k fittest profile-satisfying individuals out of the $\binom{n}{2} + n$ individuals are selected and thus form the next generation.

In the ApproxTree implementation [16] the genetic algorithm ran for a fixed number k of generations and stopped. The resulting outcome was considered to be pretty good approximation of the result. In the author opinion this is a weak hypothesis as the number k was chosen arbitrarily based on the expert knowledge and cannot claim to be a pretty good approximation of the actual result. The convergence indicator in the genetic algorithm used in ApproxTree+ was modified as follows: we say that the method has converged when k last individuals did not increase the result any more. Experiments with the ApproxTree+ analysis tool have shown that there are many extra reproduction cycles that the genetic algorithm executes because there is still a possibility for tiny improvement – the outcome may be improved by a couple of €, then by a couple of cents, then by a smaller fractions. In practice such precision is not required and only consumes computational resources. For this reason an additional parameter δ was introduced which denoted the minimal difference in the outcomes of the two subsequent generations of individuals which could be considered an improvement. Then the convergence indicator may be re-defined as follows: the method has converged whenever k last generations do not improve the result more than the threshold δ. There is still a potential pitfall which can make the algorithm run infinitely and it is related to the attacker profiling. The attacker profile may contain so tight restrictions on the adversarial budget (in some cases the budget may at all be equal to zero) that there will be no chance to generate n individuals fitting the considered attacker profile, e.g. in the case when there are no attack suites with zero cost. An additional check was added to the initial population generation phase in which the number of attempts to generate the initial population are counted and it this number exceeds a certain threshold m the algorithm stops. The algorithm may fail to generate the initial population for various reasons – for instance, when the attacker profile limitations are too strict and no profile-satisfying individuals exist. In this case the considered enterprise is secure as there are no profitable ways for the considered attacker described by an attacker profile to attack the enterprise. Likewise the algorithm may fail to generate the initial population due to the stochastic nature of the approach and non-systematic traversal through the landscape of possible solutions. As the individuals are generated randomly there is a chance that the profile-satisfying individuals were not generated and were left behind. It is not possible to provide a definitive answer why the initial population could not be generated without any knowledge about the search space landscape.

7.4 Performance Analysis

In order to assess the performance of the introduced computational method a benchmarking experiment was conducted. A set of random attack trees with random quantitative annotations on the leaves was generated. The attack tree generation procedure was a two-step process. The outcome of the first step was the randomly generated monotone Boolean function with predefined number of variables and from 2 to 5 operands per operator. The choice of the operator type (either conjunction or disjunction) was a random choice from the uniformly distributed set of values. The next step attached quantitative annotations to each variable in the considered monotone Boolean function. The cost annotation was estimated on the interval $[100, 1000]$, the success probability was chosen from the interval $(0, 1)$, the choice of difficulty level was a random choice from the set of possible difficulty levels $[low, medium, high, very\,high]$, and the value for the time annotation was a random choice from the set of time scale levels $[seconds, hours, minutes, days]$. The intervals and sets of ordinal levels were considered to be uniformly distributed assuming the uniform distribution of the PRNG output.

The main question that needed to be answered is whether attacker profiling added any extra computational overhead to the method. For this reason the performance of the ApproxTree+ method was compared to the performance of the ApproxTree method. Fig. 7.1 shows the cumulative time distribution among the phases of the ApproxTree+ algorithm considering one single adversarial profile, and Fig. 7.2 shows the time distribution among the phases of the ApproxTree algorithm.

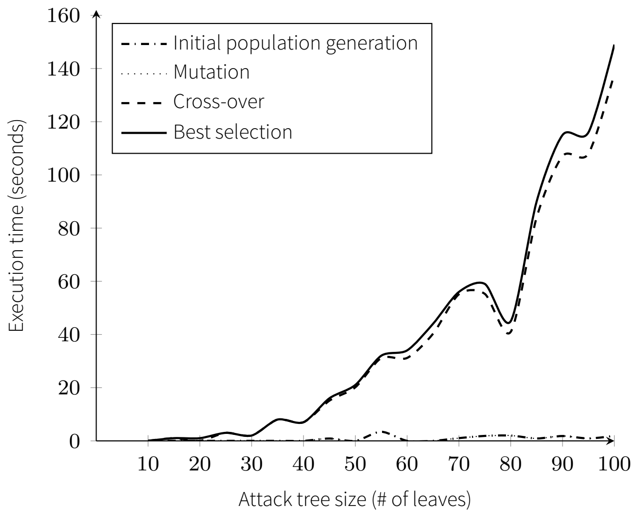


Figure 7.1: Cumulative time distribution of ApptoxTree+ phases

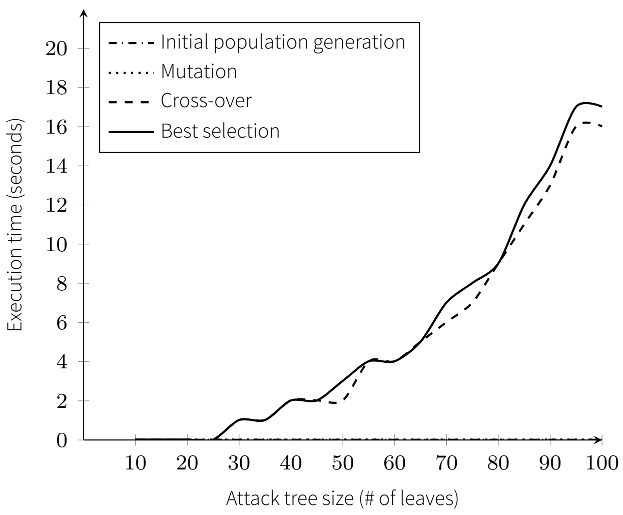


Figure 7.2: Cumulative time distribution of ApproxTree phases

It can be seen from the cumulative time distribution diagram that the integration of attacker profiling does not introduce any significant computational overhead. In both cases the initial population generation phase is almost immediate, as well as the mutation phase. The most computationally extensive phase is the cross-over phase which consumes approximately $85 - 99\%$ of the cumulative time distribution among all the phases. The last phase – the phase where the fittest individuals are selected – does not introduce any significant workload and consumes approximately $1 - 15\%$ of the cumulative time distribution among all the phases. The cross-over phase is the most time consuming as each individual is crossed with every other individual in the population thus producing $n \times n$ cross operations, where n is the amount of individuals in the initial population.

Fig. 7.3 shows that the execution time of the ApproxTree+ algorithm is proportional to the ApproxTree algorithm. The increased execution time arises from the fact that as a rule one does not assess risks using just one single attacker profile, as it is reasonable to assess risks of the considered enterprise using a set of possible attacker profiles so that the results would produce meaningful insight on the risk landscape – thus the overall execution time is proportional to the number of the considered attacker profiles. The analysis of the convergence speed shows that the convergence speed of ApproxTree does not exceed the convergence speed of the ApproxTree+. Additionally it does not depend on the the size of the attack tree – independently from the size of the attack tree the speed of convergence stays approximately at the same level.

Additionally the effect of the genetic algorithm control parameters such as the mutation rate, the size of the initial population on the speed of convergence to assess whether the parameters of the genetic algorithm used by ApproxTree still remain optimal for ApproxTree+. Fig. 7.5 shows that the convergence speed decreases with the increase in the mutation rate from approximately 2 generations

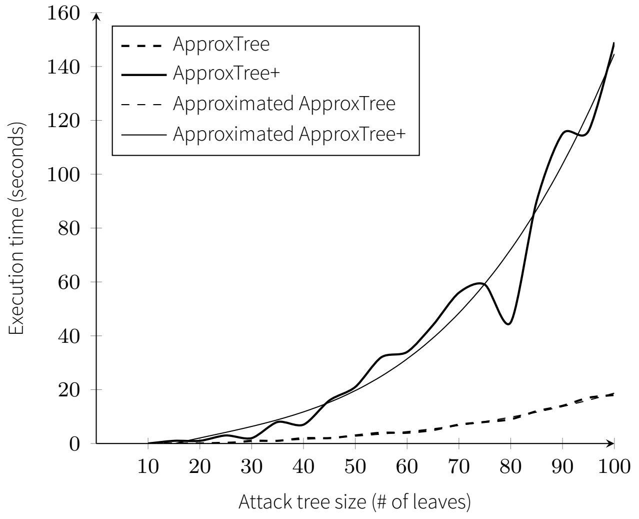


Figure 7.3: Execution time

in the case when the mutation rate is 10% up to 6 generations in the case when mutation rate is 90%.

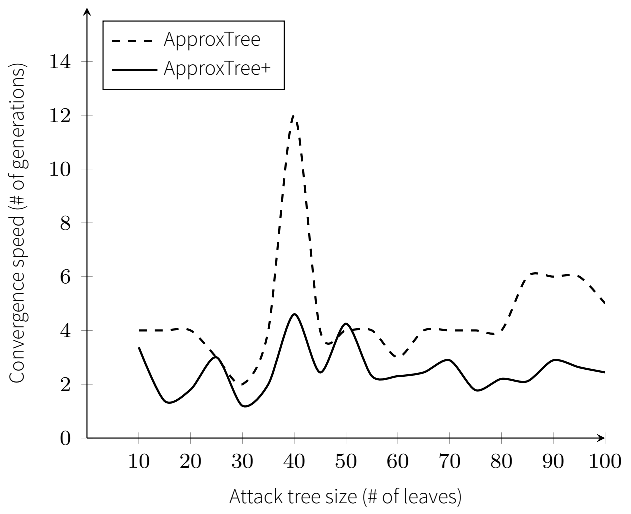


Figure 7.4: Convergence speed

Independently of the mutation rate the speed of convergence of ApproxTree+ algorithm does not exceed the speed of convergence of ApproxTree as shown in Fig. 7.4. The benchmarking results have shown that the mutation step has no significant effect whatsoever on the speed of convergence. We were unable to find any case where the method experience premature convergence and thus produce under-optimal solutions. Even when the mutation phase was excluded entirely the global optimum solution was nevertheless always reached. This might have

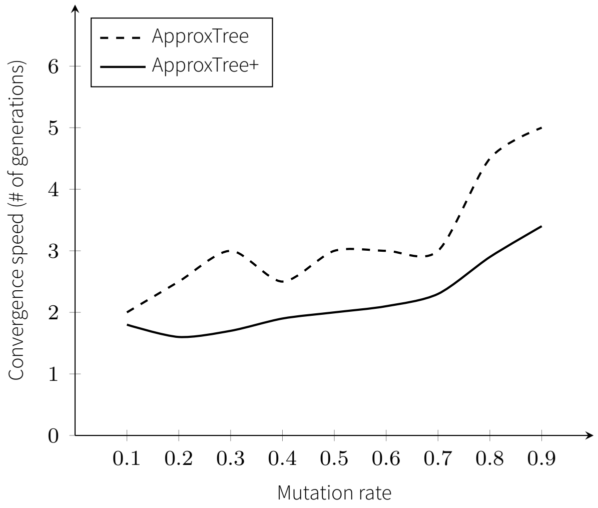


Figure 7.5: Convergence speed as a function of the mutation rate

happened because of the "good" initial population – if the size of the initial population is rather big compared to the number of satisfying solutions it is highly likely that the initial population will contain all the solutions in the first generation already and results in immediate convergence. The immediate convergence was observed in some cases during benchmarking. If the initial population does not contain all the solutions still it may be "good enough" so that the cross-over step produces the entire domain of solutions. Fig. 7.6 shows that the increase in the initial population size the speed of convergence increases stabilizing at a value of approximately 1.6 generations for the initial population size greater than $4n$ (n being the number of leaves in the attack tree) in the case of the ApproxTree approach. In the case of ApproxTree+ we can see slight, but firm decrease in the convergence speed. In some cases when initial population size was less than n the computational method was unable to reach global optimum, which may happen when rather small initial population limits the amount of possible solutions that may be reached and the mutation rate is small enough and does not improve the situation.

The precision assessment shows that in ApproxTree+, as well as in ApproxTree, either the result converges to the global optimum (most profitable attack suite) or the computational method fails to generate the initial population of individuals. In case of profiling, the attacker profile may contain so strict constraints that not a single profile satisfying attack suite may exist. The more strict constraints are used in the considered attacker profiles the higher is the probability that no profile satisfying individuals will be generated. However it would be a false claim to state that the profile satisfying solutions definitely do not exist in this case, as the state when ApproxTree+ is unable to generate the initial population corresponds to the two possible conditions – either no profile satisfying attack suites exist (and

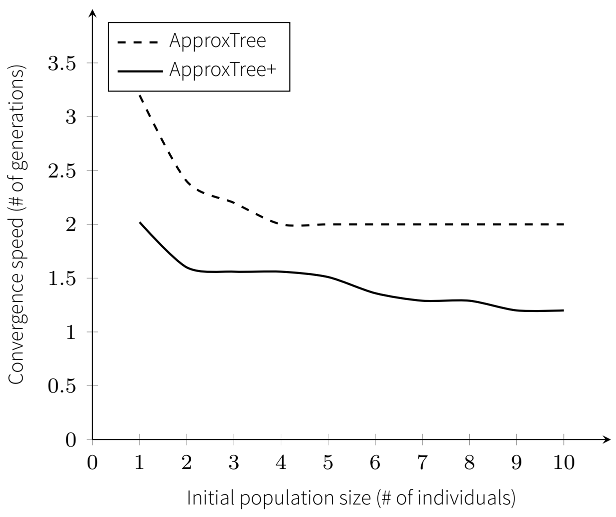


Figure 7.6: Convergence speed as a function the size of the initial population

thus the considered attack scenario has no profitable solutions), or such attack suites exist, however the attack suite generation procedure failed to generate profile satisfying individuals for the initial population due to the stochastic nature of the process.

7.5 Conclusions

The chapter demonstrated the application of attacker profiling in the framework of attack tree analysis by introducing the new analysis tool named ApproxTree+ and demonstrating that integrating attacker profiling into an existing analysis method does not introduce any significant computational overhead.

Chapter 8

Attack Tree Analyzer

In order to show that the considered organization is insecure in the improved failure-free model it is sufficient to find at least one critical min-term of the monotone Boolean function with weight less than $\mathcal{P}$. The problem of finding such a critical min-term – the weighted monotone satisfiability (WMSAT) problem – was shown to be NP-complete. For the details the reader is referred to Theorem 4.3.1 in Chapter 4. The efficient computational methods which use expenses propagation technique can obtain the adversarial utility upper bounds in time linear in the size of the attack tree. In the case of the so-called independent attack trees – trees having no common moves – the propagation methods produce exact result. However real-life attacks tend to re-use intermediate achievements and sub-goals for the benefit of the attacker. Such attack can be graphically expressed in the form of a directed acyclic graphs – DAGs. Analyzing DAGs is possible if to apply cost reduction prior to analysis. Reducing costs makes the game easier for the attacker and pushes the upper bound even further thus resulting in even more over-secured systems. The amount of the required extra investments into security has not been estimated yet. This chapter outlines the genetic approach for this challenge and introduces the adaptive genetic algorithm to solve a monotone satisfiability problem in the improved failure-free model, approximating the exact adversarial outcome from below. The corresponding tool named the Attack Tree Analyzer [23] is capable of providing an approximated estimation of the exact adversarial utility in the improved failure-free model, and can come up with the result in reasonable time. Since genetic algorithms depend on a number of non-trivial control parameters we face a multi-objective optimization problem and we consider several heuristic criteria to solve it.

8.1 Genetic Algorithm

A genetic algorithm is typically characterized by the set of the following parameters:

- a genetic representation of chromosomes or individuals (feasible solutions for the optimization problem)
- a population of encoded solutions
- fitness function which evaluates the optimality of the solution
- genetic operators that generate a new population from the existing one: selection, cross-over, mutation
- a set of control parameters: population size, cross-over and mutation rates, conditions for reproduction termination

The reproduction process as well as the conditions under which the reproduction process terminates is the same as in the ApproxTree+ algorithm outlined in Chapter 7. An individual is any feasible solution to the considered optimization problem. Considering the optimization problem the improved failure-free model is facing, a feasible solution is any profile-satisfying min-term of the monotone Boolean function. Linear binary representation of individuals has been chosen to facilitate the robustness of the cross-over and mutation operations. The algorithm used to generate individuals is demonstrated by Algorithm 8.1.1.

Algorithm 8.1.1: Individual generation algorithm

```
Data: The root of a propositional directed acyclic graph (PDAG)
      representing a monotone Boolean function. An empty individual
      with all bits set to 0.
Result: Live individual.
Function GeneratePopulation (root, empty individual)
if the root is a leaf then
    get the index of the leaf
    set corresponding individual's bit to 1
else if the root is a conjunctive refinement then
    forall the children of the root do
        recursive call: child considered as root parameter
else if the root is a disjunctive refinement then
    choose at least one child
    forall the chosen children do
        recursive call: child considered as root parameter
```

We allow duplicate entries to be present in the population for the sake of maintaining generic variation and keeping the population size constant throughout the reproduction process. It is well known in the world of genetic algorithms that genetic variation directly influences the chances of premature convergence – thus increasing genetic variation in the population is one of the design goals.

The choice of the population size is important – to small population does not contain enough genetic variation to maintain the exploration capabilities, while too big population already contains enough genetic variation to efficiently explore the search space and only results in the performance overhead in the cross-over operation. This means that there exists an optimal population size corresponding to the minimal population size capable of producing the best result. Thus the optimal size of the population sets the lower bound for reasonable population size and the upper bound is solely based on the performance considerations – what is the reasonable time the analysts would agree to wait for the analysis to come up with the result. In the case of sub-optimal population size there is a high risk to converge to sub-optimal solutions and if the population size is bigger than the optimal size it does not add any value, except for the increase in time required run the analysis. If the optimal population in some certain case is k% of the size of the attack tree (the number of leaves in the attack tree) then any population size greater than k% and capable of producing the result in reasonable time would suit to be used for analysis.

All the empirical tests presented in this chapter were executed on the following hardware: PC/Intel Core i5-4590 CPU @ 3.30 GHz, 8GB RAM, Windows 8.1 (64 bit) operating system. In order to determine the lower bound for suitable population size an experiment was conducted on a set of randomly generated attack trees of different sizes ranging from 10 to 100 leaves with step 3. Every attack tree was analyzed with the genetic algorithm using varying population sizes ranging from 2 to 50 individuals with fixed cross-over operator type and mutation rate. For every execution of the genetic algorithm the outcome precision was measured. Precision was calculated using the following equation:

$$\text{Precision} = \frac{\text{Genetic algorithm outcome}}{\text{Precise outcome}} \quad ,$$

where the precise outcome corresponds to the adversarial utility calculated using formula (4.3) in Section 4.4.1 of Chapter 4 which is exact and used as the baseline – the exact utility to which any outcome of approximated calculations is compared. Thus the precision ranges from 0 to 100%.
Fig. 8.1 is an example of a single measurement taken in the case of the randomly generated attack tree with 100 leafs using uniform cross-over operator and mutation rate 0.1. It can be seen that indeed the lower bound for feasible population size exists and in this particular case corresponds to the population size of 8 individuals. In the case of smaller population size the result is sub-optimal. Starting from the population size equal to 8 individuals the approximated result reaches the global optimum and remains exact with the increase in the population size. The research revealed that there is no obvious nor straightforward relation between the size of the analyzed tree and the optimal population size. Apart from the size of the tree the optimal population size might depend on (at the very least) the structure of the attack tree itself. Fig 8.2 shows the percentage of the attack

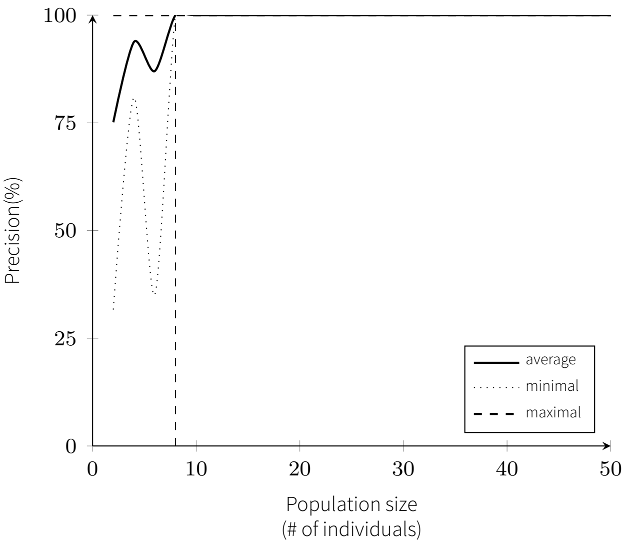


Figure 8.1: Optimal population size

trees in the conducted experiment for which the specific population size would fit. It can be seen that in general the population size equal to 180% of the size of the tree would fit every considered attack tree. The population size 200% chosen by Jürgenson *et al.* for their ApproxTree algorithm [16] was quite a reasonable choice.

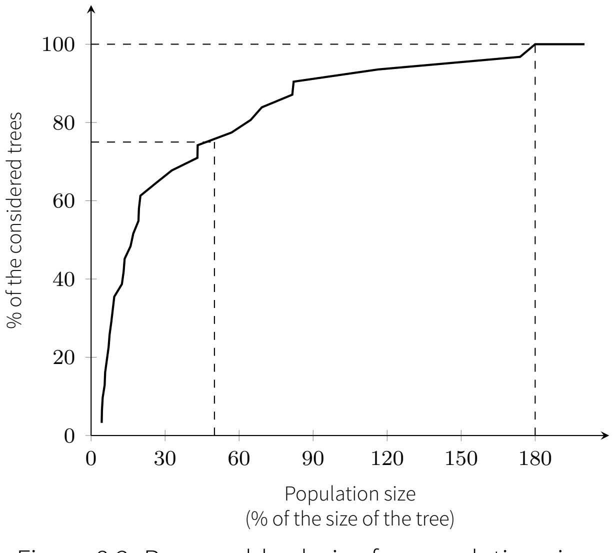


Figure 8.2: Reasonable choice for population size

We show that the cross-over operations take 90-99% of the cumulative time required to run the analysis [21]. Fig 8.3 shows the time measurement for the introduced genetic algorithm depending on the size of the population.

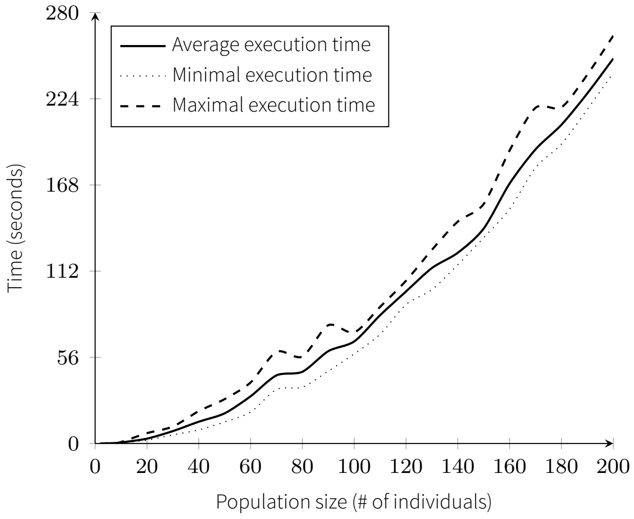


Figure 8.3: Population size effect on GA execution time

The fitness function calculates the utility of the satisfiability game defined by equation (4.1) in Chapter 4. The power of the genetic algorithm arises from the cross-over which causes randomized but still structured exchange of genetic material between individuals in assumption that "good" individuals will produce even better ones. The cross-over rate controls the probability at which individuals are subjected to cross-over. Individuals that are not subjected to cross-over remain unmodified. The higher cross-over rate is the quicker the new solutions get introduced in the population. At the same time chances increase for the solutions to get disrupted faster than the selection can exploit them. The selection operator selects individuals for crossing and its role in the algorithm is to direct the search towards promising solutions. In the genetic algorithm introduced in this chapter parent selection is entirely disabled defaulting to crossing every individual with every other individual and thus the cross-over rate is equal to one. This choice can be justified by the scalable selection pressure which comes along with the selection mechanisms after the reproduction.

Notable cross-over techniques include the single-point, the two-point and the uniform cross-over types. Figures 8.4 and 8.5 demonstrate the differences between the convergence speeds resulting from using various types of cross-over operators.

It can be seen that the considered cross-over operators do not have any significant differences nor effect on the convergence speed of the genetic algorithm. The choice fell upon the uniform cross-over type as it enables more exploratory approach to cross-over than the traditional exploitative approach resulting in a more complex exploration of the search space maintaining at the same time the exchange of "good" information. The algorithm representing the cross-over oper-

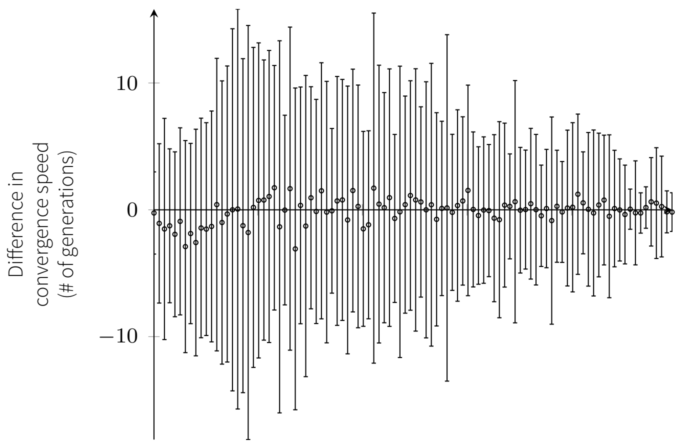


Figure 8.4: Uniform crossover compared to single point crossover

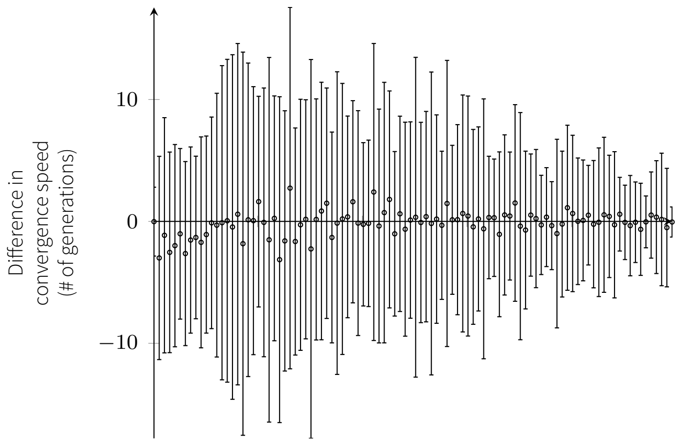


Figure 8.5: Uniform crossover compared to two point crossover

ation is shown in Algorithm 8.1.2.

The role of the mutation operator is to restore lost or unexplored genetic material into the population thus increasing the genetic variance and preventing premature convergence to sub-optimal solutions. The mutation rate controls the rate at which "genes" are subjected to mutation. High levels of the mutation rate turn genetic algorithm into a random search algorithm, while too low levels of mutation rate are unable to restore genetic material efficiently enough and thus the algorithm risks to converge to sub-optimal solutions. Typically the mutation rate is kept rather small in the range $0.005 - 0.05$. In the genetic algorithm for the improved failure-free model the mutation operation is a part of the cross-over operation which mutates the genes which have the same value in the corresponding positions in both of the parent individuals. The uniform cross-over randomly picks corresponding bits in parent individuals to be used in the new generated individual and thus in the case when corresponding bits are different this already pro-

Algorithm 8.1.2: The uniform crossover operation

```
Data: The population of individuals represented as a sorted set.
Result: The population with new added individuals, created during the
        crossover operation.
1 Function GeneratePopulation (sorted set of individuals)
2 initialize a new set of individuals
3 forall the individual i in the population do
4     forall the individual j different from i do
5         new individual := the result of cross operation between individuals i
          and j
6         if new individual is alive then
7             add the new individual to the set of new individuals
8 add the set of new individuals to the population
```

vides sufficient genetic variation. However in the case when bits have the same value this yields just a single choice and the bit in the same position will have the same value as both of its parents. In order to increase the genetic variation (compared to its parents) only these bits are mutated.

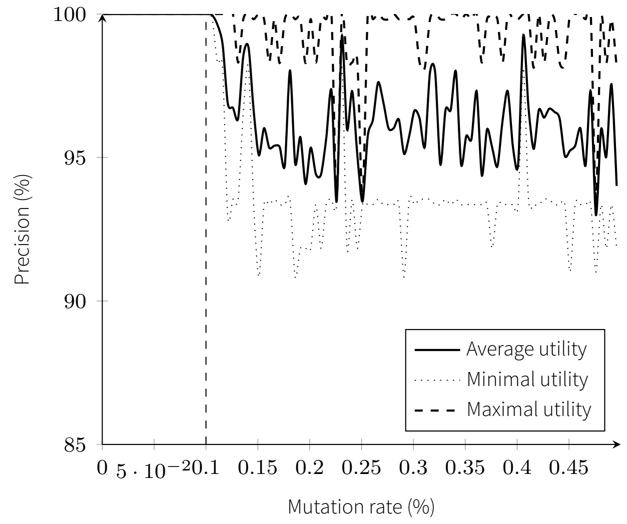


Figure 8.6: GA mutation rate effect

Fig. 8.6 demonstrates the effect of the mutation rate on the utility function by an example of an attack tree with 100 leaves analyzed with the genetic algorithm with initial population size equal to 50 individuals. It can be seen that when the mutation rate exceeds value 0.1 the genetic algorithm turns into a random search algorithm – thus it is reasonable to keep the mutation rate considerably small.

Similar experiments were conducted on a larger set of randomly generated attack trees and the results have shown that the optimal mutation rate is not necessarily small – in some cases the optimal mutation rate was 0.6 or even higher. Thus the optimal mutation rate cannot be preset from the very beginning – it highly depends on the structure of the considered fitness landscape. However it is still reasonable to follow the general rule of thumb and keep the mutation rate small assuming that this should work well in the majority of the cases.

It is important to determine the practical applicability boundaries for the suggested genetic algorithm. Practical applicability here means the maximal size of the attack tree which can be analyzed in reasonable time. The time frame of two hours was selected as the time frame to denote "reasonable time". Extrapolation of the time consumption curve in Fig. 8.7 shows that theoretically the genetic algorithm is capable of analyzing attack trees containing up to 800 leaves in reasonable time. This can be considered an advancement compared to the ApproxTree algorithm [16] which would require more than 900 hours to complete such a task. The execution time complexity estimations are outlined in Table 8.1.

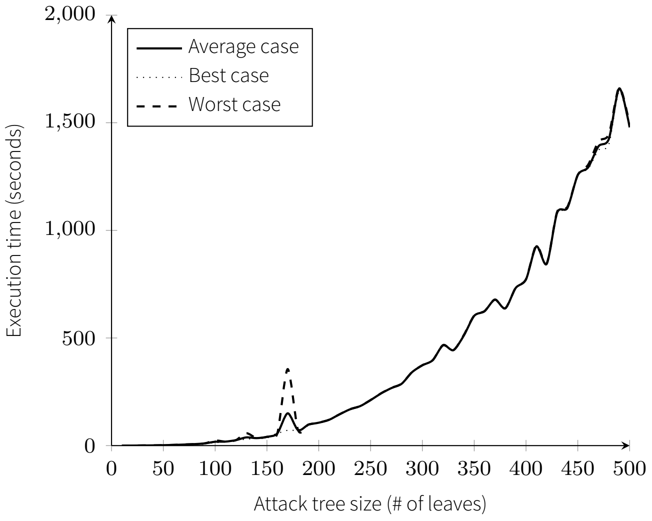


Figure 8.7: GA execution time

Table 8.1: GA execution time complexity estimations

Case	Approximation polynomial	R^2 coefficient
Worst	$1.68 \cdot 10^{-5}n^3 - 0.003n^2 + 0.7015n - 23.03$	0.99
Average	$1.41 \cdot 10^{-5}n^3 - 0.001n^2 + 0.25n - 8.81$	0.99
Best	$1.26 \cdot 10^{-5}n^3 + 1.62 \cdot 10^{-5}n^2 + 0.047n - 2.55$	0.99

For comparison, the execution time complexity of the ApproxTree algorithm was estimated to be $\mathcal{O}(n^4)$, where n is the number of leaves in the attack tree. Such a difference in performance results from the fact that ApproxTree runs for a fixed number of generations, whereas the genetic algorithm introduced in this chapter runs until local convergence, as well as the fact that the utility function used as a fitness function used in the ApproxTree algorithm and corresponding parallel model by Jürgenson *et al.* [14] is considerably more complex compared to the corresponding utility function used in the improved failure-free model.

8.2 Adaptive Genetic Approach

The section compares the genetic algorithm suggested in Section 8.1 to the adaptive genetic approach [36]. The authors suggest to vary the rates of the cross-over and mutation operations adaptively depending on the fitness values of the solutions in the population. High fitness solutions are "protected" while the solutions with sub-average fitness are totally disrupted. The paper introduces a way to detect whether the algorithm is converging by evaluating the difference between the maximal and the average fitness values $f_{\max} - \bar{f}$ in the population which is likely to be less for the population which is converging than for a population scattered across the solution space. Thus the corresponding values of the mutation and crossover rates are increased when the algorithm is converging to an optimum solution and decreased when the population gets too scattered. The authors concluded that the performance of such an adaptive genetic algorithm is in general superior to the performance of genetic algorithms but varies considerably from problem to problem. This section introduces the adaptive genetic algorithm suitable for analyzing the improved failure-free model and compares this approach to the genetic algorithm outlined in Section 8.1.

In the case of the adaptive genetic algorithm the corresponding crossover and mutation rates are initialized with their initial values which further on change adaptively during the run time of the algorithm and the only parameter which remains fixed is the population size. Similarly to the genetic algorithm there exists a lower bound for the optimal population size corresponding to the minimal population size capable of producing the maximal result.
Fig. 8.8 demonstrates the result corresponding to the computations using various population sizes in the experiment setup similar to the one for the genetic algorithm. In the case of the genetic algorithm the maximal value was stable with the increase in the population size, however in the case of the adaptive genetic algorithm some fluctuations were present.

Fig. 8.9 shows how many trees (%) from the conducted experiment the considered population size would fit. It can be seen that in general the population size equal to 200% of the size of the tree would fit every considered attack tree. Based on these observations it is possible to say that the adaptive genetic algorithm seems to be more robust but less stable compared to the genetic algorithm and requires bigger population size in order to produce optimal results for the ma-

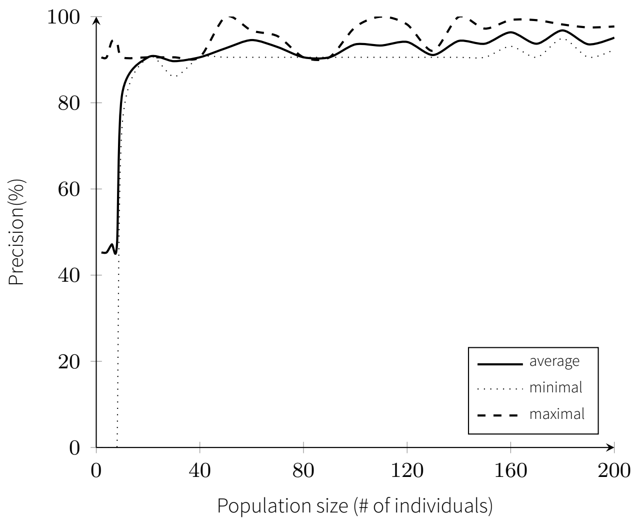


Figure 8.8: Optimal population size

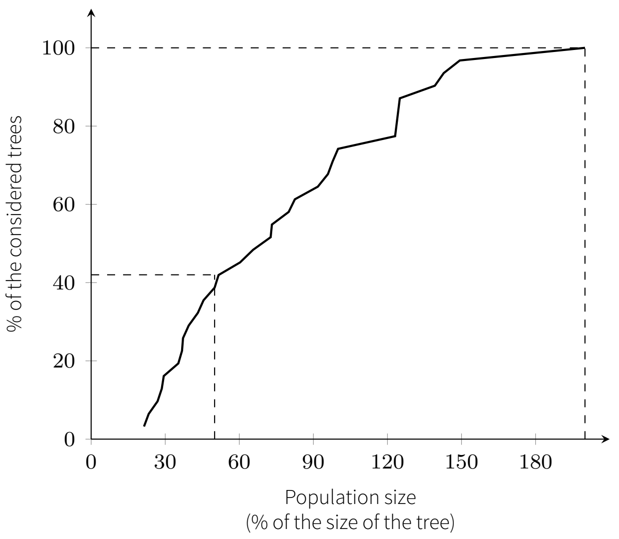


Figure 8.9: Reasonable choice for population size

jority of the cases.

Similarly to the genetic algorithm we estimate the maximal size of the attack tree which the adaptive genetic algorithm is capable of analyzing within reasonable time frame set to two hours. Extrapolating the time consumption curve with the most extreme values trimmed out in Fig. 8.10 a conclusion can be made that theoretically the adaptive genetic algorithm is capable of analyzing attack trees containing up to 26000 leaves in reasonable time and that is approximately 32 times more efficient compared to the genetic algorithm.

The execution time complexity estimations for the adaptive genetic algorithm

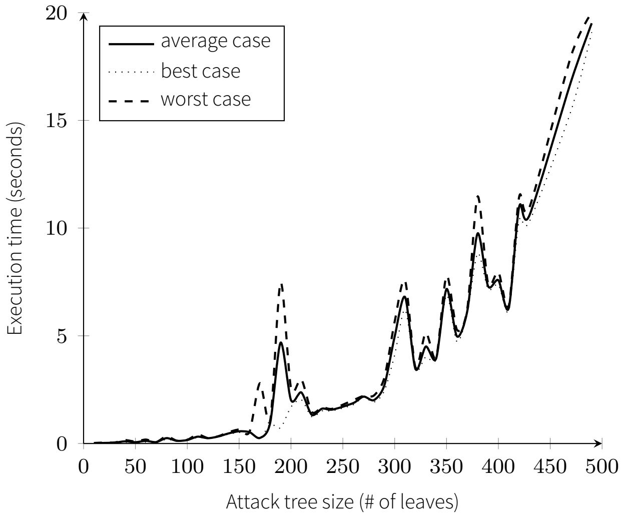


Figure 8.10: AGA execution time

are outlined in Table 8.2.

Table 8.2: AGA execution time complexity estimations

Case	Approximation polynomial	R^2 coefficient
Worst	$3.985x^3 - 0.0001x^2 + 0.0358x - 1.1970$	0.90
Average	$3.5731x^3 - 0.0001x^2 + 0.0267x - 0.8786$	0.94
Best	$3.1892x^3 - 0.0001x^2 + 0.0192x - 0.6115$	0.96

8.3 Conclusions

This chapter addressed the problem of efficient approximation of attack tree evaluation of the improved failure-free game. The genetic approach was considered as one of the possibilities for approximation since it is known to have worked on similar problems previously. However genetic algorithms depend on a variety of loosely connected parameters (e.g. cross-over and mutation operations and their corresponding rates). Selecting them all simultaneously is a non-trivial task requiring a dedicated assessment effort for each particular problem type. This chapter presents the first systematic study of the of the genetic algorithm control parameters optimization for the attack tree evaluation. A series of experiments has been conducted and heuristic evidence for optimal parameter selection was collected. It turned out that adaptive genetic algorithm converges generally faster than the genetic algorithm and provides similar level of accuracy, but at the price of potentially larger population sizes. Since usually there are no major technical obstacles to increasing the size of the population, the adaptive genetic algorithm should be preferred over plain genetic algorithm.

Chapter 9

Conslusions and Future Research

The thesis focuses on analyzing the security of organizations against targeted profit-oriented attackers. There are no scientifically justified and widely accepted metrics of strength against attacks, but such metrics exists in many other engineering areas – for instance, in civil engineering. This observation was central to this work and motivated the author to look for simple reliable computational methods to assess whether the analyzed organization is sufficiently secure, or some extra security controls need to be deployed. The reliability of the computational methods arises from the requirement that such methods should not produce false-positive results. If the security assessment is based on the assumption that an attacker will not attack if it is not profitable for him, every model which tries to calculate the precise adversarial outcome will always contain a margin for an error and the entire attacker model will fail, as exact computational methods are capable of producing false-positive results. Reliable computational methods do not need to calculate the exact result, but approach the result from above thus calculating its upper bound. The upper bound is a reliable metrics for operational security risks. Similar upper-bound oriented approach exists in civil engineering where engineers use upper bounds to calculate the stress in the case of which the building will definitely not break. In a reliable model the security assessment is based on the upper bound analysis and the corresponding computational methods calculate the upper bounds as the result.

The existing models which tried to increase the modeling granularity provide attackers with possibilities to behave in a manner, similar to the attacker behavior in real life, resulted in a substantial increase in complexity of the computational methods. However, if to provide attackers with even more capabilities, the complexity drops.

The author was inspired by the fully-adaptive model of Buldas *et al.* [6]. Due to artificial limitations placed on the adversary either the penalties in their model contain the potential future profits of the adversary (and hence are larger than they are in real life) or the model does not produce reliable upper bounds. Elimination of these artificial limitations resulted in the model which is somewhat more easy to analyze.

Solving the improved failure-free satisfiability game is equivalent to a weighted monotone satisfiability problem, which was shown to be NP-complete, which is good, as problems of finding optimal strategies for security games tend to belong to PSPACE.

Optimal strategies in the new model turned out to be non-adaptive, and the order in which an attacker launches attack steps is irrelevant. The simplicity of optimal strategies in the improved failure-free satisfiability games allowed to create reliable and efficient propagation rules that can calculate adversarial upper bounds in near-linear time. In order to analyze attack trees with common moves using the propagation rules, expenses reduction technique should be applied.

The improved failure-free model assumes adversaries with unlimited budget. If to assume that the adversarial budget is limited, the computational methods become reasonably complex. Optimal strategies in the limited failure-free model are fully-adaptive in the general case, and only in some specific cases they are non-adaptive. It has been shown that if the organization is secure in the budgeted model in the case of a single attack, as well as elementary disjunctive game, it will be secure in the improved failure-free model as well. Only in the case of an elementary conjunctive game the result may differ – namely, an organization may be secure in the budgeted model, and be at the same time insecure in the improved failure-free model. This may happen iff the adversarial reward is estimated with high precision which hardly can be achievable in real life. Due to this the budgeted model may be applied if the adversarial reward can be precisely estimated.

Protecting organizations according to the upper bound oriented approach results in over-secured organizations. The extent to which organizations are over-secured has not been studied yet. The tool named Attack Tree Analyzer uses genetic algorithm to solve the improved failure-free game and approximate the exact result from below and thus allows to estimate the precision and effectiveness of the upper bound oriented methods.

The attacker profiling concept emerged from the observation that it is pretty hard to provide meaningful estimations to some of the quantitative annotations of operational security risk, such as success likelihood, or time required to execute an attack. It turned out that these annotations are jointly influenced by an entire set of underlying characteristics of the infrastructure, as well as the characteristics of the considered threat agents. Attacker profiling is the concept of separation of the infrastructure characteristics from the characteristics of the threat agents. Such an approach allows to calculate quantitative annotations of operational security risks from these values. This approach adds flexibility to operational security risk analysis and provides a deeper insight on the surrounding threat landscape.

Attacker profiling has been successfully integrated into existing risk assessment tool ApproxTree. It has been shown that integrating attacker profiling considerations into the existing risk analysis models does not produce any computational overhead. The resulting tool named ApproxTree+ can take attacker profiling and adversarial budget limitations into account.

The subsequent research will focus on broader research of security games, enhancing the existing model with a defender model. The computational methods will study possible interactions between the attacker and the defender. The decision if an organization is secure will be in this case based on equilibrium analysis of optimal attacker and defender strategies.

Bibliography

[1] Arnold, F., Belinfante, A., van der Berg, F., Guck, D., Stoelinga, M.: Dftcalc: A tool for efficient fault tree analysis. In: Bitsch, F., Guiochet, J., Kaâniche, M. (eds.): SAFECOMP 2013, LNCS 8153, 293–301. Springer (2013)

[2] Arnold, F., Pieters, W., Stoelinga, M.: Quantitative penetration testing with item response theory. Tech. report TR-CTIT-13-20. Centre for Telematics and Information Technology, University of Twente. (2013)

[3] Blomquist, A., Arvola, M.: Personas in action: ethnography in an interaction design team. In: NordiCHI '02, 197–200. (2002)

[4] Buldas, A., Laud, P., Priisalu, J., Saarepera, M., Willemson, J.: Rational choice of security measures via multi-parameter attack trees. In: López, J. (ed.): CRITIS 2006, LNCS 4347, 235–248. Springer (2006)

[5] Buldas, A., Lenin, A.: New efficient utility upper bounds for the fully adaptive model of attack trees. In: Das, S.K., Nita-Rotaru, C., Kantarcioglu, M. (eds.): GameSec 2013. LNCS 8252, 192–205. Springer (2013)

[6] Buldas, A., Stepanenko, R.: Upper bounds for adversaries' utility in attack trees. In: Grossklags, J., Walrand, J.C. (eds.): GameSec 2012, LNCS 7638, 98–117. Springer (2012)

[7] Castro, J.W., Acuña, S.T., Juzgado, N.J.: Integrating the Personas Technique into the Requirements Analysis Activity. In: Gelbukh, A.F., Adiba, M.E. (eds.): ENC'08, 104–112. IEEE (2008)

[8] Faily, S., Flechais, I.: Barry is not the weakest link: eliciting secure system requirements with personas. In: McEwan, T., McKinnon, L. (eds.): BCS HCI, 124–132. ACM (2010)

[9] Faily, S., Flechais, I.: Persona cases: a technique for grounding personas. In: Tan, D.S., Amershi, S., Begole, B., Kellogg, W.A., Tungare, M. (eds.): CHI'11, 2267–2270. ACM (2011)

[10] Friedman, M. In: Essays in positive economics. Univ. of Chicago Press, Chicago, Ill. [u.a.] 15,22,31 (1953)

[11] Geer, D., Hoo, K.S., Jaquith, A.: Information Security: Why the Future Belongs to the Quants. IEEE Security & Privacy **1**(4) 24–32 (2003)

[12] Gollmann, D., Herley, C., Koenig, V., Pieters, W., Sasse, M.A.: Socio-Technical Security Metrics. Dagstuhl Reports **4**(12) 1–28 (2014)

[13] Grüne-Yanoff, T.: Paradoxes of Rational Choice Theory. In: Roeser, S., Hillerbrand, R., Sandin, P., Peterson, M. (eds.): Handbook of Risk Theory, 499–516. Springer Netherlands (2012)

[14] Jürgenson, A., Willemson, J.: Computing Exact Outcomes of Multi-parameter Attack Trees. In: Meersman, R., Tari, Z. (eds.): OTM 2008, LNCS 5332, 1036–1051. Springer (2008)

[15] Jürgenson, A., Willemson, J.: Serial Model for Attack Tree Computations. In: Lee, D., Hong, S. (eds.): ICISC 2009, LNCS 5984, 118–128. Springer (2009)

[16] Jürgenson, A., Willemson, J.: On Fast and Approximate Attack Tree Computations. In: Kwak, J., Deng, R.H., Won, Y., Wang, G. (eds.): ISPEC 2010, LNCS 6047, 56–66. Springer (2010)

[17] Kárpáti, P., Opdahl, A.L., Sindre, G.: Experimental Comparison of Misuse Case Maps with Misuse Cases and System Architecture Diagrams for Eliciting Security Vulnerabilities and Mitigations. In: ARES 2011, 507–514. IEEE (2011)

[18] ”Know Your Enemies” series: Honeynet Project. Know Your Enemy The Tools and Methodologies of the Script Kiddie. http://project.honeynet.org (2000)

[19] ”Know Your Enemies” series: Honeynet Project. Know Your Enemy II: Tracking the blackhat’s moves. http://project.honeynet.org (2001)

[20] ”Know Your Enemies” series: Honeynet Project. Know Your Enemy III: They Gain Root. http://project.honeynet.org (2000)

[21] Lenin, A., Willemson, J., Sari, D.: Attacker profiling in quantitative security assessment based on attack trees. In Fischer-Hübner, S., Bernsmed, K. (eds.): NordSec 2014, LNCS 8788, 199–212. Springer (2014)

[22] Lenin, A., Buldas, A.: Limiting adversarial budget in quantitative security assessment. In: Poovendran, R., Saad, W. (eds.): GameSec 2014, LNCS 8840, 155–174. Springer (2014)

[23] Lenin, A., Willemson, J., Charnamord, A.: Genetic Approximations for the Failure-Free Security Games. In: Khouzani, M., Panaousis, E., Theodorakopoulos, G. (eds.): GameSec 2015, LNCS 9406, 311–321. Springer (2015)

[24] Mauw, S., Oostdijk, M.: Foundations of Attack Trees. In: Won, D., Kim, S. (eds.): ICISC 2005, LNCS 3935, 186–198. Springer (2005)

[25] Papadimitriou, C.H.: Computational complexity. Addison-Wesley (1994)

[26] Pardue, H., Landry, J., Yasinsac, A.: A Risk Assessment Model for Voting Systems using Threat Trees and Monte Carlo Simulation. In: RE-VOTE 2009, 55–60. IEEE (2009)

[27] Phillips, C., Swiler, L.P.: A Graph-based System for Network-vulnerability Analysis. In: NSPW '98, 71–79. ACM (1998)

[28] Pieters, W., Hadziosmanović, D., Lenin, A., Morales, A.L.M., Willemson, J.: TREsPASS: Plug-and-play attacker profiles for security risk analysis. In: 35th IEEE Symposium on Security and Privacy, IEEE Computer Society (2014)

[29] Pieters, W., van der Ven, S.H.G., Probst, C.W.: A move in the security measurement stalemate: elo-style ratings to quantify vulnerability. In: Ford, R., Zurko, M.E., Herley, C., Whalen, T. (eds.): NSPW'12, 1–14. ACM (2012)

[30] Sallhammar, K., Knapskog, S.J., Helvik, B.E.: Building a Stochastic Model for Security and Trust Assessment Evaluation. ECRIM News. `http://q2s.ntnu.no/publications/open/2005/Mass_media/2005_sallhammar_BSM.pdf` (2005)

[31] Schumacher, M.: Security Engineering with Patterns - Origins, Theoretical Models, and New Applications. In: Goos, G., Hartmanis, J., Leeuwen, J. (eds.): LNCS 2754. Springer (2003)

[32] Schumacher, M., Roedig, U.: Security engineering with patterns. In: Henney, K., Schütz, D. (eds.): PLoP 2001. (2001)

[33] Schneier, B.: Attack Trees: Modeling security threats. Dr. Dobb's Journal **24**(12) 21–29 (1999)

[34] Scott, J.: Rational Choice Theory. In Browning, G., Halcli, A., Webster, F. (eds.): Understanding Contemporary Society: Theories of the Present. SAGE Publications Ltd 126–139 (2000)

[35] Sonnenreich, W., Albanese, J., Stout, B.: Return On Security Investment (ROSI) - A Practical Quantitative Modell. Journal of Research and Practice in Information Technology **38**(1) (2006)

[36] Srinivas, M., Patnaik, L.M.: Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Transactions on Systems, Man, and Cybernetics **24**(4) 656–667 (1994)

[37] Tipton, H., Baker, P.: Official (ISC)2 guide to the CISSP CBK. In: Official (ISC)2 guide to the CISSP CBK. (2010)

[38] Verendel, V.: Quantified security is a weak hypothesis: a critical survey of results and assumptions. In: Somayaji, A., Ford, R. (eds.): NSPW '09, 37–50. ACM (2009)

[39] Veseley, W., Goldberg, F., Roberts, N., Haasl, D.: Fault tree handbook. Technical report NUREG–0492 (1981)

[40] Weiss, J.D.: A System Security Engineering Process. In: 14th National Computer Security Conference, 572–581. NIST/NCSC (1991)

Abstract

The thesis researches methods for analyzing the security of organizations against targeted profit-oriented attacks. Compared to the previous models and methods, the thesis proposes a more reliable model and more efficient computational methods for obtaining upper bounds of the expected outcome of rational attackers, considering the so-called fully adaptive adversarial setting in which the attacker is able to run atomic sub-attacks in arbitrary order.

It is shown that in the new model, there always exist optimal attacking strategies that are non-adaptive in the sense that the next tried sub-attack does not depend on the results of the previously tried attacks. Due to the simplicity of the optimal strategies in the new model, upper bounds for adversarial utility can be found in near-linear time, and hence the new model is especially suitable for creating simple and reliable engineering methods in information security.

In the result of this research several models and computational methods to assess the attacker expected outcome have been created. The thesis is characterized by the attempt to avoid false-positiveness – states when the organization is protected against rational attacks in the model, but in practice profitable ways to attack such organization still may exist.

Thus the thesis contributes to the creation of correct and reliable computational methods for security engineering. Similarly, in civil engineering it is not required to find the precise values of stress under which the construction breaks, but to create a building with sufficient tolerance margin, that could be easily computed by engineers and technicians.

Kokkuvõte

Doktoritöös uuritakse meetodeid, millega saab analüüsida organisatsioonide turvalisust eesmärgipõhiste kasule orienteeritud rünnete vastu. Töö põhitulemus on senistest usaldusväärsem mudel ja senistest efektiivsemad arvutusmeetodid ratsionaalsete ründajate ootetulu ülemtõkete leidmiseks nn täisadaptiivses ründemudelis, kus ründaja võib üritada atomaarseid alamründeid mis tahes järjestuses. Näidatakse, et uues mudelis leiduvad alati optimaalsed ründestrateegiad, mis on mitte-adaptiivsed selles mõttes, et järgmisena üritatav alamrünne ei sõltu eelnevalt üritatud alamrünnete tulemustest. Optimaalsete strateegiate lihtsuse tõttu leiduvad uues mudelis lineaarsele lähedase keerukusega ootetulu ülemtõkke leidmise algoritmid ja seetõttu on uus mudel eriti sobilik lihtsate ja usaldusväärsete inseneriarvutuste loomiseks infoturbes.

Uurimistöö käigus on loodud mitmed mudelid ja arvutusmeetodid rünajate ootetulu hindamiseks. Uurimistööd iseloomustab püüe vältida valepositiivsust, s.t. olukorda, kus mudeli järgi on süsteem ratsionaalsete ründajate eest kaitstud, kuid praktikas võib siiski leiduda ründajatele kasulik ründamisviis.

Töö põhisuund on seega kaasa aidata korrektsete ja usaldusväärsete inseneriarvutuste loomisele infoturbes. Ka näiteks tsiviilehituses ei ole vaja kunagi leida maja kokkukukkumise täpseid piire, vaid pigem ehitada maja mõistliku tugevusvaruga, mis on lihtsasti atvutatav inseneridele ja tehnikutele.

Acknowledgements

I would like to show appreciation to everybody who helped me with advice and support during my Ph.D. studies.

In particular, I would like to express deep gratitude to my supervisor Prof. Dr. Ahto Buldas for helping me in doing the first steps, for his wise advice and support, and for encouraging me to finish this thesis.

I would like to express my appreciation to Cybernetica AS for providing an outstanding environment for productive research and study.

Special thanks to my colleagues Dr. Jan Willemson for his collaboration during the research, as well as to Dr. Peeter Laud for supporting my research and providing outstanding possibilities for it.

Finally, I would like to thank Prof. Dr. Rein Kuusik, the head of the department of informatics in Tallinn University of Technology for his warm and supportive attitude towards young researchers.

Appendices

Appendix A

Curriculum Vitae

Curriculum Vitae

Aleksandr Lenin

Personal Data

place and date of Birth:	Tallinn, Estonia | 23 March 1986
phone:	+372 56315469
email:	aleksandr.lenin@cyber.ee

Career

feb 2012 – Current	Researcher Cybernetica AS
sep 2012 – Current	Visiting lecturer Tallinn University of Technology, Faculty of Information Technology, Department of Informatics
sep 2012 – Current	Visiting lecturer Tallinn University of Technology, Faculty of Information Technology, Department of Computer Science
sep 2011 – June 2011	Teaching Assistant Tallinn University of Technology, Faculty of Information Technology, Department of Computer Engineering

Languages

Russian:	Mothertongue
English:	Fluent
Estonian:	Fluent
German:	Basic Knowledge
Japanese:	Basic Knowledge

Education

SEP 2013	PhD student Thesis: *Reliable and Efficient Determination of the Likelihood of Rational Attacks* Supervisor: Prof. Dr. Ahto Buldas, Chair of Information Security, TUT Institutions: Tallinn University of Technology
DEC 2012	Master of Science in Engineering (*cum laude*, Cybersecurity) Thesis: *New Efficient Utility Upper Bounds for the Fully Adaptive Model of Attack Trees* Supervisor: Prof. Dr. Ahto Buldas, Chair of Information Security, TUT Institutions: Tallinn University of Technology, University of Tartu
DEC 2011	Bachelor of Science in Engineering (Information Technology) Thesis: *Solving Combinatorial Problems using Search Trees* Supervisor: Aleksander Sudnitsõn, Associate Professor, TUT Institution: Tallinn University of Technology

Current Grants and Projects

NOV 2014	National Science Foundation (NSF) Award
JAN 2014	Theory and Practice of Secure Intergovernmental, -organizational, and -personal Information Exchange (IUT27-1)
FEB 2013	Technology-supported Risk Estimation by Predictive Assessment of Socio-technical Security (ICT-318003, TRE$_S$PASS)

Thesis under Supervision

JUN 2016	Bachelor of Science in Engineering (Information Technology) Student: Anton Charnamord Thesis: *Efficient Approximation Algorithms in the Improved Failure-Free Model* Institution: Tallinn University of Technology
JUN 2016	Bachelor of Science in Engineering (Information Technology) Student: Zuljen Dedegkajev Thesis: *Item Response Theory based Approach to Quantitative Metrics of Operational Security Risks* Institution: Tallinn University of Technology

Thesis Supervised

JUN 2015	Bachelor of Science in Engineering (Information Technology) Student: Vlada Plaskovitskaja Thesis: *Integration of the TRE$_S$PASS Toolset and the* iske *Tool* Institution: Tallinn University of Technology
JAN 2015	Bachelor of Science in Engineering (Information Technology) Student: Jelena Plehhanova Thesis: *Assessment of integration possibilities of the TRE$_S$PASS toolset into the* iske *Tool* Institution: Tallinn University of Technology
JUN 2014	Master of Science in Engineering (Cybersecurity) Student: Mai Kraft Thesis: *Performance Analysis of Attacker Profiling in Quantitative Security Risk Assessment* Institution: Tallinn University of Technology
JAN 2014	Master of Science in Engineering (Cybersecurity) Student: Dyan Permata Sari Thesis: *Attacker Profiling in Quantitative Security Assessment* Institution: Tallinn University of Technology Co-supervision with Prof. Dr. Ahto Buldas

Publications

1. Buldas, A., Lenin, A.: New efficient utility upper bounds for the fully adaptive model of attack trees. In: Das, S.K., Nita-Rotaru, C., Kantarcioglu, M. (eds.): GameSec 2013. LNCS 8252, 192–205. Springer (2013)

2. Lenin, A., Willemson, J., Sari, D.: Attacker profiling in quantitative security assessment based on attack trees. In Fischer-Hübner, S., Bernsmed, K. (eds.): NordSec 2014, LNCS 8788, 199–212. Springer (2014)

3. Lenin, A., Buldas, A.: Limiting adversarial budget in quantitative security assessment. In: Poovendran, R., Saad, W. (eds.): GameSec 2014, LNCS 8840, 155–174. Springer (2014)

4. Pieters, W., Hadziosmanovi , D., Lenin, A., Morales, A.L.M., Willemson, J.: TREsPASS: Plug-and-play attacker profiles for security risk analysis. In: 35th IEEE Symposium on Security and Privacy, IEEE Computer Society (2014)

5. Lenin, A., Willemson, J., Charnamord, A.: Genetic approximations for

the failure-free security games. In: Khouzani, M., Panaousis, E., Theodorakopoulos, G. (eds.): GameSec 2015, LNCS 9406. (to appear) Springer (2015)

Technical Reports

1. TRE$_S$PASS Deliverable 3.1.2: “Final Requirements for Quantitative Analysis Tools” (2014)
2. TRE$_S$PASS Deliverable 3.3.2: “Methods for Stochastic Analysis” (2014)
3. TRE$_S$PASS Deliverable 5.1.2: “Final Requirements for Process Integration” (2014)
4. TRE$_S$PASS Deliverable 5.3.2: “Best Practices for Model Creation and Sharing” (2014)
5. Gollmann, D., Herley, C., Koenig, V., Pieters, W., Sasse, M.A.: Socio-Technical Security Metrics. Dagstuhl Reports **4**(12) 1–28 (2014)

Invited Talks

DEC 2014	“Socio-Technical Security Metrics” Schloss Dagstuhl, Dagstuhl Seminar 14491
NOV 2013	“Estonian e-Government” University College of the Cayman Islands (UCCI), Grand Cayman

Appendix B

Elulookirjeldus

Elulookirjeldus

Aleksandr Lenin

Isiklikud andmed

sünniaeg:	Tallinn, Eesti | 23 March 1986
telefon:	+372 56315469
email:	aleksandr.lenin@cyber.ee

Teenistuskäik

feb 2012 – Praeguseni	Teadur Cybernetica AS
sep 2012 – Praeguseni	Külalislektor Tallinna Tehnikaülikool, Infotehnoloogia teaduskond, Informaatika instituut
sep 2012 – Praeguseni	Külalislektor Tallinna Tehnikaülikool, Infotehnoloogia teaduskond, Arvutiteaduse instituut
sep 2011 – June 2011	Assistent Tallinna Tehnikaülikool, Infotehnoloogia teaduskond, Arvutitehnika instituut

Keelteoskus

Vene:	Emakeel
Inglise:	Sujuv
Eesti:	Sujuv
Saksa:	Algteadmised
Jaapani:	Algteadmised

Haridus

sep 2013	doktorant Lõputöö teema: *Ratsionaalsete rünnete tõepära efektiivne kindlakstegemine* Juhendaja: Prof. Dr. Ahto Buldas, Infoturbe õppetooli juhataja, TTÜ Asutus: Tallinna Tehnikaülikool
dec 2012	Tehnikateaduste Magister (*cum laude*, Küberkaitse) Lõputöö teema: *Uued efektiivsed ründaja ootetulu ülemtõkked täisadaptiivses ründepuude mudelis* Juhendaja: Prof. Dr. Ahto Buldas, Infoturbe õppetooli juhataja, TTÜ Asutused: Tallinna Tehnikaülikool, Tartu Ülikool
dec 2011	Tehnikateaduste Bakalaureus (Infotehnoloogia) Lõputöö teema: *Kombinatoorikaprobleemide lahendamine otsimispuude abil* Juhendaja: Aleksander Sudnitsõn, dotsent, TTÜ Asutus: Tallinna Tehnikaülikool

Projektide ja Grantide Loetelu

nov 2014	National Science Foundation (nsf) Award
jan 2014	Riikide, organisatsioonide ja isikute vahelise turvalise infovahetuse teooria ja praktika (iut27-1)
feb 2013	Tehnoloogiline tugi riskide arvutamiseks sotsio-tehnilise turvalisuse ennetava hindamise kaudu (ict-318003, TRE$_S$PASS)

Juhendamisel Väitekirjad

jun 2016	Tehnikateaduste bakalaureus (Infotehnoloogia) Tudeng: Anton Charnamord Lõputöö teema: *Efektiivsed lähendusalgoritmid parandatud veavabas mudelis* Asutus: Tallinna Tehnikaülikool
jun 2016	Tehnikateaduste bakalaureus (Infotehnoloogia) Tudeng: Zuljen Dedegkajev Lõputöö teema: *Item Response Theory based Approach to Quantitative Metrics of Operational Security Risks* Asutus: Tallinna Tehnikaülikool

Juhendatud Väitekirjad

JUN 2015 Tehnikateaduste Bakalaureus (Infotehnoloogia)
Tudeng: Vlada Plaskovitskaja
Lõputöö teema: *TRE$_S$PASS rakendustööriistade* ISKE*sse integreerimine*
Asutus: Tallinna Tehnikaülikool

JAN 2015 Tehnikateaduste Bakalaureus (Infotehnoloogia)
Tudeng: Jelena Plehhanova
Lõputöö teema: *TRE$_S$PASS rakendustööriistade* ISKE*sse integreerimise võimaluste analüüs*
Asutus: Tallinna Tehnikaülikool

JUN 2014 Tehnikateaduste Magister (Küberkaitse)
Tudeng: Mai Kraft
Lõputöö teema: *Ründaja profileerimise jõudluse analüüs Kvantitatiivses Turvariski Analüüsis*
Asutus: Tallinna Tehnikaülikool

JAN 2014 Tehnikateaduste Magister (Küberkaitse)
Tudeng: Dyan Permata Sari
Lõputöö teema: *Ründajate profileerimine kvantitatiivses turvaanalüüsis*
Asutus: Tallinna Tehnikaülikool
Kaasjuhendamine koos Prof. Dr. Ahto Buldas

Publikatsioonid

1. Buldas, A., Lenin, A.: New efficient utility upper bounds for the fully adaptive model of attack trees. In: Das, S.K., Nita-Rotaru, C., Kantarcioglu, M. (eds.): GameSec 2013. LNCS 8252, 192–205. Springer (2013)

2. Lenin, A., Willemson, J., Sari, D.: Attacker profiling in quantitative security assessment based on attack trees. In Fischer-Hübner, S., Bernsmed, K. (eds.): NordSec 2014, LNCS 8788, 199–212. Springer (2014)

3. Lenin, A., Buldas, A.: Limiting adversarial budget in quantitative security assessment. In: Poovendran, R., Saad, W. (eds.): GameSec 2014, LNCS 8840, 155–174. Springer (2014)

4. Pieters, W., Hadziosmanovi , D., Lenin, A., Morales, A.L.M., Willemson, J.: TREsPASS: Plug-and-play attacker profiles for security risk analysis. In: 35th IEEE Symposium on Security and Privacy, IEEE Computer Society (2014)

5. Lenin, A., Willemson, J., Charnamord, A.: Genetic approximations for

the failure-free security games. In: Khouzani, M., Panaousis, E., Theodorakopoulos, G. (eds.): GameSec 2015, LNCS 9406. (to appear) Springer (2015)

Teadusaruanded

1. TRE$_S$PASS Deliverable 3.1.2: "Final Requirements for Quantitative Analysis Tools" (2014)
2. TRE$_S$PASS Deliverable 3.3.2: "Methods for Stochastic Analysis" (2014)
3. TRE$_S$PASS Deliverable 5.1.2: "Final Requirements for Process Integration" (2014)
4. TRE$_S$PASS Deliverable 5.3.2: "Best Practices for Model Creation and Sharing" (2014)
5. Gollmann, D., Herley, C., Koenig, V., Pieters, W., Sasse, M.A.: Socio-Technical Security Metrics. Dagstuhl Reports **4**(12) 1–28 (2014)

Esinemised

DEC 2014	"Socio-Technical Security Metrics" Schloss Dagstuhl, Dagstuhl Seminar 14491
NOV 2013	"Estonian e-Government" University College of the Cayman Islands (UCCI), Grand Cayman

List of Publications

The following publications are re-printed in the appendices of this dissertation:

I. Buldas, A., Lenin, A.: New efficient utility upper bounds for the fully adaptive model of attack trees. In: Das, S.K., Nita-Rotaru, C., Kantarcioglu, M. (eds.): GameSec 2013. LNCS 8252, 192–205. Springer (2013)

II. Lenin, A., Willemson, J., Sari, D.: Attacker profiling in quantitative security assessment based on attack trees. In Fischer-Hübner, S., Bernsmed, K. (eds.): NordSec 2014, LNCS 8788, 199–212. Springer (2014)

III. Lenin, A., Buldas, A.: Limiting adversarial budget in quantitative security assessment. In: Poovendran, R., Saad, W. (eds.): GameSec 2014, LNCS 8840, 155–174. Springer (2014)

IV. Pieters, W., Hadziosmanović, D., Lenin, A., Morales, A.L.M., Willemson, J.: TRE$_S$-PASS: Plug-and-play attacker profiles for security risk analysis. In: 35th IEEE Symposium on Security and Privacy, IEEE Computer Society (2014)

V. Lenin, A., Willemson, J., Charnamord, A.: Genetic Approximations for the Failure-Free Security Games. In: Khouzani, M., Panaousis, E., Theodorakopoulos, G. (eds.): GameSec 2015, LNCS 9406, 311–321. Springer (2015)

Appendix C

New Efficient Utility Upper Bounds for the Fully Adaptive Model of Attack Trees

New Efficient Utility Upper Bounds for the Fully Adaptive Model of Attack Trees

Ahto Buldas[1,2,3] and Aleksandr Lenin[1,3] ⋆

[1] Cybernetica AS
[2] Guardtime AS
[3] Tallinn University of Technology

Abstract. We present a new fully adaptive computational model for attack trees that allows attackers to repeat atomic attacks if they fail and to play on if they are caught and have to pay penalties. The new model allows safer conclusions about the security of real-life systems and is somewhat (computationally) easier to analyze. We show that in the new model optimal strategies always exist and finding the optimal strategy is (just) an NP-complete problem. We also present methods to compute adversarial utility estimation and utility upper bound approximated estimation using a bottom-up approach.

1 Introduction

Protection of information systems becomes an integral part in the deployment of technologies that operate sensitive information, the leakage of which may cause irreversible damage to affected parties. In new technology deployment, its protection and security are the concerns in the first place. It is impossible to achieve 100% level of protection. By applying various security measures one can just approach this limit. Various methods of risk assessment have been suggested, and each of them has its advantages and disadvantages. Quantitative security analysis based on attack trees has become a subject of extensive research [8, 1–6, 9].

Attack trees may be used for visualization purposes only, but also for computing *adversarial utility*, i.e. attacking the system can be modeled as an economic single-player game played by the attacker. It is assumed that the attacker behaves rationally—attacks only if the attack game is beneficial for him. Such a *rational attacker paradigm* was first introduced by Buldas *et al.* [1]. In order to estimate the adversarial utility, their model used computational rules for AND and OR nodes to compute a list of parameters for every node based on the parameters of its successor nodes. Buldas and Stepanenko [3] introduced the

⋆ Part of the research leading to these results has received funding from the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement ICT-318003 (TREsPASS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.

so-called *fully adaptive model* where adversaries are allowed to try atomic attacks in an arbitrary order, depending on the results of the previous trials. They also introduced the *upper bound ideology* by pointing out that in order to verify the security of the system, it is not necessary to compute the exact adversarial utility but only *upper bounds*—if adversarial utility has a negative upper bound in the fully adaptive model, it is safe to conclude that there are no beneficial ways of attacking the system, assuming that all reasonable atomic attacks are captured in the attack tree. A similar approach is used in civil engineering—it is more practical to know an upper bound of the stress value that will definitely not break a construction than the precise stress value at which the construction breaks. In [3] two ways were introduced to compute the upper bounds: (1) simplified computational rules (for AND and OR nodes); and (2) assuming more powerful adversaries in a way that simplifies the computational model, for example, in their *infinite repetition model* the attacker is allowed to repeat atomic attacks (any number of times) if they fail.

Motivation of this work: Even the fully adaptive model of Buldas and Stepanenko [3] does not completely follow their upper bound ideology. Mostly the atomic attacks are associated with criminal behavior and hence in the attack tree models [1, 4, 5, 9, 3] atomic attacks are associated with *penalties* that the attacker has to pay if he is caught. In the model [3] an additional restriction is introduced—the attacker is not able to play on after getting caught. As this seems not to be true in all real-life cases, either the penalties in their model contain the potential future profits of attackers (and hence be larger than they are in real life) or the model does not give reliable upper bounds. Moreover, it seems that such a *game over* assumption actually makes the computational model more complex.

The aim of this work: We present a new fully adaptive computational model for attack trees that allows the adversary to repeat atomic attacks if they fail and to play on if he is caught. We show that such a model will be somewhat easier to analyze. For example, in the case of conjunctive composition $\mathcal{X}_1 \wedge \ldots \wedge \mathcal{X}_n$ of atomic attacks the order in which they are tried by the adversary is unimportant.

Summary of results: We show (Sec. 3) that in the new model optimal strategies always exist and they are in the form of directed single-branched BDDs with self-loops. We introduce methods to compute a precise estimation of the adversarial utility and an approximated estimation of the utility upper bound using a bottom-up *utility propagation* approach. We also show (Sec. 4) that solving the attack game in the new model is an NP-complete problem. We also present efficient methods to compute the lower bound for expenses (Sec. 5).

2 Definitions and Related Work

In this section we will formally define some common terms and definitions within the current model which will be used further throughout the paper.

2.1 Definitions

Definition 1 (Derived function). *If $\mathcal{F}(x_1, \ldots, x_m)$ is a Boolean function and $v \in \{0,1\}$, then by the derived Boolean function $\mathcal{F}|_{x_j=v}$ we mean the function $\mathcal{F}(x_1, \ldots, x_{j-1}, v, x_{j+1}, \ldots, x_m)$ derived from $\mathcal{F}$ by the assignment $x_j := v$.*

Definition 2 (Constant functions). *By* **1** *we mean a Boolean function that is identically true and by* **0** *we mean a Boolean function that is identically false.*

Definition 3 (Min-term). *By a min-term of a Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ we mean a conjunction of variables $x_{i_1} \wedge x_{i_2} \wedge \ldots \wedge x_{i_k}$ such that $x_{i_1} \wedge x_{i_2} \wedge \ldots \wedge x_{i_k} \Rightarrow \mathcal{F}(x_1, \ldots, x_m)$ is a tautology.*

Definition 4 (Critical min-term). *A min-term $x_1 \wedge \ldots \wedge x_k$ of $\mathcal{F}$ is critical if none of the sub-terms $x_1, \ldots, x_{j-1}, x_{j+1}, \ldots, x_m$ is a min-term of $\mathcal{F}$.*

Definition 5 (Satisfiability game). *By a satisfiability game we mean a single-player game in which the player's goal is to satisfy a monotone Boolean function $\mathcal{F}(x_1, x_2, \ldots, x_k)$ by picking variables x_i one at a time and assigning $x_i = 1$. Each time the player picks the variable x_i he pays some amount of expenses $\mathcal{E}_i$, which is modeled as a random variable. With a certain probability p_i the move x_i succeeds. Function $\mathcal{F}$ representing the current game instance is transformed to its derived form $\mathcal{F}|_{x_i=1}$ and the next game iteration starts. The game ends when the condition $\mathcal{F} \equiv 1$ is satisfied and the player wins the prize $\mathcal{P} \in \mathcal{R}$, or when the condition $\mathcal{F} \equiv 0$ is satisfied, meaning the loss of the game, or when the player stops playing. With a probability $1 - p_i$ the move x_i fails. The player may end up in a different game instance represented by the derived Boolean function $\mathcal{F}|_{x_i \equiv 0}$ in the case of a game without move repetitions, and may end up in the very same instance of the game $\mathcal{F}$ in the case of a game with repetitions. Under certain conditions with a certain probability the game may end up in a forced failure state, i.e. if the player is caught and this implies that he cannot continue playing, i.e. according to the Buldas-Stepanenko model [3]. The rules of the game are model-specific and may vary from model to model. Thus we can define three common types of games:*

1. *SAT Game Without Repetitions - the type of a game where an adversary can perform a move only once.*
2. *SAT Game With Repetitions - the type of a game where an adversary can re-run failed moves again an arbitrary number of times.*
3. *Failure-Free SAT Game - the type of a game in which all success probabilities are equal to 1. It can be shown that any game with repetitions is equivalent to a failure-free game (Thm. 5).*

Definition 6 (Line of a game). *By a line of a satisfiability game we mean a sequence of assignments $\lambda = \langle x_{j_1} = v_1, \ldots, x_{j_k} = v_k \rangle$ (where $v_j \in \{0,1\}$) that represent the player's moves, and possibly some auxiliary information. We say that λ is a* ***winning line*** *if the Boolean formula $x_{i_1} \wedge \ldots \wedge x_{i_k} \Rightarrow \mathcal{F}(x_1, \ldots, x_n)$ is a tautology, where $\mathcal{F}$ is a Boolean function of the satisfiability game.*

Definition 7 (Strategy). *By a strategy $\mathcal{S}$ for a game $\mathcal{G}$ we mean a rule that for any line λ of $\mathcal{G}$ either suggests the next move $x_{j_{k+1}}$ or decides to give up.*

Strategies can be represented graphically as binary decision diagrams (BDDs).

Definition 8 (Line of a strategy). *A line of a strategy $\mathcal{S}$ for a game $\mathcal{G}$ is the smallest set $\mathcal{L}$ of lines of $\mathcal{G}$ such that (1) $\langle\rangle \in \mathcal{L}$ and (2) if $\lambda \in \mathcal{L}$, and $\mathcal{S}$ suggests x_j as the next move to try, then $\langle \lambda, x_j = 0\rangle \in \mathcal{L}$ and $\langle \lambda, x_j = 1\rangle \in \mathcal{L}$.*

Definition 9 (Branch). *A branch β of a strategy $\mathcal{S}$ for a game $\mathcal{G}$ is a line λ of $\mathcal{S}$ for which $\mathcal{S}$ does not suggest the next move. By $\mathcal{B}_\mathcal{S}$ we denote the set of all branches of $\mathcal{S}$.*

For example, all winning lines of $\mathcal{S}$ are branches.

Definition 10 (Expenses of a branch). *If $\beta = \langle x_{i_1=v_1}, \ldots, x_{i_k=v_k}\rangle$ is a branch of a strategy $\mathcal{S}$ for $\mathcal{G}$, then by expenses $\epsilon_\mathcal{G}(\mathcal{S}, \beta)$ of β we mean the sum $\overline{\mathcal{E}}_{i_1} + \ldots + \overline{\mathcal{E}}_{i_k}$ where by $\overline{\mathcal{E}}_{i_j}$ we mean the mathematical expectation of $\mathcal{E}_{i_j}$.*

Definition 11 (Prize of a branch). *The prize $\mathcal{P}_\mathcal{G}(\mathcal{S}, \beta)$ of a branch β of a strategy $\mathcal{S}$ is $\mathcal{P}$ if β is a winning branch, and 0 otherwise.*

Definition 12 (Utility of a strategy). *By the utility of a strategy $\mathcal{S}$ in a game $\mathcal{G}$ we mean the sum: $\mathcal{U}(\mathcal{G}, \mathcal{S}) = \sum\limits_{\beta\in\mathcal{B}_\mathcal{S}} Pr(\beta) \cdot [\mathcal{P}_\mathcal{G}(\mathcal{S}, \beta) - \epsilon_\mathcal{G}(\mathcal{S}, \beta)]$. For the empty strategy $\mathcal{U}(\mathcal{G}, \emptyset) = 0$.*

Definition 13 (Prize and Expenses of a strategy). *By the expenses $\mathcal{E}(\mathcal{G}, \mathcal{S})$ of a strategy $\mathcal{S}$ we mean the sum $\sum\limits_{\beta\in\mathcal{B}_\mathcal{S}} Pr(\beta) \cdot \epsilon_\mathcal{G}(\mathcal{S}, \beta)$. The prize $\mathcal{P}(\mathcal{G}, \mathcal{S})$ of $\mathcal{S}$ is $\sum\limits_{\beta\in\mathcal{B}_\mathcal{S}} Pr(\beta) \cdot \mathcal{P}_\mathcal{G}(\mathcal{S}, \beta)$.*

It is easy to see that $\mathcal{U}(\mathcal{G}, \mathcal{S}) = \mathcal{P}(\mathcal{G}, \mathcal{S}) - \mathcal{E}(\mathcal{G}, \mathcal{S})$.

Definition 14 (Utility of a satisfiability game). *The utility of a SAT game $\mathcal{G}$ is the limit $\mathcal{U}(\mathcal{G}) = \sup\limits_{\mathcal{S}} \mathcal{U}(\mathcal{G}, \mathcal{S})$ that exists due to the bound $\mathcal{U}(\mathcal{G}, \mathcal{S}) \leqslant \mathcal{P}$.*

Definition 15 (Optimal strategy). *By an optimal strategy for a game $\mathcal{G}$ we mean a strategy $\mathcal{S}$ for which $\mathcal{U}(\mathcal{G}) = \mathcal{U}(\mathcal{G}, \mathcal{S})$.*

It can be shown that for satisfiability games optimal strategies always exist.

2.2 Related Work

In the *fully-adaptive model* introduced by Buldas and Stepanenko [3] the attacker does not use a specific attack suite or a specific ordering, but picks the next atomic attack arbitrarily based on the results of the previously tried atomic attacks. Their so-called *infinite repetition model* assumes that the adversary has a possibility to re-run failed attacks again immediately or later after trying some other atomic attacks. The so-called *failure-free model* [3] assumes all success probabilities are equal to 1, meaning that the player will achieve his goal anyway, but with either positive or negative utility. Due to the failure-free model concept a strategy can be represented as a set of moves $\{\mathcal{X}_1, \dots, \mathcal{X}_k\}$ and the order in which those moves will be launched is not a concern.

The model [3] assumes that the stops playing immediately upon attack detection that is an unnatural restriction placed on the adversarial actions since in reality countermeasures cannot be applied immediately and usually the adversary has some time within which he may continue attacking and may achieve his goal. Therefore it would be natural to expect that the attacker does not stop his actions if his attack is detected by the defensive security measures deployed on the analyzed system. Apart from that, Buldas-Stepanenko model is arithmetically more complex due to the force-failure states and an optimal strategy depends on the order of atomic attacks.

3 New Model

In the new model the adversary does not stop when launched attacks are detected and continues attacking until he achieves his goal. The new model is similar to the parallel model by Jürgenson and Willemson [4–6], except that it applies the infinite repetition model concept and introduces new methods that allow us to compute the adversarial utility upper bounds. Due to the slightly simplified rules of the game the new model became more simple and manageable than Jürgenson-Willemson and Buldas-Stepanenko models, thus easier to use and analyze. The new model allowed us to elaborate efficient upper bound computation methods that run in time linear in the size of the attack tree.

Lemma 1. *For every repeatable satisfiability game $\mathcal{G}$ with $\mathcal{U}(\mathcal{G}) > 0$ there is x_j such that $\sup\limits_{\mathcal{S} \in \mathbb{S}_{x_j}} \mathcal{U}(\mathcal{G}, \mathcal{S}) = \mathcal{U}(\mathcal{G})$, where $\mathbb{S}_{x_j}$ is the set of all non-empty strategies with x_j as the first move.*

Proof. As every $\mathcal{S} \neq \emptyset$ has the first move x_i, we have $\mathcal{U}(\mathcal{G}) = \sup\limits_{\mathcal{S}} \mathcal{U}(\mathcal{G}, \mathcal{S}) = \max_i \sup\limits_{\mathcal{S} \in \mathbb{S}_{x_i}} \mathcal{U}(\mathcal{G}, \mathcal{S})$, and hence there is x_j such that $\mathcal{U}(\mathcal{G}) = \sup\limits_{\mathcal{S} \in \mathbb{S}_{x_j}} \mathcal{U}(\mathcal{G}, \mathcal{S})$. □

Lemma 2. *For every repeatable satisfiability game G and for every atomic variable x_j: $\sup\limits_{\mathcal{S} \in \mathbb{S}_{x_j}} \mathcal{U}(\mathcal{G}, \mathcal{S}) = -\mathcal{E}_j + p_j \mathcal{U}(\mathcal{G}|_{x_j=1}) + (1 - p_j)\mathcal{U}(\mathcal{G})$.*

Proof. This is because the part $\mathcal{S}'$ of $\mathcal{S}$ for playing $\mathcal{G}|_{x_j=1}$ and the part $\mathcal{S}''$ of $\mathcal{S}$ for playing $\mathcal{G}$ after an unsuccessful trial of x_j can be chosen independently. □

Theorem 1. *Repeatable satisfiability games have optimal strategies.*

Proof. If $\mathcal{U}(\mathcal{G}) = 0$, then $\mathcal{S} = \emptyset$ is optimal. For the case $\mathcal{U}(\mathcal{G}) > 0$ we use induction on the number m of atomic variables. If $m = 0$, there are no moves and $\emptyset$ is the only possible strategy and is optimal by definition. In case $m > 0$ and supposing that every repeatable satisfiability game with $m-1$ atomic variables has an optimal strategy, by *Lemma 1*, there is x_j such that $\mathcal{U}(\mathcal{G}) = \sup\limits_{\mathcal{S}\in\mathbb{S}_{x_j}} \mathcal{U}(\mathcal{G},\mathcal{S})$. Let $\mathcal{S}_0$ be the strategy that repeats x_j until x_j succeeds and then behaves like an optimal strategy for $\mathcal{G}|_{x_j=1}$ (a game with $m-1$ atomic variables). Utility of $\mathcal{S}_0$ is $\mathcal{U}(\mathcal{G},\mathcal{S}_0) = -\frac{\mathcal{E}_j}{p_j} + \mathcal{U}(\mathcal{G}|_{x_j=1})$. On the other hand, by *Lemma 2* we have $\mathcal{U}(\mathcal{G}) = \sup\limits_{\mathcal{S}\in\mathbb{S}_{x_j}} \mathcal{U}(\mathcal{G},\mathcal{S}) = -\mathcal{E}_j + p_j\,\mathcal{U}(\mathcal{G}|_{x_j=1}) + (1-p_j)\,\mathcal{U}(\mathcal{G})$, that implies $\mathcal{U}(\mathcal{G}) = -\frac{\mathcal{E}_j}{p_j} + \mathcal{U}(\mathcal{G}|_{x_j=1}) = \mathcal{U}(\mathcal{G},\mathcal{S}_0)$ and hence $\mathcal{S}_0$ is optimal. □

Corollary 1. *In every repeatable satisfiability game there exist optimal strategies in the form of directed single-branched BDDs with self-loops.*

Proof. Let $\mathcal{S}$ be an optimal strategy for $\mathcal{G}$ and $\mathcal{X}_{j_1}$ be the first move suggested by $\mathcal{S}$. In case of a failure, $\mathcal{X}_{j_1}$ remains the best move and hence $\mathcal{S}$ has a self-loop at $\mathcal{X}_{j_1}$. In case of success, the Boolean function of the game reduces to $\mathcal{F}|_{\mathcal{X}_{j_1}=1}$. Let $\mathcal{X}_{j_2}$ be the next move suggested by $\mathcal{S}$. Similarly, we conclude that there is a self-loop at $\mathcal{X}_{j_2}$ in case of a failure, and so on. This leads to the BDD in Fig. 1. □

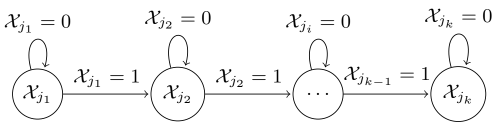


Fig. 1. A strategy in the form of a directed single-branched BDD with self-loops.

Theorem 2. *If $\mathcal{S}$ is a strategy in the form of a self-looped BDD (Fig. 1), then*

$$\mathcal{E}(\mathcal{G},\mathcal{S}) = \frac{\overline{\mathcal{E}}_{j_1}}{p_{j_1}} + \ldots + \frac{\overline{\mathcal{E}}_{j_k}}{p_{j_k}} \ .$$

Proof. The probability that a move succeeds at the n-th try is $p(1-p)^{n-1}$ and the average expenses are $\overline{\mathcal{E}}(1-p)^{n-1}$ and hence the total success probability is $p \cdot \sum\limits_{n=1}^{\infty}(1-p)^{n-1} = 1$ and the average expenses are $\overline{\mathcal{E}} \cdot \sum\limits_{n=1}^{\infty}(1-p)^{n-1} = \frac{\overline{\mathcal{E}}}{p}$. □

It is obvious that in the new model $\mathcal{P}(\mathcal{G},\mathcal{S}) \in \{0, \mathcal{P}\}$.

Definition 16 (Winning strategy). *A strategy $\mathcal{S}$ is winning if $\mathcal{P}(\mathcal{G},\mathcal{S}) = \mathcal{P}$.*

Definition 17 (Expenses of a game). *By the average expenses of a game $\mathcal{G}$ we mean $\mathcal{E}(\mathcal{G}) = \inf\limits_{\mathcal{S}} \mathcal{E}(\mathcal{G},\mathcal{S})$, where $\mathcal{S}$ varies over all winning strategies.*

Theorem 3. *For any satisfiability game $\mathcal{G}$: $\mathcal{U}(\mathcal{G}) = \max\{0, \mathcal{P} - \mathcal{E}(\mathcal{G})\}$.*

Proof. Let $\mathcal{S}$ be an optimal strategy for $\mathcal{G}$. If $\mathcal{S}$ is not winning, $\mathcal{U}(\mathcal{G},\mathcal{S}) = -\mathcal{E}(\mathcal{G},\mathcal{S}) \leqslant 0$ and hence $\emptyset$ is optimal and hence $\mathcal{U}(\mathcal{G}) = 0$. If $\mathcal{S}$ is winning then $\mathcal{U}(\mathcal{G}) = \sup\limits_{\mathcal{S}} \mathcal{U}(\mathcal{G},\mathcal{S}) = \mathcal{P} - \inf\limits_{\mathcal{S}} \mathcal{E}(\mathcal{G},\mathcal{S}) = \mathcal{P} - \mathcal{E}(\mathcal{G})$. □

Due to the features of the new model, in order to compute the utility it is sufficient to compute the expenses, that allows us use the expenses propagation technique introduced below. Moreover, we will show that solving a satisfiability game in the new model is equivalent to solving a weighted monotone satisfiability problem.

3.1 Precise Utility Computation

The algorithm described in [3] is good because it is independent of the Boolean circuit structure and only depends on the Boolean function of the game. It is described formally as the following recursive relation:

$$\mathcal{U}(\mathcal{G}) = \max\left\{ 0,\ -\frac{\overline{\mathcal{E}}_i}{p_i} + \mathcal{U}(\mathcal{G}|_{x_i=1}),\ \mathcal{U}(\mathcal{G}|_{x_i=0}) \right\} \ , \tag{1}$$

with initial conditions $\mathcal{U}(\mathbf{1}) = \mathcal{P}$ and $\mathcal{U}(\mathbf{0}) = 0$, and where x_i is any variable that $\mathcal{G}$ contains. The algorithm allows us to compute the precise adversarial utility value in time exponential in the size of the game. The computational complexity of the algorithm is $\mathcal{O}(2^n)$.

3.2 Utility Upper Bound Estimation Using Utility Propagation

The model of [3] uses attack tree representation of the Boolean formula for utility propagation using the following inequalities:

$$\begin{aligned}\mathcal{U}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) &\leqslant \min\{\mathcal{U}(\mathcal{G}_1), \ldots, \mathcal{U}(\mathcal{G}_k)\} \ , \\ \mathcal{U}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k) &\leqslant \mathcal{U}(\mathcal{G}_1) + \ldots + \mathcal{U}(\mathcal{G}_k) \ .\end{aligned}$$

Firstly, we can show that even in the model of [3] we can actually use more precise folmula (2). Secondly, in the new model we can use expenses propagation approach that turns out to be even more precise.

Theorem 4. *In the model of [3] and in the new model:*

$$\mathcal{U}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k) = \max\{\mathcal{U}(\mathcal{G}_1), \ldots, \mathcal{U}(\mathcal{G}_k)\} \ . \tag{2}$$

The proof is based on the fact that optimal strategies can be represented by critical min-terms of $\mathcal{F}$.

Proof. Let $\mathcal{S}$ be the optimal strategy for the game $\mathcal{G} = \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$. According to *Thm. 8* in [3], $\mathcal{S}$ can be represented as a critical min-term $\mathcal{X}_{j_1} \wedge \ldots \wedge \mathcal{X}_{j_m}$ of the Boolean function of $\mathcal{G} = \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$, this means that there exists such j that $\mathcal{X}_{j_1} \wedge \ldots \wedge \mathcal{X}_{j_m}$ is a min-term of $\mathcal{G}_j$. Hence $\mathcal{X}_{j_1} \wedge \ldots \wedge \mathcal{X}_{j_m} \Rightarrow \mathcal{G}_j$ is a tautology. This means that

$$\mathcal{U}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k) = \mathcal{U}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k, \mathcal{S}) \leqslant \mathcal{U}(\mathcal{G}_j) \leqslant \max\{\mathcal{U}(\mathcal{G}_1), \ldots, \mathcal{U}(\mathcal{G}_k)\} \ .$$

On the other hand, for any j let $\mathcal{S}_j$ be the optimal strategy of $\mathcal{G}_j$. As $\mathcal{G}_j \Rightarrow \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$ is a tautology, $\mathcal{U}(\mathcal{G}_j) = \mathcal{U}(\mathcal{G}_j, \mathcal{S}_j) \leqslant \mathcal{U}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k)$. As j was arbitrary, this implies $\max\{\mathcal{U}(\mathcal{G}_1), \ldots, \mathcal{U}(\mathcal{G}_k)\} \leqslant \mathcal{U}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k)$. Combining these two inequalities we reach equation (2). □

Algorithm 3.1 utilizes the conjunctive and disjunctive bottom-up adversarial utility propagation rules in every game instance starting from the atomic moves and ending up in the root instance of the game. The algorithm allows us to compute the adversarial utility upper bound in time linear in the size of the game, thus complexity is $\mathcal{O}(n)$.

ALGORITHM 3.1: Iterated utility propagation in conjunctive/disjunctive game instances

```
Input: Satisfiability game instance $\mathcal{G}$
Output: Utility upper bound (real number)
1 Procedure ComputeUtilityUpperBound ($\mathcal{G}$)
2 if $m$ is an instance of a conjunctive game then
3     /* $\mathcal{G}_1, \ldots, \mathcal{G}_k$ are the sub-games of game $m$ */
4     return min{ $\mathcal{U}(\mathcal{G}_1), \ldots, \mathcal{U}(\mathcal{G}_k)$ }
5 else if $m$ is an instance of a disjunctive game then
6     /* $\mathcal{G}_1, \ldots, \mathcal{G}_k$ are the sub-games of game $m$ */
7     return max{ $\mathcal{U}(\mathcal{G}_1), \ldots, \mathcal{U}(\mathcal{G}_k)$ }
8 else $m$ is leaf
9     return $\mathcal{U}(m)$
```

4 Computational Complexity of the New Model

Definition 18 (Weighted Monotone Satisfiability /WMSAT/). *Given a threshold value $\mathcal{P}$ and a monotone Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ with corresponding weights $w(x_i) = w_i$ decide whether there is a satisfying assignment $\mathcal{A}$ with a total weight $w(\mathcal{A}) < \mathcal{P}$.*

Theorem 5. *In the new model, the problem of deciding whether $\mathcal{U}(\mathcal{G}) > 0$ is equivalent to the weighted monotone satisfiability problem with the same Boolean function as in $\mathcal{G}$ and with weights of the input variables x_i defined by $w_i = \frac{\mathcal{E}_i}{p_i}$ with threshold value $\mathcal{P}$.*

Proof. If $\mathcal{S}$ is an optimal strategy in the *infinite repetition model* and is a single-branched BDD with self-loops with nodes $\mathcal{X}_{i_1}, \ldots, \mathcal{X}_{i_k}$, then the assignment $\mathcal{A} = \langle x_{i_1} = \ldots = x_{i_k} = 1 \rangle$ satisfies the Boolean function of the game and its total weight is $w(\mathcal{A}) = \frac{\overline{\mathcal{E}}_{i_1}}{p_{i_1}} + \ldots + \frac{\overline{\mathcal{E}}_{i_k}}{p_{i_k}}$. If $\mathcal{U}(\mathcal{G}) > 0$, then

$$0 < \mathcal{U}(\mathcal{G}) = \mathcal{U}(\mathcal{G}, \mathcal{S}) = \mathcal{P} - \frac{\overline{\mathcal{E}}_{i_1}}{p_{i_1}} - \ldots - \frac{\overline{\mathcal{E}}_{i_k}}{p_{i_k}} = \mathcal{P} - w(\mathcal{A}) \ .$$

Thus, $w(\mathcal{A}) < \mathcal{P}$. If there is an assignment $\mathcal{A} = \langle x_{i_1} = \ldots = x_{i_k} = 1 \rangle$ in the WMSAT model with a total weight $w(\mathcal{A}) = \frac{\overline{\mathcal{E}}_{i_1}}{p_{i_1}} + \ldots + \frac{\overline{\mathcal{E}}_{i_k}}{p_{i_k}} < \mathcal{P}$, then the strategy depicted in Fig. 1 has the utility

$$\mathcal{U}(\mathcal{G}, \mathcal{S}) = \mathcal{P} - \frac{\overline{\mathcal{E}}_{i_1}}{p_{i_1}} - \ldots - \frac{\overline{\mathcal{E}}_{i_k}}{p_{i_k}} > 0 \ ,$$

and hence $\mathcal{U}(\mathcal{G}) \geqslant \mathcal{U}(\mathcal{G}, \mathcal{S}) > 0$. □

The parameter $\frac{\overline{\mathcal{E}}_i}{p_i}$, the cost-success ratio, is similar to the time-success ratio parameter used in cryptography [7]. This parameter can be estimated more precisely and is measurable in monetary units, as opposed to the respective probability and expenses parameters in the existing models.

Theorem 6. *The Weighted Monotone Satisfiability Problem is NP-complete.*

Proof. We will show that the Vertex Cover problem can be polynomially reduced to the WMSAT problem. Let $\mathcal{G}$ be the graph with a vertex set $\{v_1, \ldots, v_m\}$. We define a Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ as follows. For each edge (v_i, v_j) of $\mathcal{G}$ we define the clause $\mathcal{C}_{i_j} = x_i \vee x_j$. The Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ is defined as the conjunction of all $\mathcal{C}_{i_j}$ such that (v_i, v_j) is an edge of $\mathcal{G}$. Let the weight w_i of each x_i be equal to 1.

It is obvious that $\mathcal{G}$ has a vertex cover $\mathcal{S}$ of size $|\mathcal{S}| < \mathcal{P}$ iff the monotone Boolean function $\mathcal{F}(x_1, \ldots, x_m)$ has a satisfying assignment with a total weight less than $\mathcal{P}$. □

5 Efficient Computation of Expenses Lower Bounds

This section presents some examples of computing the adversarial expenses lower bound using expenses propagation.

5.1 Expenses Propagation

Let $\mathcal{G} = \mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k$ be a disjunctive game. For the disjunctive game to succeed at least one of its sub-games needs to be tried and successfully completed. We need to choose one single sub-game which is the cheapest one. Therefore, the utility upper bound of the game $\mathcal{G}$ may be computed using the method that for each sub-game computes utility estimation $\mathcal{U}(\mathcal{G}_i)$:

1. Find the cheapest sub-game (the sub-game $\mathcal{G}_i$ having minimal $\mathcal{E}(\mathcal{G}_i)$ value).
2. For this sub-game compute the utility upper bound as: $\mathcal{U}(\mathcal{G}) = \mathcal{P} - \mathcal{E}(\mathcal{G}_i)$.

It can be shown that in the new model

$$\mathcal{E}(\mathcal{G}_1 \vee \ldots \vee \mathcal{G}_k) = \min\{\mathcal{E}(\mathcal{G}_1), \ldots, \mathcal{E}(\mathcal{G}_k)\} \ ,$$
$$\max\{\mathcal{E}(\mathcal{G}_1), \ldots, \mathcal{E}(\mathcal{G}_k)\} \leqslant \mathcal{E}(\mathcal{G}_1 \wedge \ldots \wedge \mathcal{G}_k) \leqslant \mathcal{E}(\mathcal{G}_1) + \ldots + \mathcal{E}(\mathcal{G}_k) \ .$$

The last inequality turns into an equation if the games $\mathcal{G}_1, \ldots, \mathcal{G}_k$ have no common moves.

5.2 Expenses Reduction

In the following section we will discuss the problem associated with the games that have common moves and suggest a solution to it.

Let $\mathcal{G}_1$ and $\mathcal{G}_2$ be sub-games of game $\mathcal{G}$. Those sub-games may in turn contain the conjunctive as well as disjunctive sub-games alternately with no evidence if those sub-games contain no common moves. We assume that some of the sub-games may contain common moves and that the optimal strategy might utilize them. Thus some of the atomic attacks may be referenced more than once and multiply their corresponding investments into the expenses parameter that these nodes propagate.

In order to get the correct utility for the intermediate sub-games $\mathcal{G}_1, \mathcal{G}_2$ and, eventually, $\mathcal{G}$ we artificially reduce the expenses $\mathcal{E}(\mathcal{X}_i)$ for the common moves and produce the modified move parameter $\tilde{\mathcal{E}}(\mathcal{X}_i)$ that will here and further be referenced as *reduced expenses*. It is reasonable to reduce expenses by the amount of occurrences of the same move in a sub-game. In graph representation we reduce the expenses by the amount of references (incoming edges) to the atomic move. Let us denote the number of occurrences of the atomic move $\mathcal{X}_i$ as $e_{\mathcal{X}_i}$. Thus $\tilde{\mathcal{E}}(\mathcal{X}_i) = \frac{\mathcal{E}(\mathcal{X}_i)}{e_{\mathcal{X}_i}}$.

Although by reducing the expenses of the common moves we make them easier to play, the idea behind this is that if the system can be proven to be secure even if some of the atomic attacks are artificially made easier than they really are, this implies that the attacks against the real system are infeasible.

We present an *Algorithm 5.1* for expenses lower bound computation using expenses propagation which runs in time linear in the size of the game, thus complexity is $\mathcal{O}(n)$. The utility upper bound can be computed as $\mathcal{U}(\mathcal{G}) = \mathcal{P} - \mathcal{E}(\mathcal{G})$, where $\mathcal{E}(\mathcal{G})$ is the expenses lower bound of the game. The local optimum decisions that are made in the disjunctive games are not subject to the problem discovered by Jürgenson and Willemson in [6], as the optimal strategy, according to *Thm. 8* in [3], is a critical min-term of the Boolean function representing the game instance, and a critical min-term is not redundant. Thus local optimum decisions are global optimum decisions in the new model and allow us to use the expenses propagation approach and the recursive algorithm 5.1.

ALGORITHM 5.1: Iterated *Expenses* propagation

```
Input: The game $\mathcal{G}$
Output: The expenses of the game $\mathcal{E}(\mathcal{G})$ (a real number)
Procedure ComputeExpenses ($\mathcal{G}$)
if m is a conjunctive game instance then
    expenses := 0
    forall the sub-games i of m do
        expenses += ComputeExpenses (i)
else if m is a disjunctive game instance then
    cheapest := FindCheapestSubGame ($m$)
    return cheapest
else m is an atomic move
    return $\mathcal{E}(m)$
```

6 Interpretation of Results

The new model allows us to compute the adversarial utility upper bound. In case it is positive the analyzed system lacks security at some point and profitable attack vectors, that can result in a positive outcome for an attacker, are likely to exist. If the utility upper bound is 0, we may conclude that the system is potentially secure against rational gain-oriented attackers. The presented model still relies on the ability of analysts to construct an attack tree precisely enough to capture all feasible attack vectors and reflect the real system being modeled. Security has to be a continuous cyclic process where the list of threats and vulnerabilities is being continuously revised.

7 Open Questions and Future Research

The research on the presented model is still unfinished. Future research will focus on the unsolved problems and open questions.

As mentioned earlier, attack tree models including the new one, depend on the metrics assigned to the atomic attacks. Unfortunately, efficient frameworks for metrics estimation do not exist yet. Should one be developed, it would be a valuable addition not only to this model, but to all the models that utilize attack trees.

Secondly, current attack tree models using the game-theoretic approach have one single node—the root node that is assigned the prize parameter—the revenue for an attacker. However, in reality some intermediate nodes may have their own value for an attacker and in the model may be assigned with their own prize. Those nodes can represent the secondary goals of an attacker and affect the strategy in certain cases.

Finally, it would be useful to extend the model capabilities to take possible defensive measures into account and to extend the notion of *attack trees* to the notion of *attack-defense trees.* Those defensive measures, if applied, can affect the parameters the respective nodes propagate.

Although quantified security analysis is an area that has been thoroughly studied, we cannot say that the results meet the requirements of real life. Further research in this area is required to ensure the reliability and trustworthiness of the developed models.

References

1. Buldas, A., Laud, P., Priisalu, J., Saarepera, M., Willemson, J.: Rational choice of security measures via multi-parameter attack trees. In Lopez, J. (Ed.): CRITIS 2006, LNCS 4347, pp. 235-248, 2006.
2. Buldas, A., Mägi, T.: Practical security analysis of e-voting systems. In Miyaji, A., Kikiuchi, H., Rannenberg, K. (Eds.): IWSEC 2007, LNCS 4752, pp 320-335, 2007.
3. Buldas, A., Stepanenko, R.: Upper bounds for adversaries' utility in attack trees. In: J. Grossklags and J. Walrand (Eds.): GameSec 2012, LNCS 7638, pp.98-117, 2012.
4. Jürgenson, A., Willemson, J.: Computing exact outcomes of multi-parameter attack trees. In Meersman, R., Tari, Z. (Eds.): OTM 2008, LNCS 5332, pp 1036-1051, 2008.
5. Jürgenson, A., Willemson, J.: Serial model for attack tree computations. In Lee, D., Hong, S. (Eds.): ICISC 2009, LNCS 5984, pp 118-142, 2010.
6. Jürgenson, A., Willemson, J.: Efficient Semantics of Parallel and Serial Models of Attack Trees, TUT, 2010
7. Luby, M.: Pseudorandomness and Cryptographic Applications. In Hanson, D.R., Tarjan, R.E. (Eds.): Princeton Computer Science Notes. Princeton University Press, 1996.
8. Mauw, S., Oostdijk, M.: Foundations of attack trees. In Won, D., Kim, S. (Eds.): ICISC 2005, LNCS 3935, pp 186-198, 2005.
9. Niitsoo, M.: Optimal adversary behavior for the serial model of financial attack trees. In Echizen, I., Kunihiro, N., Sasaki, R. (Eds.): IWSEC 2010, LNCS 6364, pp 354-370, 2010.

A Computational Example

Example 1. In order to demonstrate the application of the proposed model, an example is presented. Consider the attack tree shown in Fig. 2. Attack tree leaves parameters are shown in Table 1 and the computed expenses and utilities in Table 2.
The *Expenses* parameter present in the model is represented as a function of attack preparation costs $\mathcal{C}$, attack detection probability r and the penalty Π such that $\mathcal{E}(\mathcal{X}_i) = \mathcal{C}_{\mathcal{X}_i} + r_{\mathcal{X}_i}\Pi_{\mathcal{X}_i}$. Firstly, each of the atomic attacks $\mathcal{X}_i$ parameters *success probability* $p_{\mathcal{X}_i}$ and *expenses* $\mathcal{E}(\mathcal{X}_i)$ are transformed to the form applicable for the failure-free Model: $p_{\mathcal{X}_i} \to 1$; $\mathcal{E}(\mathcal{X}_i) \to \mathcal{E}_{inf}(\mathcal{X}_i) = \frac{\mathcal{E}(\mathcal{X}_i)}{p}$.
Afterwords the *expenses reduction technique* is applied producing the reduced expenses $\tilde{\mathcal{E}}(\mathcal{X}_i)$ of each of the atomic attacks. In this particular case the expenses parameters of the nodes remain the same, except for the $\mathcal{A}_{2.1.2}$ node which is referenced twice and thus its reference count $e = 2$, thus $\mathcal{E}_{inf}(\mathcal{A}_{2.1.2}) = \frac{\mathcal{E}(\mathcal{A}_{2.1.2})}{2} =$

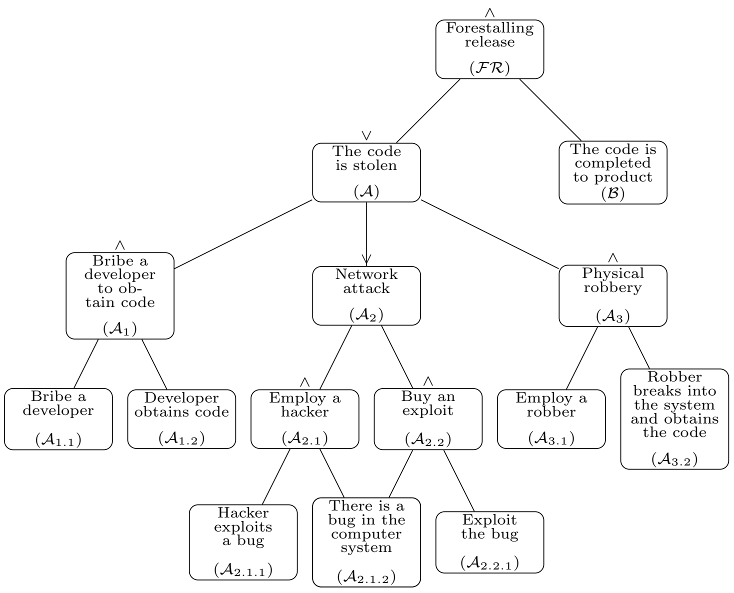

∧
Forestalling release ($\mathcal{FR}$)
∨
The code is stolen ($\mathcal{A}$)
The code is completed to product ($\mathcal{B}$)
∧
Bribe a developer to ob- tain code ($\mathcal{A}_1$)
Network attack ($\mathcal{A}_2$)
∧
Physical robbery ($\mathcal{A}_3$)
Bribe a developer ($\mathcal{A}_{1.1}$)
Developer obtains code ($\mathcal{A}_{1.2}$)
∧
Employ a hacker ($\mathcal{A}_{2.1}$)
∧
Buy an exploit ($\mathcal{A}_{2.2}$)
Employ a robber ($\mathcal{A}_{3.1}$)
Robber breaks into the system and obtains the code ($\mathcal{A}_{3.2}$)
Hacker exploits a bug ($\mathcal{A}_{2.1.1}$)
There is a bug in the computer system ($\mathcal{A}_{2.1.2}$)
Exploit the bug ($\mathcal{A}_{2.2.1}$)

Fig. 2. A sample attack tree for a software developing company. The conjunctive game instances are depicted with red background and with ∧ label above the node, the disjunctive game instances are depicted with blue background and with ∨ label above the node and atomic moves are depicted with green background.

Table 1. Estimated and calculated values of atomic attacks

Threat	Description	p	r	$\mathcal{C}$	Π	Expenses	FFM Expenses	RED Expenses	Utility
$\mathcal{B}$	Stolen code is used in products	0.9	0.9	10^6	10^6	190000	2111111.1	2111111.1	-1010111.1
$\mathcal{A}_{1.1}$	Bribe a developer	0.1	0.2	10^6	10^3	1000200	10002000	10002000	-8901000
$\mathcal{A}_{1.2}$	Developer obtains code	0.9	0.005	0	10^5	500	555.5	555.5	1100444.4
$\mathcal{A}_{2.1.1}$	Hacker exploits a bug	0.5	0.5	10^3	1	1000.5	2001	2001	1098999
$\mathcal{A}_{2.1.2}$	An exploitable bug exists	0.006	0.005	0	0	0	0	0	1101000
$\mathcal{A}_{2.2.1}$	Exploit the bug	0.5	0.1	0	1	0.1	0.2	0.2	1100999.8
$\mathcal{A}_{3.1}$	Employ a robber	0.9	0.001	10^5	10^4	100010	111122.222	111122.222	989877.778
$\mathcal{A}_{3.2}$	Robber obtains the code	0.5	0.9	10^3	10^5	91000	182000	182000	919000

$\frac{0}{2} = 0$. Secondly, the adversarial utility is calculated using two methodologies, the expenses propagation approach as well as the utility propagation approach. Calculated parameters of intermediate nodes properties are introduced below. The root node $\mathcal{FR}$ prize $\mathcal{P}$ = 1,101,000.

Table 2. Attack Tree Expenses and Utility

Node	Description	Type	Expenses	Utility[1]	Utility[2]
$\mathcal{R}$	Forestalling release	AND	2111111.311	-1010111.311	-1010111.111
$\mathcal{A}$	Steal the code	OR	0.2	-	1100999.8
$\mathcal{A}_1$	Insider attack	AND	10002555.556	-	-8901000
$\mathcal{A}_2$	Network attack	OR	0.2	-	1100999.8
$\mathcal{A}_{2.1}$	Employ a hacker	AND	2001	-	1098999
$\mathcal{A}_{2.2}$	Buy an exploit	AND	0.2	-	1100999.8
$\mathcal{A}_3$	Physical robbery	AND	293122.222	-	919000

[1] Expenses propagation
[2] Utility propagation

Appendix D

Limiting Adversarial Budget in Quantitative Security Assessment

Limiting Adversarial Budget in Quantitative Security Assessment

Aleksandr Lenin[1,2] and Ahto Buldas[1,2,3] *

[1] Cybernetica AS, Mealuse 2/1, Tallinn, Estonia
[2] Guardtime AS, Tammsaare tee 60, Tallinn, Estonia
[3] Tallinn University of Technology, Ehitajate tee 5, Tallinn, Estonia

Abstract. We present the results of research of limiting adversarial budget in attack games, and, in particular, in the failure-free attack tree models presented by Buldas-Stepanenko in 2012 and improved in 2013 by Buldas and Lenin. In the previously presented models attacker's budget was assumed to be unlimited. It is natural to assume that the adversarial budget is limited and such an assumption would allow us to model the adversarial decision making more close to the one that might happen in real life. We analyze three atomic cases – the single atomic case, the atomic AND, and the atomic OR. Even these elementary cases become quite complex, at the same time, limiting adversarial budget does not seem to provide any better or more precise results compared to the failure-free models. For the limited model analysis results to be reliable, it is required that the adversarial reward is estimated with high precision, probably not achievable by providing expert estimations for the quantitative annotations on the attack steps, such as the cost or the success probability. It is doubtful that it is reasonable to face this complexity, as the failure-free model provides reliable upper bounds, being at the same time computationally less complex.

1 Introduction

The failure-free models [2, 3] provide reliable utility upper bounds, however this results in systems that might be over-secured. It has not been studied how much extra cost the upper-bound oriented methods cause. We present the intermediate results of researching the model assuming that the adversarial budget is limited and compare the results of analysis using adaptive strategies with limited budget to the analysis results of the failure-free model, in which the adversary is not limited in any way. The adversarial limitation is the only limitation applied to

* The research leading to these results has received funding from the European Regional Development Fund through Centre of Excellence in Computer Science (EXCS), the Estonian Research Council under Institutional Research Grant IUT27-1, and the the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement ICT-318003 (TREsPASS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.

the adversary, all other assumptions and concepts are identical to the failure-free model.

The assumption that the adversarial budget is limited is natural, as this is what happens in reality. Limited budget models the adversarial strategic decision making in a better way, which is more close to the one likely to be observed in real life and the research on the adaptive strategies with limited budget is an important research area in quantitative security analysis based on attack trees.

We analyze three cases: the atomic attack case, the atomic AND, and the atomic OR analyzing the effect of limiting adversarial budget in fully-adaptive strategies [2, 3]. We show that the atomic attack case and the atomic AND case do not provide whatsoever better or more reliable results, compared to the existing failure-free models. The atomic AND case might provide more precise result, but in this case analysts must estimate the adversarial reward with the required precision, which in real-life scenarios might be less than €1. If they fail to do that, the results of such an analysis are unreliable. In practice, it is doubtful that analysts would be able to come up with such precise estimations. Even if such precise estimations existed, the model would not provide reliable results, as there is still margin for human mistake and in case analysts might overlook the estimations provided to such parameters as cost of the attack step, or the adversarial reward, the results of the analysis would not be reliable. On the contrary, the existing failure-free models with unlimited adversarial budget provide reliable utility upper bounds, despite the fact that this may result in over-secured systems.

It seems that limited budget makes the model much more complex compared to the unlimited budget approach. For example, optimal strategies that were shown to be non-adaptive in the failure-free models [2, 3] can be adaptive and more complex to analyze in the limited budget model. The best move to undertake in certain states of the game changes bouncing between the attack steps.

Even the elementary cases studied in this paper become quite complex considering limited budget assumption compared to the corresponding cases in the failure-free models [2, 3]. It is doubtful that the more general case will have a graceful easy solution to derive optimal strategies. Considering the requirement to be able to estimate the adversarial reward very precisely it is doubtful that it is reasonable to face the complexity of the calculations on the limited adaptive strategies.

The outline of the paper is the following: Section 1 provides a high-level overview of the problem and briefly outlines the results obtained so far. Section 2 describes the work related to the presented approach, Section 3 provides definitions of terms used throughout the paper. Section 4 describes the effect of limited budget assumption on the fully-adaptive strategies and the strategic decision making undertaken by the adversary. Finally, Section 5 summarizes the obtained results, outlines questions still left open, and describes interesting problems for future research.

2 Related Work

In this section, we outline the work that has lead to and influenced the development of the presented model.

2.1 Schneier Attack Trees' Concept

The idea of analyzing security using the so-called attack trees was popularized by Schneier in [7]. The author suggested to use attack trees as a convenient hierarchical representation of an attack scenario. The analysis implied that the analysts had to estimate one single parameter they would like to reason about, for each of the leaves in the attack tree. Then the bottom-up parameter propagation approach was applied to propagate the results of calculations towards the root node of the tree, the result of the root node was considered the result of such an analysis. The suggested bottom-up parameter propagation method allowed to reason about such parameters like minimal/average/maximal cost of the attack scenario, likelihood of its success, etc. The analysis relied on an assumption that the analyzed parameters are mutually independent, which allowed to analyze them independently of each other and to derive some meaningful conclusions about the security of the systems based on the obtained results.

2.2 Buldas-Priisalu Model

The model of Buldas *et al.* [1] is remarkable for introducing the multi-parameter approach to the quantitative security risk analysis. The model is based on the assumption of a rational adversary who is always trying to maximize his average outcome. The authors state that in order to assess security it is sufficient to assess adversarial utility. If the utility is negative or zero, the system is reasonably secure, as attacking it is not profitable. If the utility is positive, the adversary has an incentive to attack and attacking is profitable for him. The adversary undertakes strategic decision-making in accordance with the rationality assumption – the adversary will start attacking iff it is profitable. Additionally, authors state that malicious actions are, as a rule, related to criminal behavior and for this reason they applied economic reasoning in their model which considers the risk of detection and potential penalties of the adversary. Their model introduced a novel way to think about security and gave start to multi-parameter quantitative security analysis. Jürgenson *et al.* have shown that Buldas *et al.* model is inconsistent with Mauw-Oostijk foundations [6] and introduced the so-called parallel model [4] and the serial model [5] which provided more reliable results, however in both models the adversary did not behave in a fully adaptive way.

2.3 Buldas-Stepanenko Fully Adaptive Model

In the Buldas-Stepanenko fully adaptive model [3] the adversaries behave in a fully adaptive way launching atomic attack steps in an arbitrary order, depending on the results of the previous trials. However, the model had force-failure states, when the adversary could not continue playing and thus adversarial fully adaptive behavior was limited. In their model optimal strategies are non-adaptive and in some cases, like atomic OR or atomic AND, may be easily derived by calculating certain invariants. In their failure-free model the adversary was expected to launch attack steps until success, thus the failure-free model is similar to the fully adaptive model with the difference that in the failure-free model success probabilities of the attack steps are equal to 1. The most significant contribution of the paper [3] is the upper bounds ideology by which the models should estimate adversarial utility from above, trying to avoid false-positive security results.

2.4 Improved Failure-Free Model

The improved failure-free model [2] improves the Buldas-Stepanenko failure-free model [3] by eliminating the force-failure states. In the improved model the adversarial behavior more fully conforms to the upper bounds ideology introduced in [3] – the adversary may repeat failed attack steps and play on when caught. It turned out that the elimination of the force failure states has made the model computationally easier. The authors show that in the new model optimal strategies always exist. Optimal strategies are single-branched BDD-s where the order of attack steps is irrelevant. Additionally, authors show that finding an optimal strategy in the new model is NP-complete. Two computational methods were introduced – the one allowing to compute the precise adversarial utility value, and the one which allowed to derive the approximated estimation of adversarial utility upper bound.

3 Definitions

Definition 1 (Derived function). *If $\mathcal{F}(x_1, \ldots, x_m)$ is a Boolean function and $v \in \{0, 1\}$, then by the derived Boolean function $\mathcal{F}|_{x_j=v}$ we mean the function $\mathcal{F}(x_1, \ldots, x_{j-1}, v, x_{j+1}, \ldots, x_m)$ derived from $\mathcal{F}$ by the assignment $x_j := v$.*

Definition 2 (Constant functions). *By **1** we mean a Boolean function that is identically true and by **0** we mean a Boolean function that is identically false.*

Definition 3 (Satisfiability game). *By a satisfiability game we mean a single-player game in which the player's goal is to satisfy a monotone Boolean function $\mathcal{F}(x_1, x_2, \ldots, x_k)$ by picking variables x_i one at a time and assigning $x_i = 1$. Each time the player picks the variable x_i he pays some amount of expenses $\mathcal{E}_i \in \mathfrak{R}$, sometimes also modelled as a random variable. With a certain probability p_i the move x_i succeeds. Function $\mathcal{F}$ representing the current game instance is transformed to its derived form $\mathcal{F}|_{x_i=1}$ and the next game iteration starts. The game ends when the condition $\mathcal{F} \equiv 1$ is satisfied and the player wins the prize $\mathcal{P} \in \mathfrak{R}$, or when the player stops playing. With probability $1 - p_i$ the move x_i fails. The player may end up in a different game instance represented by the* *derived Boolean function $\mathcal{F}|_{x_i \equiv 0}$ in the case of a game without move repetitions, and may end up in the very same instance of the game $\mathcal{F}$ in the case of a game with repetitions. Under certain conditions with a certain probability the game may end up in a forced failure state, i.e. if the player is caught and this implies that he cannot continue playing, i.e. according to the Buldas-Stepanenko model [3]. The rules of the game are model-specific and may vary from model to model. Thus we can define three common types of games:*

1. *SAT Game Without Repetitions - the type of a game where an adversary can perform a move only once.*
2. *SAT Game With Repetitions - the type of a game where an adversary can re-run failed moves again an arbitrary number of times.*
3. *Failure-Free SAT Game - the type of a game in which all success probabilities are equal to 1. It is shown in [2] that any game with repetitions is equivalent to a failure-free game (Thm. 5).*

Definition 4 (Satisfiability game with limited budget). *By a satisfiability game with limited budget we mean the SAT game with move repetitions in which the current state of the game is described by the Boolean function $\mathcal{F}(x_1, \ldots, x_k)$ and the budget λ – $\langle \mathcal{F}, \lambda \rangle$. Every move x_i made by the player changes the state of the game. If x_i succeeded, the game moves into the state $\langle \mathcal{F}|_{x_i=1}, \lambda - \mathcal{C}_i \rangle$ and if x_i has failed, the new state of the game is $\langle \mathcal{F}|_{x_i=0}, \lambda - \mathcal{C}_i \rangle$, where $\mathcal{C}_i$ is the cost of x_i. The game ends if the player has satisfied the Boolean function $\mathcal{F} \equiv 1$ and reached the state $\langle \mathbf{1}, \lambda \rangle$ thus winning the game, or when the player has reached the state $\langle \mathcal{F}, \lambda \rangle$ in the case of which the expenses of every possible move $\mathcal{E}_i > \lambda$ and $\mathcal{F}$ has not been satisfied, meaning the loss of the game.*

Definition 5 (Line of a game). *By a line of a satisfiability game we mean a sequence of assignments $\gamma = \langle x_{j_1} = v_1, \ldots, x_{j_k} = v_k \rangle$ (where $v_j \in \{0, 1\}$) that represent the player's moves, and possibly some auxiliary information. We say that γ is a **winning line** if the Boolean formula $x_{i_1} \wedge \ldots \wedge x_{i_k} \Rightarrow \mathcal{F}(x_1, \ldots, x_n)$ is a tautology, where $\mathcal{F}$ is a Boolean function of the satisfiability game.*

Definition 6 (Strategy). *By a strategy $\mathcal{S}$ for a game $\mathcal{G}$ we mean a rule that for any line γ of $\mathcal{G}$ either suggests the next move $x_{j_{k+1}}$ or decides to give up.*

Strategies can be represented graphically as binary decision diagrams (BDDs).

Definition 7 (Line of a strategy). *A line of a strategy $\mathcal{S}$ for a game $\mathcal{G}$ is the smallest set $\mathcal{L}$ of lines of $\mathcal{G}$ such that (1) $\langle\rangle \in \mathcal{L}$ and (2) if $\gamma \in \mathcal{L}$, and $\mathcal{S}$ suggests x_j as the next move to try, then $\langle \gamma, x_j = 0 \rangle \in \mathcal{L}$ and $\langle \gamma, x_j = 1 \rangle \in \mathcal{L}$.*

Definition 8 (Branch). *A branch β of a strategy $\mathcal{S}$ for a game $\mathcal{G}$ is a line γ of $\mathcal{S}$ for which $\mathcal{S}$ does not suggest the next move. By $\mathcal{B}_{\mathcal{S}}$ we denote the set of all branches of $\mathcal{S}$.*

For example, all winning lines of $\mathcal{S}$ are branches.

Definition 9 (Expenses of a branch). *If $\beta = \langle x_{i_1=v_1}, \ldots, x_{i_k=v_k} \rangle$ is a branch of a strategy $\mathcal{S}$ for $\mathcal{G}$, then by expenses $\epsilon_{\mathcal{G}}(\mathcal{S}, \beta)$ of β we mean the sum $\overline{\mathcal{E}}_{i_1} + \ldots + \overline{\mathcal{E}}_{i_k}$ where by $\overline{\mathcal{E}}_{i_j}$ we mean the mathematical expectation of $\mathcal{E}_{i_j}$.*

Definition 10 (Prize of a branch). *The prize $\mathcal{P}_{\mathcal{G}}(\mathcal{S}, \beta)$ of a branch β of a strategy $\mathcal{S}$ is $\mathcal{P}$ if β is a winning branch, and 0 otherwise.*

Definition 11 (Utility of a strategy). *By the utility of a strategy $\mathcal{S}$ in a game $\mathcal{G}$ we mean the sum: $\mathcal{U}(\mathcal{G}, \mathcal{S}) = \sum\limits_{\beta \in \mathcal{B}_{\mathcal{S}}} Pr(\beta) \cdot [\mathcal{P}_{\mathcal{G}}(\mathcal{S}, \beta) - \epsilon_{\mathcal{G}}(\mathcal{S}, \beta)]$. For the empty strategy $\mathcal{U}(\mathcal{G}, \emptyset) = 0$.*

Definition 12 (Prize and Expenses of a strategy). *By the expenses $\mathcal{E}(\mathcal{G}, \mathcal{S})$ of a strategy $\mathcal{S}$ we mean the sum $\sum\limits_{\beta \in \mathcal{B}_{\mathcal{S}}} Pr(\beta) \cdot \epsilon_{\mathcal{G}}(\mathcal{S}, \beta)$. The prize $\mathcal{P}(\mathcal{G}, \mathcal{S})$ of $\mathcal{S}$ is $\sum\limits_{\beta \in \mathcal{B}_{\mathcal{S}}} Pr(\beta) \cdot \mathcal{P}_{\mathcal{G}}(\mathcal{S}, \beta)$.*

It is easy to see that $\mathcal{U}(\mathcal{G}, \mathcal{S}) = \mathcal{P}(\mathcal{G}, \mathcal{S}) - \mathcal{E}(\mathcal{G}, \mathcal{S})$.

Definition 13 (Utility of a satisfiability game). *The utility of a SAT game $\mathcal{G}$ is the limit $\mathcal{U}(\mathcal{G}) = \sup\limits_{\mathcal{S}} \mathcal{U}(\mathcal{G}, \mathcal{S})$ that exists due to the bound $\mathcal{U}(\mathcal{G}, \mathcal{S}) \leqslant \mathcal{P}$.*

Definition 14 (Optimal strategy). *By an optimal strategy for a game $\mathcal{G}$ we mean a strategy $\mathcal{S}$ for which $\mathcal{U}(\mathcal{G}) = \mathcal{U}(\mathcal{G}, \mathcal{S})$.*

It is shown in [2] that for satisfiability games optimal strategies always exist [2].

4 Limiting Adversarial Budget in the Improved Failure-Free Model.

In this paper we focus on the fully adaptive adversarial strategies assuming that the adversarial budget is limited. Budget limitation is the only limitation used, compared to the improved failure-free model [2]. Adversaries still behave in a fully adaptive way and are allowed to launch failed attack steps again in any order, until the budget gets so small that no attack steps can be launched. When the budget decreases by a considerable amount, monetary limitation starts effecting possible strategic choices of the attacker – possible set of choices reduces (the adversary may launch only some subset of the attack steps) and eventually, this subset becomes an empty set. It turns out that the optimal strategy depends on the amount of the monetary resource available to the adversary.

In the improved failure-free model the state of the game is represented by the Boolean function $\mathcal{F}$. If the attack step has failed, the adversary finds himself in the very same state of the game $\mathcal{F}$. Due to this non-adaptive strategies always exist in the set of optimal strategies of the game.

This is not always the case when we consider budget limitations – in general, optimal strategies are adaptive, except for some certain sets of parameters in case of which optimal strategies are non-adaptive. When we consider budget limitations the state of the game is represented by the Boolean function $\mathcal{F}$ and the budget λ. We denote the utility in a certain game state $\langle \mathcal{F},\, \lambda \rangle$ with $\mathcal{U}^{\lambda}(\mathcal{F})$. When an attack step fails, the adversary finds himself in an another state of the game represented by $\mathcal{U}^{\lambda-\mathcal{C}}(\mathcal{F})$, where $\mathcal{C}$ is the cost of the failed attack step. The relation between the utility upper bound $\mathcal{U}^{\infty}(\mathcal{F})$ in [2] and the utility $\mathcal{U}^{\lambda}(\mathcal{F})$ given budget λ is the following:

$$\mathcal{U}^{\infty}(\mathcal{F}) = \lim_{\lambda \to \infty} \mathcal{U}^{\lambda}(\mathcal{F}) \ .$$

In some certain cases optimal strategies are non-adaptive, but in general they are not. This makes computations reasonably complex. When the adversarial budget increases, his utility increases as well and approaches the adversarial utility upper bound in the improved failure-free model [2]. It turns out that in the case of a reasonably big budget the complexity added by the budget limitation does not add any value nor give any additional benefits, as the difference between the utility in the model with budget limitations and the utility upper bound becomes negligible.

In this paper we focus on the three elementary games – the single attack case, the atomic AND and the atomic OR game and show the effect of budget limitations in these games. Even these elementary cases become quite complex when taking budget limitations into account. It becomes doubtful if the practical application of the model with budget limitations is efficient and reliable. Using complex computational procedures we face the risk to make the model inapplicable for the practical cases, while the negligible deviation between the results of the model with budget limitations and the one without them in case of a reasonably big budget (which is the expected case in real-life scenarios) and much less complex and more efficient computations induces us to give preference to the model without budget limitations, despite the fact that it overestimates adversarial power and capabilities for the cases when the adversarial budget is reasonably small.

4.1 Single Atomic Attack Case

In case the adversary may choose from a single available choice, he will continue launching the attack step until it succeeds, or as long as the budget allows it. Such a strategy may be represented in the form of a single-branched BDD as in Fig. 1:
In accordance with the strategy, the adversary launches an attack step $\mathcal{X}$ with cost $\mathcal{C}$ and success probability p. If it succeeds, the adversary has accomplished the attack and has won the game. If $\mathcal{X}$ fails, the adversary finds himself in another state of the game $\langle \mathcal{X},\, \lambda - \mathcal{C} \rangle$. Thus, adversarial utility may be expressed in the form of the relation (1):

$$\mathcal{U}^{\lambda}(\mathcal{X}) = \max\left\{0,\ \mathcal{U}(\mathcal{X}) + (1-p)\cdot \mathcal{U}^{\lambda-\mathcal{C}}(\mathcal{X})\right\} \ . \tag{1}$$

$$\begin{tikzcd}
\mathcal{X} \arrow[r, "\overline{\mathcal{X}}"] & \mathcal{X} \arrow[r, "\overline{\mathcal{X}}"] & \cdots \arrow[r, "\overline{\mathcal{X}}"] & \mathcal{X} \\
\mathcal{U}^{\lambda}(\mathcal{X}) & \mathcal{U}^{\lambda-\mathcal{C}_x}(\mathcal{X}) & & \mathcal{U}^{\lambda-k\cdot\mathcal{C}_x}(\mathcal{X})
\end{tikzcd}$$

Fig. 1. An adaptive strategy suggesting to iterate attack step $\mathcal{X}$ until it succeeds or as long as the adversarial budget allows to launch the attack step.

It can be seen (see Fig. 2) that the adversarial utility changes in the points where the budget is multiples of the cost of the attack step. In case the adversarial budget is less than the cost of the attack step, the adversary cannot launch a single attack step and thus his utility is 0. The optimal strategy in this case is an empty strategy – the attacker will be better off not trying to attack. In case the budget exceeds the cost of the attack step, the utility grows with each subsequent trial to launch an attack step, as every subsequent trial increases the likelihood of success that the attack step will succeed. Thus, adversarial utility asymptotically approaches the utility upper bound in the model without budget limitations.

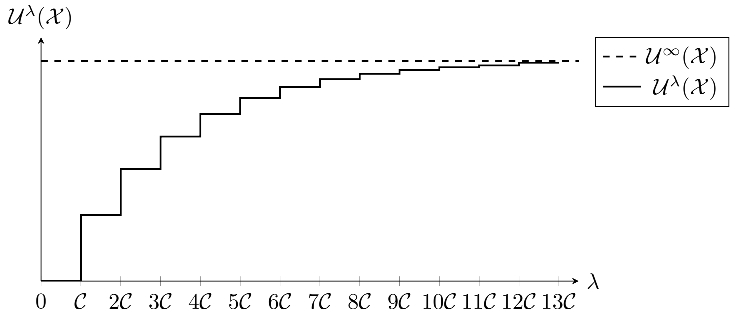


Fig. 2. Single atomic attack case.

The utility value that the adversary may achieve, given budget λ, may be expressed in the form of equation (2):

$$\mathcal{U}^{\lambda}(\mathcal{X}) = \left[\mathcal{P} - \frac{\mathcal{C}}{p}\right] \cdot \left[1 - (1-p)^{\lfloor \frac{\lambda}{\mathcal{C}} \rfloor}\right] = \mathcal{U}^{\infty}(\mathcal{X}) \left[1 - (1-p)^{\lfloor \frac{\lambda}{\mathcal{C}} \rfloor}\right] \quad , \qquad (2)$$

where $\mathcal{U}^{\infty}(\mathcal{X})$ is the utility upper bound [2].

Comparison with the Improved Failure-Free Model

We will investigate the case when the improved failure-free model analysis result states that the system is *insecure*, while the budgeted model result states that the system is *secure*.

According to the improved failure-free model the adversarial utility $\mathcal{U}^{\infty}(\mathcal{X}) = \mathcal{P} - \frac{\mathcal{C}}{p}$. The system is secure in case $\mathcal{P} \leqslant \frac{\mathcal{C}}{\mathcal{P}}$ and insecure in case $\mathcal{P} > \frac{\mathcal{C}}{\mathcal{P}}$.

$$\begin{cases} \mathcal{U}^{\infty}(\mathcal{X}) & = \mathcal{P} - \frac{\mathcal{C}}{p} > 0 \\ \mathcal{U}^{\lambda}(\mathcal{X}) & = \left[\mathcal{P} - \frac{\mathcal{C}}{p}\right]\left[1 - (1-p)^{\left\lfloor \frac{\lambda}{\mathcal{C}} \right\rfloor}\right] \leqslant 0 \end{cases} \tag{3}$$

It can be seen that the condition (3) can be reached only when the adversary has no resources to attack ($\lambda < \mathcal{C}$). Thus limiting adversarial budget does not provide more trustworthy nor more reliable results compared to the improved failure-free model in case of single atomic attack games. If in the case of some positive budget λ the adversarial utility is positive, it will be less or equal to zero in the model with budget limitations only if $\lambda < \mathcal{C}$. In other words, if the system is insecure in the improved failure-free model, it will also be insecure in the model with budget limitations for any adversarial budget, sufficient to launch the attack step at least once.

4.2 Two Attack Steps

In the case of atomic games of 2 possible attack steps $\mathcal{X}_i$ and $\mathcal{X}_j$ and corresponding costs $\mathcal{C}_{\mathcal{X}_i}$ and $\mathcal{C}_{\mathcal{X}_j}$, the adversarial utility changes in the so-called *lattice points* which are the projections of points $(n\mathcal{C}_{\mathcal{X}_i}, m\mathcal{C}_{\mathcal{X}_j})$ in two-dimensional Euclidean space into one-dimensional space using the formula $\mathcal{L}_i = n\mathcal{C}_{\mathcal{X}_i} + m\mathcal{C}_{\mathcal{X}_j}$, where $n \in \left\{1, 2, \ldots, \left\lfloor \frac{\lambda}{\mathcal{C}_{\mathcal{X}_i}} \right\rfloor\right\}$, $m \in \left\{1, 2, \ldots, \left\lfloor \frac{\lambda}{\mathcal{C}_{\mathcal{X}_j}} \right\rfloor\right\}$, $\forall i : \mathcal{L}_i \leqslant \lambda$ (see Fig. 3). In the case of three attack steps the utility changes in the projections of points in three-dimensional space into one-dimensional space. Thus with the increase in the amount of possible attack steps the lattice argument space becomes more complex.

It can be shown that the distance between the two adjacent lattice points has a lower bound.

Theorem 1. *If the relation of attack step costs may be expressed in terms of a rational fraction (a fraction of two rational numbers, corresponding cost values may be irrational)* $\frac{\mathcal{C}_{\mathcal{X}_i}}{\mathcal{C}_{\mathcal{X}_j}} = \frac{p}{q}$, *then the distance between two adjacent lattice points* $\mathcal{L}_i$ *and* $\mathcal{L}_{i+1}$ *will be not less than* $\frac{\mathcal{C}_{\mathcal{X}_j}}{q}$.

Proof. The distance δ between the two adjacent lattice points $\mathcal{L}_i$ and $\mathcal{L}_{i+1}$ may be expressed as

$$\delta = |(n-n')\mathcal{C}_{\mathcal{X}_i} + (m-m')\mathcal{C}_{\mathcal{X}_j}| = |\underbrace{(n-n')p + (m-m')q}_{\alpha\in\mathbb{Z}}| \cdot \frac{\mathcal{C}_{\mathcal{X}_j}}{q}$$

$$= \begin{cases} 0 & \text{, if } \alpha = 0, \\ \geqslant \frac{\mathcal{C}_{\mathcal{X}_j}}{q} & \text{, if } \alpha \neq 0. \end{cases}$$

□

If the ratio of the attack step costs is irrational, lattice points appear with increasing frequency eventually positioning infinitely close to each other. In real life we can expect the costs to be rational (it would be difficult to estimate an irrational value for the cost parameter) and for this reason the above mentioned bound exists in the practical cases.

Atomic OR Case

In the case of an atomic OR game in order to win it is sufficient that any of the two attack steps, $\mathcal{X} = \{\mathcal{X}_i,\, \mathcal{X}_j\}$ succeeds. The initial state of the game is $\langle \mathcal{X}_i \vee \mathcal{X}_j,\, \lambda\rangle$ and the subset of available attack steps to launch is $\{\mathcal{X}_i, \mathcal{X}_j\}$. In each state of the game the player may choose to launch any attack step from the subset of available attack steps, or to discontinue playing. The attacker launches an attack step $\mathcal{X}_k$ from this set. If $\mathcal{X}_k$ succeeded the game moves into the state $\langle \mathbf{1},\, \lambda - \mathcal{C}_k\rangle$, where $\mathcal{C}_k$ is the cost of the launched attack step, and the player has won the game. If the attack has failed, the game moves into the state $\langle \mathcal{X}_i \vee \mathcal{X}_j,\, \lambda - \mathcal{C}_k\rangle$ and the game goes on while $\mathcal{E}_k \leqslant \lambda$. At some point the current λ will reduce the set of available attacks to one (cheapest) attack, and eventually, the set of possible attacks becomes an empty set. Upon reaching the state in which $\mathcal{E}_k > \lambda$ and the Boolean function of the game has not been satisfied – the player has lost the game.
Adversarial utility may be expressed in the form of the relation (4):

$$\mathcal{U}^{\lambda}(\mathcal{X}_i \vee \mathcal{X}_j) = \max \begin{cases} 0 \ , \\ \mathcal{U}(\mathcal{X}_i) + (1 - p_{\mathcal{X}_i})\, \mathcal{U}^{\lambda - \mathcal{C}_{\mathcal{X}_i}}(\mathcal{X}_i \vee \mathcal{X}_j) \ , \\ \mathcal{U}(\mathcal{X}_j) + (1 - p_{\mathcal{X}_j})\, \mathcal{U}^{\lambda - \mathcal{C}_{\mathcal{X}_j}}(\mathcal{X}_i \vee \mathcal{X}_j) \ . \end{cases} \tag{4}$$

In certain cases under certain conditions the optimal strategy in the atomic OR case is non-adaptive and suggests to repeat one of the attacks independently of the current state of the game. We will bring an example of such a case.

Theorem 2. *If the costs of the attacks are equal, the attack having greater success probability will be best to try in every state of the game.*

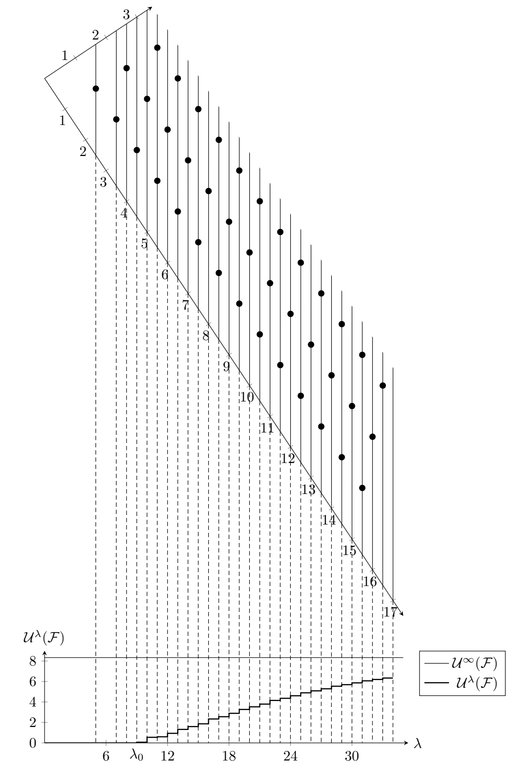


Fig. 3. Projections of the lattice points in the two-dimensional space into the one-dimensional space.

Proof. Assume that $\mathcal{C}_{\mathcal{X}_i} = \mathcal{C}_{\mathcal{X}_j} = \mathcal{C}$. The utility of the game may be expressed in the form of

$$\mathcal{U}^\lambda(\mathcal{X}_i \vee \mathcal{X}_j) = \max \begin{cases} 0 \ , \\ \mathcal{U}(\mathcal{X}_i) + (1 - p_{\mathcal{X}_i}) \cdot \mathcal{U}^{\lambda - \mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) \ , \\ \mathcal{U}(\mathcal{X}_j) + (1 - p_{\mathcal{X}_j}) \cdot \mathcal{U}^{\lambda - \mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) \ . \end{cases}$$

Optimal strategy will suggest to try attack $\mathcal{X}_i$ if

$$\mathcal{U}(\mathcal{X}_i) + (1 - p_{\mathcal{X}_i}) \cdot \mathcal{U}^{\lambda - \mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) > \mathcal{U}(\mathcal{X}_j) + (1 - p_{\mathcal{X}_j}) \cdot \mathcal{U}^{\lambda - \mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) \quad (5)$$

Solving inequality (5) we reach condition $p_{\mathcal{X}_i} > p_{\mathcal{X}_j}$. □

Algorithm 4.1 outlines the recursive procedure to calculate maximal adversarial utility in the atomic OR game given budget λ according to (4).

Algorithm 4.1: Maximal utility of the atomic OR case with the given budget

```
   Input: Attack step $\mathcal{X}_i$ cost i_cost
   Input: Attack step $\mathcal{X}_i$ probability i_pr
   Input: Attack step $\mathcal{X}_j$ cost j_cost
   Input: Attack step $\mathcal{X}_j$ probability j_pr
   Input: Prize of the game prize
   Input: Budget budget
   Output: Maximal adversarial utility value (a real number)
1  Procedure AtomicOr (i_cost, i_pr, j_cost, j_pr, prize, budget)
2  if budget is less than i_cost and j_cost then
3      return (0)
4  i_utility := -i_cost + i_pr · prize
5  j_utility := -j_cost + j_pr · prize
6  if budget is greater than i_cost then
7      $u_i$ = i_utility + (1-i_pr) · AtomicOr (i_cost, i_pr, j_cost, J_pr, prize,
       budget-i_cost)
8      if $u_i$ is negative then
9          $u_i$ := 0
10 if budget is greater than j_cost then
11     $u_j$ = j_utility + (1-j_pr) · AtomicOr (i_cost, i_pr, j_cost, j_pr, prize,
       budget-j_cost)
12     if $u_j$ is negative then
13         $u_j$ := 0
14 if $u_i$ is not less than $u_j$ then
15     maximal_utility := $u_i$
16 else
17     maximal_utility := $u_j$
18 return (maximal_utility)
```

We show how the best move changes in the atomic OR game, depending on the current budget λ demonstrating it by several examples:

The first example (Fig. 4) shows that the best move bounces between the two attack steps when the budget is rather small, and sticks to one attack step later on. By $\emptyset$ we mean that the best move is not to start attacking at all.

The second example (Fig. 5) demonstrates the case when both of the attack steps are equally good when the budget is rather small and thus there is no difference for the attacker whether to launch attack step $\mathcal{X}_i$ or to launch attack step $\mathcal{X}_j$. But when the budget increases, the adversary has a clear preference for one attack over the other one. By $=$ we mean that launching attack step $\mathcal{X}_i$ is as good as launching attack step $\mathcal{X}_j$.

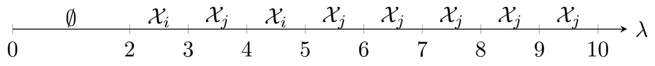


Fig. 4. Atomic OR case with parameters $\mathcal{C}_{\mathcal{X}_i} = 2, p_{\mathcal{X}_i} = 0.3, \mathcal{C}_{\mathcal{X}_j} = 3, p_{\mathcal{X}_j} = 0.48, Prize = 30$.

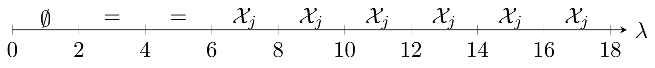


Fig. 5. Atomic OR case with parameters $\mathcal{C}_{\mathcal{X}_i} = 2, p_{\mathcal{X}_i} = 0.05, \mathcal{C}_{\mathcal{X}_j} = 6, p_{\mathcal{X}_j} = 0.9, Prize = 30$.

The third example (Fig. 6) demonstrates the case when the costs of the attacks are irrational, but their relation may be expressed in terms of a fraction of rational numbers. It can be seen that the best move to undertake in a certain state of the game alternates between attack steps $\mathcal{X}_i$ and $\mathcal{X}_j$.

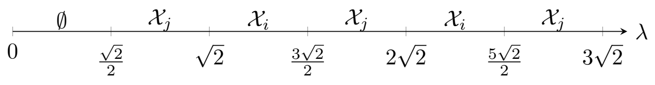


Fig. 6. Atomic OR case with parameters $\mathcal{C}_{\mathcal{X}_i} = \sqrt{2}, p_{\mathcal{X}_i} = 0.8, \mathcal{C}_{\mathcal{X}_j} = \frac{\sqrt{2}}{2}, p_{\mathcal{X}_j} = 0.45, Prize = 30$.

The next example (Fig. 7) demonstrates that there are cases where the optimal strategy is non-adaptive and iterates one single attack step $\mathcal{X}_j$.

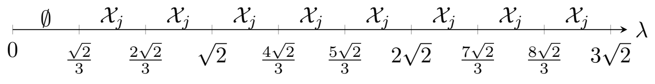


Fig. 7. Atomic OR case with parameters $\mathcal{C}_{\mathcal{X}_i} = \sqrt{2}, p_{\mathcal{X}_i} = 0.1, \mathcal{C}_{\mathcal{X}_j} = \frac{\sqrt{2}}{3}, p_{\mathcal{X}_j} = 0.38, Prize = 30$.

Comparison with the Improved Failure-Free Model

We will show that the case when the improved failure-free model analysis result states that the system is *insecure*, while the budgeted model result states that the system is *secure* is impossible. Lets consider adversarial budget $\mathcal{I}$ for which the following inequalities hold:

$$\mathcal{U}^{\mathcal{I}}(\mathcal{X}_i \vee \mathcal{X}_j) > 0 \ , \tag{6}$$
$$\mathcal{U}^{\mathcal{I}-\mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) \leqslant 0 \ , \tag{7}$$

where $\mathcal{C}$ is the cost of any of the atomic attacks. Assuming $\mathcal{I}$ is greater than the costs of attacks $\mathcal{X}_i$ and $\mathcal{X}_j$:

$$\mathcal{U}^{\mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) \leqslant 0 \ . \tag{8}$$

Let $\mathcal{X}_k$ with cost $\mathcal{C}$ and probability p be the optimal move in the considered state of the game. In this case:

$$\mathcal{U}^{\mathcal{I}}(\mathcal{X}_i \vee \mathcal{X}_j) = \mathcal{U}^{\mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) + (1-p) \cdot \mathcal{U}^{\mathcal{I}-\mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) \ . \tag{9}$$

As $\mathcal{U}^{\mathcal{I}-\mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) \leqslant 0$ by (7) and $\mathcal{U}^{\mathcal{C}}(\mathcal{X}_i \vee \mathcal{X}_j) \leqslant 0$ by (8), it contradicts with the initial assumption $\mathcal{U}^{\mathcal{I}}(\mathcal{X}_i \vee \mathcal{X}_j) > 0$. Thus it seems that there is no point in limiting adversarial budget in the elementary OR case.

Atomic AND Case

In the case of atomic AND game in the initial state of the game the adversary has to choose either to launch the attack step $\mathcal{X}_i$, or to launch $\mathcal{X}_j$ or not to start playing. If the adversary has chosen to launch attack $\mathcal{X}_i$ and it has failed, the game moves into the state $\langle \mathcal{X}_i \wedge \mathcal{X}_j, \lambda - \mathcal{C}_{\mathcal{X}_i} \rangle$. If $\mathcal{X}_i$ succeeded, the game moves into the state $\langle \mathcal{X}_i \wedge \mathcal{X}_j|_{\mathcal{X}_i=1}, \lambda - \mathcal{C}_{\mathcal{X}_i} \rangle$ which is identical to $\langle \mathcal{X}_j, \lambda - \mathcal{C}_{\mathcal{X}_i} \rangle$. In this case, the attacker has the following choices: either to launch the remaining attack $\mathcal{X}_j$ (if λ is sufficient for it), or to discontinue playing the game. If $\mathcal{X}_j$ succeeds, the game moves into the state $\left\langle \mathbf{1}, \lambda - \mathcal{C}_{\mathcal{X}_i} - \mathcal{C}_{\mathcal{X}_j} \right\rangle$ and the adversary has won the game. In case $\mathcal{X}_j$ fails, the game moves into the state $\left\langle \mathcal{X}_j, \lambda - \mathcal{C}_{\mathcal{X}_i} - \mathcal{C}_{\mathcal{X}_j} \right\rangle$ and the game continues until the budget λ is sufficient to continue playing. Adversarial utility may be expressed in the form of the relation (10).

$$\mathcal{U}^{\lambda}(\mathcal{X}_i \wedge \mathcal{X}_j) = \max \begin{cases} 0 \\ -\mathcal{C}_{\mathcal{X}_i} + p_{\mathcal{X}_i} \mathcal{U}^{\lambda-\mathcal{C}_{\mathcal{X}_i}}(\mathcal{X}_j) + (1 - p_{\mathcal{X}_i})\, \mathcal{U}^{\lambda-\mathcal{C}_{\mathcal{X}_i}}(\mathcal{X}_i \wedge \mathcal{X}_j) \\ -\mathcal{C}_{\mathcal{X}_j} + p_{\mathcal{X}_j} \mathcal{U}^{\lambda-\mathcal{C}_{\mathcal{X}_j}}(\mathcal{X}_i) + (1 - p_{\mathcal{X}_j})\, \mathcal{U}^{\lambda-\mathcal{C}_{\mathcal{X}_j}}(\mathcal{X}_i \wedge \mathcal{X}_j) \end{cases} \tag{10}$$

where (according to (2)):

$$\mathcal{U}^{\lambda-\mathcal{C}_{\mathcal{X}_i}}(\mathcal{X}_j) = \mathcal{U}^{\infty}(\mathcal{X}_j) \left[1 - (1 - p_{\mathcal{X}_j})^{\left\lfloor \frac{\lambda - \mathcal{C}_{\mathcal{X}_i}}{\mathcal{C}_{\mathcal{X}_j}} \right\rfloor} \right] \ ,$$
$$\mathcal{U}^{\lambda-\mathcal{C}_{\mathcal{X}_j}}(\mathcal{X}_i) = \mathcal{U}^{\infty}(\mathcal{X}_i) \left[1 - (1 - p_{\mathcal{X}_i})^{\left\lfloor \frac{\lambda - \mathcal{C}_{\mathcal{X}_j}}{\mathcal{C}_{\mathcal{X}_i}} \right\rfloor} \right] \ .$$

In the atomic AND game the positive utility may not be achieved immediately by the adversary. We call the minimal value of the adversarial budget, sufficient to achieve positive utility the adversarial utility *budget lower bound*, which can be computed as:

$$\lambda_0 = \min \begin{cases} 0 \ , \\ \left\lceil \log_{(1-p_{\mathcal{X}_j})} \left[1 - \frac{\mathcal{C}_{\mathcal{X}_i}}{p_{\mathcal{X}_i}\, \mathcal{U}^\infty(\mathcal{X}_j)} \right] \right\rceil \cdot \mathcal{C}_{\mathcal{X}_j} + \mathcal{C}_{\mathcal{X}_i} \ , \\ \left\lceil \log_{(1-p_{\mathcal{X}_i})} \left[1 - \frac{\mathcal{C}_{\mathcal{X}_j}}{p_{\mathcal{X}_j}\, \mathcal{U}^\infty(\mathcal{X}_i)} \right] \right\rceil \cdot \mathcal{C}_{\mathcal{X}_i} + \mathcal{C}_{\mathcal{X}_j} \ . \end{cases} \tag{11}$$

Algorithm 4.2 outlines the recursive procedure to calculate maximal adversarial utility in the atomic AND game given budget λ according to (10).

ALGORITHM 4.2: Maximal utility of the atomic AND case with the given budget

```
Input: Attack step $\mathcal{X}_i$ cost i_cost
Input: Attack step $\mathcal{X}_i$ probability i_pr
Input: Attack step $\mathcal{X}_j$ cost j_cost
Input: Attack step $\mathcal{X}_j$ probability j_pr
Input: Prize of the game prize
Input: Budget budget
Output: Maximal adversarial utility value (a real number)
1  Procedure AtomicAnd (i_cost, i_pr, j_cost, j_pr, prize, budget)
2  if budget is less than the sum of i_cost and j_cost then
3      return (0)
4  i_inf := prize − $\frac{\text{i\_cost}}{\text{i\_pr}}$
5  j_inf := prize − $\frac{\text{j\_cost}}{\text{j\_pr}}$
6  i_rep := i_inf · $\left[1 - (1 - \text{j\_pr})^{\left\lfloor \frac{\text{budget-i\_cost}}{\text{j\_cost}} \right\rfloor}\right]$
7  j_rep := j_inf · $\left[1 - (1 - \text{i\_pr})^{\left\lfloor \frac{\text{budget-j\_cost}}{\text{i\_cost}} \right\rfloor}\right]$
8  ui = -i_cost + i_pr · j_rep + (1-i_pr) · AtomicAnd (i_cost, i_pr, j_cost, j_pr, prize,
   budget-i_cost)
9  if ui is negative then
10     ui := 0
11 uj = -j_cost + j_pr · i_rep + (1-j_pr) · AtomicAnd (i_cost, i_pr, j_cost, j_pr, prize,
   budget-j_cost)
12 if uj is negative then
13     uj := 0
14 if ui is not less than uj then
15     maximal_utility := ui
16 else
17     maximal_utility := uj
18 return (maximal_utility)
```

We show how the best move changes in the atomic AND game, depending on the current budget λ demonstrating it by several examples.

The first example (Fig. 8) shows that there are certain sets of parameters which make the adversary indifferent in whether to launch attack step $\mathcal{X}_i$ or attack step $\mathcal{X}_j$ in every state of the game.

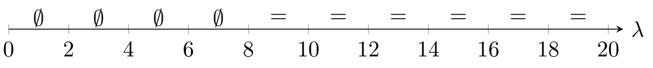


Fig. 8. Atomic AND case with parameters $\mathcal{C}_{\mathcal{X}_i} = 2, p_{\mathcal{X}_i} = 0.05, \mathcal{C}_{\mathcal{X}_j} = 6, p_{\mathcal{X}_j} = 0.9, Prize = 30$.

The second example (Fig. 9) demonstrates the case when the best move bounces between attack step $\mathcal{X}_i$ and attack step $\mathcal{X}_j$. In some states of the game both of the attack steps are equally optimal to launch.

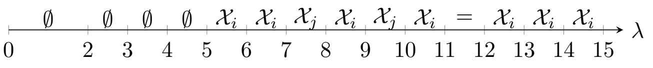

∅ ∅ ∅ ∅ $\mathcal{X}_i$ $\mathcal{X}_i$ $\mathcal{X}_j$ $\mathcal{X}_i$ $\mathcal{X}_j$ $\mathcal{X}_i$ = $\mathcal{X}_i$ $\mathcal{X}_i$ $\mathcal{X}_i$
λ
0 2 3 4 5 6 7 8 9 10 11 12 13 14 15

Fig. 9. Atomic AND case with parameters $\mathcal{C}_{\mathcal{X}_i} = 2, p_{\mathcal{X}_i} = 0.3, \mathcal{C}_{\mathcal{X}_j} = 3, p_{\mathcal{X}_j} = 0.48, Prize = 30$.

The third example (Fig. 10) demonstrates the case when the costs of the attacks are irrational, but their relation may be expressed in terms of a fraction of rational numbers. It can be seen that with the given parameters optimal strategy will suggest to iterate attack step $\mathcal{X}_j$ and thus the optimal strategy is non-adaptive.

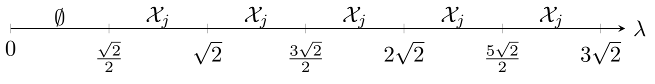

∅ $\mathcal{X}_j$ $\mathcal{X}_j$ $\mathcal{X}_j$ $\mathcal{X}_j$ $\mathcal{X}_j$
λ
0 $\frac{\sqrt{2}}{2}$ $\sqrt{2}$ $\frac{3\sqrt{2}}{2}$ $2\sqrt{2}$ $\frac{5\sqrt{2}}{2}$ $3\sqrt{2}$

Fig. 10. Atomic AND case with parameters $\mathcal{C}_{\mathcal{X}_i} = \sqrt{2}, p_{\mathcal{X}_i} = 0.8, \mathcal{C}_{\mathcal{X}_j} = \frac{\sqrt{2}}{2}, p_{\mathcal{X}_j} = 0.45, Prize = 30$.

The next example (Fig. 11) demonstrates the case when the optimal strategy is adaptive and the best move to undertake in a certain state of the game alternates between attack step $\mathcal{X}_i$ and attack step $\mathcal{X}_j$.

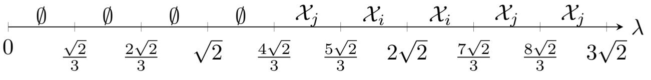


Fig. 11. Atomic AND case with parameters $\mathcal{C}_{\mathcal{X}_i} = \sqrt{2}, p_{\mathcal{X}_i} = 0.1, \mathcal{C}_{\mathcal{X}_j} = \frac{\sqrt{2}}{3}, p_{\mathcal{X}_j} = 0.38, Prize = 30$.

Comparison with the Improved Failure-Free Model

We will investigate the case when the improved failure-free model analysis result states that the system is *insecure*, while the budgeted model result states that the system is *secure*. According to the improved failure-free model the adversarial utility $\mathcal{U}^\infty(\mathcal{X}_i \wedge \mathcal{X}_j) = \mathcal{P} - \frac{\mathcal{C}_{\mathcal{X}_i}}{p_{\mathcal{X}_i}} - \frac{\mathcal{C}_{\mathcal{X}_j}}{p_{\mathcal{X}_j}}$. The system is secure in case $\mathcal{P} \leqslant \frac{\mathcal{C}_{\mathcal{X}_i}}{p_{\mathcal{X}_i}} + \frac{\mathcal{C}_{\mathcal{X}_j}}{p_{\mathcal{X}_j}}$ and insecure in case $\mathcal{P} > \frac{\mathcal{C}_{\mathcal{X}_i}}{p_{\mathcal{X}_i}} + \frac{\mathcal{C}_{\mathcal{X}_j}}{p_{\mathcal{X}_j}}$.

Let the adversarial budget λ suffice to launch m attack steps in total and the adversarial strategy may be the one as shown in Fig. 12 and for the sake of simplicity lets assume that $\mathcal{C}_{\mathcal{X}_i} = \mathcal{C}_{\mathcal{X}_j} = \mathcal{C}$ and $p_{\mathcal{X}_i} = p_{\mathcal{X}_j} = p$.

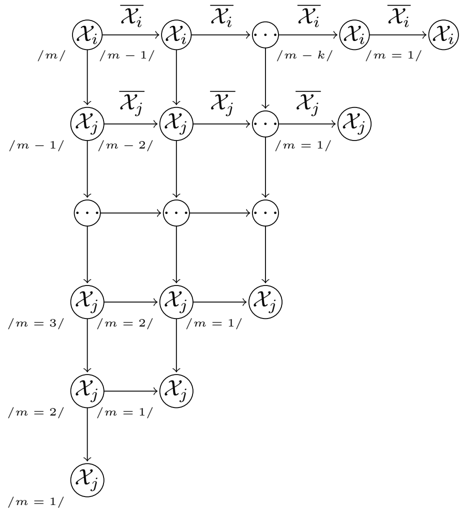


Fig. 12. An adaptive strategy consisting of two attack steps $\mathcal{X}_i$ and $\mathcal{X}_j$, with adversarial budget λ.

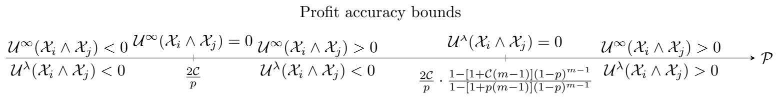
Adversarial utility may in this case be computed as shown in (12).

$$\begin{aligned}\mathcal{U}^{m\times\mathcal{C}}(\mathcal{X}_i\wedge\mathcal{X}_j) &= \left[\mathcal{U}^{\infty}(\mathcal{X}_j)-\frac{\mathcal{C}}{p}\right]\left[1-(1-p)^{m-1}\right]-(m-1)(1-p)^{m-1}\left[p\,\mathcal{U}^{\infty}(\mathcal{X}_j)\right] \\ &= \left[\mathcal{P}-\frac{2\mathcal{C}}{p}\right]\left[1-(1-p)^{m-1}\right]-(m-1)(1-p)^{m-1}\left[p\,\mathcal{P}-\mathcal{C}\right]\end{aligned} \tag{12}$$

According to the budgeted model the strategy is not profitable for an attacker, while the improved failure-free model states that the strategy is profitable if:

$$\frac{2\mathcal{C}}{p} < \mathcal{P} \leqslant \frac{2\mathcal{C}}{p}\cdot\frac{1-[1+\mathcal{C}(m-1)]\,(1-p)^{m-1}}{1-[1+p(m-1)]\,(1-p)^{m-1}}\ . \tag{13}$$

Inequality (13) shows the interval for the value of prize within which the result of the limited budget model and result of the improved failure-free models differ. We will show what happens to the results of the analysis of both models in the broader view.



Profit accuracy bounds
$\mathcal{U}^{\infty}(\mathcal{X}_i\wedge\mathcal{X}_j)<0$
$\mathcal{U}^{\lambda}(\mathcal{X}_i\wedge\mathcal{X}_j)<0$
$\mathcal{U}^{\infty}(\mathcal{X}_i\wedge\mathcal{X}_j)=0$
$\frac{2\mathcal{C}}{p}$
$\mathcal{U}^{\infty}(\mathcal{X}_i\wedge\mathcal{X}_j)>0$
$\mathcal{U}^{\lambda}(\mathcal{X}_i\wedge\mathcal{X}_j)<0$
$\mathcal{U}^{\lambda}(\mathcal{X}_i\wedge\mathcal{X}_j)=0$
$\frac{2\mathcal{C}}{p}\cdot\frac{1-[1+\mathcal{C}(m-1)](1-p)^{m-1}}{1-[1+p(m-1)](1-p)^{m-1}}$
$\mathcal{U}^{\infty}(\mathcal{X}_i\wedge\mathcal{X}_j)>0$
$\mathcal{U}^{\lambda}(\mathcal{X}_i\wedge\mathcal{X}_j)>0$
$\mathcal{P}$

Fig. 13. Comparison of the improved failure-free model to the limited budget model.

Thus, Fig. 13 shows that if prize is less than $\frac{2\mathcal{C}}{p}$ then the system is *secure* according to both models. If prize is greater than $\frac{2\mathcal{C}}{p}\cdot\frac{1-[1+\mathcal{C}(m-1)](1-p)^{m-1}}{1-[1+p(m-1)](1-p)^{m-1}}$ then the system is *insecure* according to both models. Only when the prize is between $\frac{2\mathcal{C}}{p}$ and $\frac{2\mathcal{C}}{p}\cdot\frac{1-[1+\mathcal{C}(m-1)](1-p)^{m-1}}{1-[1+p(m-1)](1-p)^{m-1}}$ the limited budget model may produce result different from the result of the improved failure-free model.

We have experimented with various parameters and observed that the prize interval (13) becomes negligibly small – less than 1 €. In practice, as a rule, it is practically impossible to estimate the value of the protected assets with the precision of less than €1 and for this reason we think that the limited budget model may produce false-positive results in case analysts are unable to estimate prize with required precision and this makes us give preference to the failure-free models which provides reliable utility upper bounds.

Table 1 demonstrates an example of such calculations. It can be seen that already with rather small increase in budget (approximately 3 times greater than the costs of the attack steps) the prize must be estimated with precision less than €1 in order to ensure reliability of the results.

The first column in a table describes the monetary budget of the adversary. The second column describes the interval for possible prize values, the column named span shows the length of such an interval. Precision is the length of uncertainty interval divided by mean value.

Table 1. *Initial setting: Prize: €30 Cost: €2 Probability: 0.3*

Lambda (#)	$\mathcal{P}$ Domain (€)	Span (€)	Deviation (€)	Precision (%)
2	$(13.(3), 28.(8)]$	15.(5)	±7.(7)	0.518519
3	$(13.(3), 22.4074]$	9.07407	±4.537035	0.302469
4	$(13.(3), 19.242]$	5.9087	±2.95435	0.196957
5	$(13.(3), 17.4047]$	4.07139	±2.035695	0.135713
6	$(13.(3), 16.232]$	2.89863	±1.449315	0.0966211
7	$(13.(3), 15.4386]$	2.10531	±1.052655	0.0701772
8	$(13.(3), 14.8816]$	1.54822	±0.77411	0.0516073
9	$(13.(3), 14.4806]$	1.14723	±0.573615	0.038241

5 Conclusions and Future Research

We have analyzed the 3 kinds of elementary games – the single attack game, the atomic OR and the atomic AND, assuming that the adversarial budget is limited. In the result of limiting adversarial budget the model and computations become reasonably complex that makes it doubtful that this approach is applicable for real-life case analysis. Additionally, in case of atomic AND we have to be able to estimate the *prize* parameter quite precisely – if we fail to do that, the analysis results will be unreliable. In practice it is very hard to estimate the cost of an asset or information with the desired precision and thus is it doubtful if it is reasonable to face the complexities of budget limitations and its false positive results which might happen in the case of AND type games.

The improved failure-free model is, on the contrary, less complex and provides reliable upper bounds. Due to the fact that when the move fails the player finds himself in the very same instance of the game results in the existence of non-adaptive strategies in the set of optimal strategies of the game and the ordering of the attack steps in non-adaptive optimal strategies is irrelevant. In the model with budget limitations the subset of non-adaptive strategies exists in the set of all strategies. Non-adaptive strategies are relatively easy to derive and compute. One of the open questions is to figure out how well the most optimal strategy from the subset of non-adaptive strategies $\mathcal{U}^{\lambda}_{na}(\mathcal{G})$ might approximate the optimal strategy from the set of all possible strategies $\mathcal{U}^{\lambda}(\mathcal{G})$. If $\mathcal{U}^{\lambda}_{na}(\mathcal{G})$ provides pretty good approximation to $\mathcal{U}^{\lambda}(\mathcal{G})$, then there exists infinitely small α such that:

$$\mathcal{U}^{\lambda}_{na}(\mathcal{G}) \leqslant \mathcal{U}^{\lambda}(\mathcal{G}) \leqslant \alpha \cdot \mathcal{U}^{\lambda}_{na}(\mathcal{G}) \leqslant \mathcal{U}^{\infty}(\mathcal{G}) \ .$$

If this holds, it might enable calculation of acceptably precise result without facing the complexity and the computational overhead introduced by the precise utility calculation routines.

Secondly, it would be interesting to see when the optimal move in certain states of the game changes by bouncing between the two possible moves thus following some pattern. Additionally, to verify the hypothesis that this might happen in the theoretical case when the ratio of the costs of the move is irrational.

The bigger the adversarial budget λ is, the more adversarial utility approaches the utility upper bound in the improved failure-free model. Optimal strategies in the improved failure-free model are non-adaptive and do not depend on the ordering of the attack steps. In the case of big λ optimal strategies are likely to behave non-adaptively as well in the limited budget model. This means that optimal move in certain states of the game is likely to bounce changing from one attack to another, but with increase in λ the optimal move remains the same. It also means that the utility of various strategies, beginning with different moves, become closer to each other with the increase in λ and there should exist infinitely small δ such that

$$|\mathcal{U}^{\lambda}(\mathcal{S}_i) - \mathcal{U}^{\lambda}(\mathcal{S}_j)| \leqslant \delta \ ,$$

where $\mathcal{S}_i$ and $\mathcal{S}_j$ are the two strategies from the set of all strategies of the game.

The improved failure-free model provides reliable utility upper bounds, however this results in systems that might be over-secured. It has not been studied how much extra cost the upper-bound oriented methods cause. The assumption that the adversarial budget is limited is natural, as this is what happens in reality. Models assuming limited budget model the adversarial strategic decision making in a better way, which is more close to the one likely to be observed in real life and the research on the adaptive strategies with limited budget is an important research area in quantitative security analysis based on attack trees.

References

1. Buldas, A., Laud, P., Priisalu, J., Saarepera, M., Willemson, J.: Rational choice of security measures via multi-parameter attack trees. In Lpez, J., ed.: CRITIS. Volume 4347 of Lecture Notes in Computer Science., Springer (2006) 235–248
2. Buldas, A., Lenin, A.: New efficient utility upper bounds for the fully adaptive model of attack trees. In Das, S.K., Nita-Rotaru, C., Kantarcioglu, M., eds.: GameSec. Volume 8252 of Lecture Notes in Computer Science., Springer (2013) 192–205
3. Buldas, A., Stepanenko, R.: Upper bounds for adversaries' utility in attack trees. In Grossklags, J., Walrand, J.C., eds.: GameSec. Volume 7638 of Lecture Notes in Computer Science., Springer (2012) 98–117
4. Jrgenson, A., Willemson, J.: Computing exact outcomes of multi-parameter attack trees. In Meersman, R., Tari, Z., eds.: OTM Conferences (2). Volume 5332 of Lecture Notes in Computer Science., Springer (2008) 1036–1051
5. Jrgenson, A., Willemson, J.: On fast and approximate attack tree computations. In Kwak, J., Deng, R.H., Won, Y., Wang, G., eds.: ISPEC. Volume 6047 of Lecture Notes in Computer Science., Springer (2010) 56–66
6. Mauw, S., Oostdijk, M.: Foundations of attack trees. In Won, D., Kim, S., eds.: ICISC. Volume 3935 of Lecture Notes in Computer Science., Springer (2005) 186–198
7. Schneier, B.: Attack trees. Dr. Dobb's Journal of Software Tools **24**(12) (December 1999) 21–22, 24, 26, 28–29

Appendix E

Attacker Profiling in Quantitative Security Assessment based on Attack Trees

Attacker profiling in quantitative security assessment based on attack trees

Aleksandr Lenin[1,2], Jan Willemson[1] and Dyan Permata Sari[2] ⋆

[1] Cybernetica AS, Mäealuse 2/1, Tallinn, Estonia
[2] Tallinn University of Technology, Ehitajate tee 5, Tallinn, Estonia

Abstract. Providing meaningful estimations for the quantitative annotations on the steps of complex multi-step attacks is hard, as they are jointly influenced by the infrastructure and attacker properties. The paper introduces *attacker profiling* as the concept of separation of the infrastructure properties from the properties of malicious agents undertaking strategic decisions in the considered environment. We show that attacker profiling may be integrated into existing quantitative security assessment tools without any significant performance penalty. As an example of such integration we introduce the new analysis tool named ApproxTree+ which is an extension of the existing ApproxTree tool, enhancing it by incorporating attacker profiling capabilities into it.

1 Introduction

Targeted malicious attacks are intentional by their nature and may be interpreted as sequences of actions (attack steps) performed by malicious agents undertaking informed strategic decisions in the target infrastructure. This way we can distinguish between the two landscapes – the one which we call the *threat landscape* and the *vulnerability landscape*. The threat landscape is formed by various kinds of malicious agents – they have different sets of properties, available resources, varying intentions, motivations, views, and expectations of the target infrastructure. These properties determine strategic preferences of the agents, and eventually their behavior. The vulnerability landscape is formed by the infrastructure of the organization, its employees, assets, policies, processes, etc. Both landscapes are dynamic by their nature and are constantly changing. The threat landscape may change due to the agent behavior (e.g.

⋆ This research was supported by the European Regional Development Fund through Centre of Excellence in Computer Science (EXCS), the Estonian Research Council under Institutional Research Grant IUT27-1 and European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement ICT-318003 (TREsPASS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.

increase in resources available to the agent) as well as external events, while the vulnerability landscape may change due to the infrastructure updates (e.g. patching, component replacement, awareness training, deployment of defensive measures, etc.) as well as unintentional events.

We propose the separation between the infrastructure properties (the vulnerability landscape) and the adversarial properties (the threat landscape), represented by an *attacker profile*. This separation adds flexibility to the quantitative security analysis enabling the assessment of operational security risks using different combinations of attacker profiles and infrastructure properties providing much deeper insight on the surrounding risk landscape. Besides, attacker profiling increases the reliability of the analysis results as the separation of infrastructure properties and attacker properties allows to update these values in a timely manner independently from each other and reflect the ever changing risk landscape in a more reliable way.

The paper aims at introducing attacker profiling in the context of quantitative security analysis based on attack trees and demonstrates integration of attacker profiling into existing security assessment tools introducing the new tool named ApproxTree+. In the introduced ApproxTree+ model the considered infrastructure properties (cost, difficulty, minimal required attack time) are quantitative annotations on the attack tree leaves, while the adversarial properties (budget, skill, available time) are described by attacker profiles. Additionally we compare the performance of the profiling computations to the ApproxTree approach [1] and reassess if the genetic algorithm parameters, used by ApproxTree for fast approximations, are optimal for the profiling computations.

The outline of the paper is the following: Section 2 outlines the state of the art in quantitative security assessment, attack trees, and attacker profiling. Section 3 describes motivation for the attacker profiling in security risk assessment. Section 4 introduces the ApproxTree+ tool, while Section 5 outlines the tool performance analysis results. Section 6 briefly lists the achievements made so far and outlines areas for future research.

2 Related Work

2.1 Attack trees

Attack trees as one of the ways of quantitative security assessment, evolved from fault trees [2] and were popularized by Schneier [3] who suggested to use them as a way to model security threats and to perform quantitative security assessment using this convenient hierarchical representation by means of bottom-up single parameter propagation. Quantitative security assessment has been studied by various researchers [4–8] and different variations of techniques and methodologies were suggested.

Buldas *et al.* [9] suggested to use multi-parameter approach instead of the historical single-parameter one and applied economic reasoning by propagating adversarial utility. This kind of analysis allowed to assess whether the analyzed system is secure against targeted rational profit-oriented attacks.

Jrgenson and Willemson improved the model of Buldas *et al.*, making their parallel [10] and serial [11] models consistent with Mauw and Oostijk foundations [12] and introducing genetic approach to speed up computations. The parallel model assumed that the attacker launches attack steps, required to fulfil the attack scenario, simultaneously, while the serial model assumed that an attacker launches the attack steps in a predefined order.

Later, Buldas and Stepanenko introduced the failure-free model [13] suggesting not to limit the adversary in any way and thus analyzing fully adaptive adversarial utility upper bounds. This approach was later improved by Buldas and Lenin [14]. Their model better conforms to the upper bounds ideology and is computationally less complex.

For a more thorough overview of the quantitative security analysis using attack trees we refer the reader to [15].

2.2 Attacker profiling

Back in 1998, Philips *et al.* [16] outlined the importance of the attacker feature in attack graphs for network vulnerability analysis. Several research projects have focused on attacker profiling using honeypot in Know Your Enemies series [17–19] which outlined the range of techniques and tools that were used by attackers for reconnaissance and also motives of the blackhat community. Several researchers proposed the concept of attacker personas, which was related to goal, motivations, attitudes, and skills [20–23]. Faily *et al.* highlighted insider threat motivations and characteristics, as well as the use of attacker personas for threat identification.

Pardue *et al.* [24] mentioned the importance of attacker characteristics and also the complexity of the attacks assessing risks of an e-voting system. The authors argue that the likelihood of attacks can be referred to as *cost* of an attacker, which can be estimated on various scales and measured in various units, such as dollars, number of attackers, time invested into attacking, and effort. In addition, Sallhammar *et al.* [25]

demonstrate the process of deriving the probability of the expected attacker behavior in assumption that the attacker has complete information about the vulnerabilities of the targeted systems. Tipton *et al.* [26] argue that risk aversion, degree of difficulty, discoverability, ease of access, effectiveness of controls, effort, incentive, interest, skill level, motivation, resources required, risk of detection, and special equipment needed are the factors that can be included in attacker profiling. There are some common parameters that are most often used in research projects to define an attacker profile – these values are more feasible for quantitative analysis and give clear understanding of attacker properties.

2.3 Parallel model

The parallel model [10] by Jürgenson *et al.* allows to assess whether the analyzed system is secure against targeted rational profit-oriented attacks by assessing adversarial utility. In case the utility is positive, the system is considered to be insecure, as profitable attack vectors which may result in positive outcome for an attacker are likely to exist. Otherwise the analysis assumed that the system is reasonably secure to withstand emerging attacks.

An attack scenario, represented by an attack tree, is treated as a monotone Boolean function, each variable of which corresponds to a leaf node in the attack tree, and logic operators correspond to the refined nodes in the attack tree. The successful outcome of an elementary attack is modelled by assigning value 1 to the corresponding variable in the Boolean function. If the Boolean function is satisfied, the attacker has succeeded in the security scenario. More complex multi-step attacks are modelled as *attack suites*.

The computational method maximizes the adversarial utility over the entire set of satisfying attack suites. The complexity of the approach arises from the need to process the entire set of 2^n attack suites, which introduces unnecessary overhead. Even with the optimizations proposed [10] this approach was able to analyze attack trees of at most 20 leaves in reasonable time which has made this method inapplicable for the practical case analysis.

To overcome limitations of the parallel model [10], a set of further optimizations was proposed by Jürgenson *et al.* [1] and implemented in the tool later called ApproxTree.

More significant contribution of the paper is the development of genetic algorithm for fast approximations, which increased performance compared to [10]. The implementation of the approach described in the paper reached 89% confidence[1] level within 2 seconds of computation for the tree having up to 29 leaves. As the genetic algorithm is very scalable it has potential to be used for the analysis of practical attack trees containing more than 100 leaves. The computational complexity of the suggested approximation algorithm in the worst case was estimated to be $\mathcal{O}(n^4)$. The authors have performed benchmarking tests and experimentally derived the optimal set of values for genetic algorithm parameters.

3 Motivation for the attacker profiling

An attack tree is a hierarchical description of possible attacks against the target infrastructure. Constructing an attack tree, analysts include all possible attack scenarios in the tree. Some of them are more realistic, some are less, considering the environment in which such a system is deployed. This way, attack tree analysis assumes an overpowered attacker who is capable of launching every possible attack, included in the attack tree, against the system. However, real life attacks are, as a rule, not so powerful and thus analysis assuming the almighty adversary concept does not provide deep insight on the security risks taking into account the surrounding risk landscape. Applying attacker profile to the attack tree invalidates certain nodes and eventually entire subtrees in the initial attack tree, thus enabling the independent analysis of the derived attack scenarios, containing attacks feasible for the considered class of malicious agents. Depending on the severity of adversarial limitations used in the profile, the derived attack scenario may be much smaller and thus much easier to analyze.

Quantitative security analysis relies on quantitative annotations (e.g. likelihood of success in an attack step, time required to launch an attack step, etc.) assigned to single attack steps in complex multi-step attacks. We believe that the quantitative metrics of these annotations is jointly influenced by various sets of underlying components in threat- as well as vulnerability landscapes. Thus it is rather difficult to provide a trustworthy and reliable quantitative estimation for such parameters as it is practically impossible to estimate the cumulative effect of several underlying factors altogether. Such kind of joint estimations are, as a rule, imprecise and contain reasonable degree of uncertainty.

For example, it is almost impossible to provide a meaningful estimation for the time parameter, as the time, required for an attack step,

[1] By confidence authors mean the ratio of the trees actually computed correctly by the suggested approximation technique, compared to the precise outcome.

depends on the attacker skills, capabilities, available resources, previous experience, etc. (agent properties), as well as on the difficulty of the attack step itself (infrastructure property). Similarly, the likelihood of success depends on attacker skill, difficulty of the attack step, and time invested into attacking. The more skilful and experienced the attacker is, the more likely he is to succeed in an attack step. The more resources are available to the attacker, the more likely will he be successful in an attack step. Similar reasoning may be applied to the skill parameter – the more experienced the attacker is, the less difficult is the process for him, the less time it will take to succeed in an attack step. Less skilled attacker, given sufficient time, may be as efficient (in terms of likelihood of success) as a more skilled attacker who has less time for attacking. Similar logic may be applied to other parameters as well.

Despite that, the analysis has to deal somehow with the ever changing nature of each of the landscapes mentioned above and update (or reassess) the estimations of the corresponding quantitative annotations in a timely manner. It is unclear how to update such joint estimations in case some of its components change while the others remain unchanged, or, on the contrary, when all its components change.

In order to tackle the difficulties outlined above the propose attacker profiling as a step forward in dealing with the challenges of security metrics.

4 The ApproxTree+ model

We introduce the ApproxTree+ model – the new model for quantitative assessment of operational security risks. The computational method is built on the logic of the parallel attack tree model [10] and fast approximations of ApproxTree [1], improved by adding attacker profiling considerations into the method.

4.1 Definitions

We will use the same notation as in [10]. Let us have a set of all possible elementary attacks $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n\}$, and a Boolean function $\mathcal{F}$ corresponding to the attack tree.

Definition 1 (Attack Suite). *Attack suite $\sigma \subseteq \mathcal{X}$ is a set of elementary attacks which have been chosen by the attacker to be launched and used to try to achieve the attacker goal.*

Definition 2 (Satisfying attack suite). *A satisfying attack suite σ evaluates $\mathcal{F}$ to* true *when all the elementary attacks from the attack suite σ have been evaluated to* true*.*

Definition 3 (Attacker profile). *An attacker profile is a pair $(t, \mathcal{P}_t)$ where t is an n-tuple of attacker properties $(p_1, p_2, \ldots, p_n)$ and a function $\mathcal{P}_t(\sigma)$ defined by t which takes an attack suite σ as input and returns* true*, iff the attacker with the considered properties t is capable of launching all the attacks in σ, and* false *otherwise.*

Each of the elements p_k in t belongs to a certain domain P_k which provides quantitative metrics to the parameter. Some of the domains may represent continuous values, e.g. money, so we can take $P_k = \mathbb{R}$. Others parameters may be measured on an ordinal scale to reflect the magnitude and measured in levels e.g. High, Medium, and Low, in which case $P_k = \{H, M, L\}$.

In our research we use the following attacker properties:

1. *Budget* $t_b \in \mathbb{R}$ – the monetary resource of the attacker, measured in currency units.
2. *Skill* $t_s \in \{L, M, H\}$ – the skill level of the attacker, measured on an ordinal scale (Low/Medium/High).
3. *Time* $t_a \in \{S, MT, HR, D\}$ – the available time resource of the attacker, measured on an ordinal scale (Seconds/Minutes/Hours/Days).

The attacker properties outlined above define function $\mathcal{P}_f(\sigma)$, which returns *true* iff:

1. $t_b \geqslant \sum_{i=1}^{n} \mathrm{Cost}(\mathcal{X}_i)$,
2. $\forall \mathcal{X}_i \in \sigma : \ t_s \geqslant \mathrm{Difficulty}(\mathcal{X}_i)$, and
3. $\forall \mathcal{X}_i \in \sigma : \ t_a \geqslant \mathrm{Time}(\mathcal{X}_i)$.

Definition 4 (Profile satisfying attack suite). *A profile satisfying attack suite σ is a satisfying attack suite which satisfies all the constraints of the chosen attacker profile $(t, \mathcal{P}_t)$.*

4.2 Description of the approach

The analysis method can be described by the following rules [10]:

1. The attacker constructs the attack tree and evaluates the parameters of each of the elementary attacks following these considerations:

   - The attacker has to spend $Cost_i$ resources to prepare and launch an attack $\mathcal{X}_i$.
   - The attack $\mathcal{X}_i$ succeeds with probability p_i and fails with probability $1 - p_i$.
   - Depending on the detective security measures, the attacker sometimes has to carry additional costs after failing or succeeding with the attack. The sum of preparation and additional costs is denoted as $Expenses_i$ parameter.
   - Additionally, there is global parameter $Profit$ for the whole attack scenario, which describes the benefit of the attacker, in case the root node is achieved.
2. The attacker considers all potential attack suites – subsets $\sigma \subseteq \mathcal{X}$, where $\mathcal{X} = \{\mathcal{X}_1, \ldots, \mathcal{X}_n\}$ is the set of all elementary attacks considered in the attack scenario. Some of the attack suites satisfy the Boolean function $\mathcal{F}$, some do not. For the satisfying attack suites the attacker computes the outcome value $Outcome_\sigma$.
3. Finally, the attacker chooses the most profitable attack suite and launches the corresponding elementary attacks simultaneously.

The computational method presented in [10] aims at maximizing the expression

$$Outcome_\sigma = p_\sigma \cdot Profit - \sum_{\mathcal{X}_i \in \sigma} Expenses_i$$

over all the assignments $\sigma \subseteq \mathcal{X}$ that turn the monotone Boolean function $\mathcal{F}$ to true. The success probability of the primary threat p_σ can be computed in time linear in the size of elementary attacks n:

$$p_\sigma = \sum_{\substack{\mathcal{R} \subseteq \sigma \\ \mathcal{F}(\mathcal{R} := true) = true}} \prod_{\mathcal{X}_i \in \mathcal{R}} p_i \prod_{\mathcal{X}_j \in \sigma \setminus \mathcal{R}} (1 - p_j) \ . \tag{1}$$

In order to tackle the potential exponential amount of computations in (1), a genetic algorithm was proposed and benchmarked by Jürgenson *et al.* [1].

4.3 Approximation

The ApproxTree+ method uses the genetic algorithm to facilitate the usage of the computational method for large attack trees:

1. Create the first generation of n individuals (profile satisfying attack suites, not all of them are necessarily distinct).

2. All the individuals in the initial population are crossed with everybody else producing $\binom{n}{2}$ new individuals.
3. Each individual is mutated with probability p.
4. The mutated population is joined with the initial population.
5. Finally, n fittest profile satisfying individuals out of the $\binom{n}{2} + n$ individuals are selected and form the next generation.

The reproduction phase terminates when k last generations do not increase outcome. The complexity of the suggested approach was measured to be approximately $\mathcal{O}(0.85^n)$ using exponential regression.

5 Performance analysis

In order to assess the performance of the introduced computational method we have randomly generated a set of attack trees. The attack tree generation procedure was a two-step process. First, the random Boolean function with the predefined number of variables (leaves in the attack tree) was generated. It contained from 2 to 5 operands per operator – the values of operands in each case were chosen randomly. The next step was to provide quantitative annotations on the leaves of the attack tree. These values were chosen randomly from the predefined intervals: the *cost* parameter was estimated in the interval $[100, 1000]$, the *success probability* parameter was estimated in the interval $(0, 1)$. The value for the *difficulty* parameter was chosen from uniformly distributed values *low*, *medium*, *high*, and *very high*. The value for the *time* parameter was chosen from uniformly distributed values *seconds*, *minutes*, *hours*, and *days*[2].

One of the questions that needs to be answered is if attacker profiling adds extra computational overhead. It can be seen on the cumulative time distribution diagram (see Fig. 1) that attacker profiling does not add any significant computational overhead (in the case of a single attacker profile being analyzed) compared to the ApproxTree approach (see Fig. 2). In both methods the initial population generation phase is almost immediate, as well as the mutation phase. The main workload is performed by the crossover phase and consumes approximately 85-99% of the cumulative time distribution among all the phases. The last phase, the best individuals selection phase, does not introduce any significant workload and consumes approximately 1 - 15%. The crossover phase is the most time consuming as each individual is crossed with every other individual in the population producing $\mathcal{N} \times \mathcal{N}$ cross operations, where

[2] assuming uniform distribution of the PRNG output

$\mathcal{N}$ is the amount of individuals in the initial population. Fig. 3 shows that the execution time of the ApproxTree+ approach is proportional to the ApproxTree approach. The increased execution time arises from the fact that, as a rule, one doesn't assess risks using just a single adversarial profile, as it is reasonable to assess risks using the entire set of possible adversarial profiles so that the results would produce meaningful insight on the risk landscape – thus the overall execution time is proportional to the number of the attacker profiles under consideration.

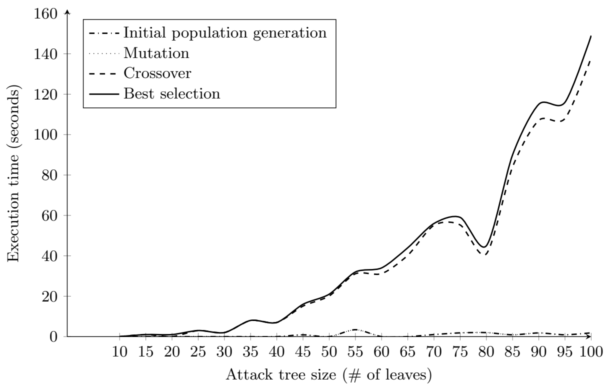


Fig. 1. Cumulative time distribution of ApptoxTree+ phases.

The analysis of the speed of convergence shows that the convergence speed of ApproxTree does not exceed the convergence speed of ApproxTree+. Additionally, it does not depend on the size of the attack tree – independently of the size of the tree, the convergence speed stays approximately at the same level.

Additionally, we have analyzed the effect of the genetic algorithm parameters such as mutation rate and initial population size on the convergence speed to assess whether the parameters of the genetic algorithm used by ApproxTree [1] are optimal for the ApproxTree+ approach.

The convergence speed decreases with the increase in the percentage of mutations from approximately 2 generations in the case when the mutation rate is 10% up to 6 generations in the case when mutation rate is 90% (see Fig. 5). Independently of the mutation rate, the speed of con-

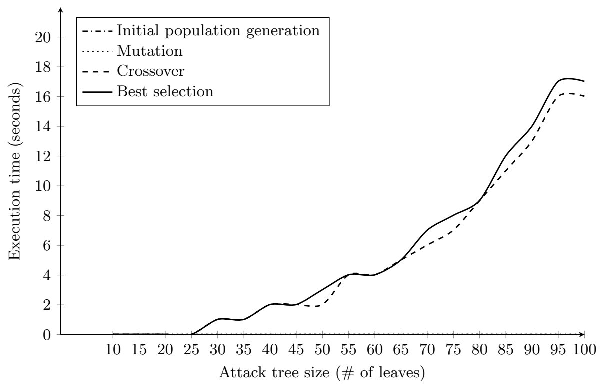


Fig. 2. Cumulative time distribution of ApproxTree phases.

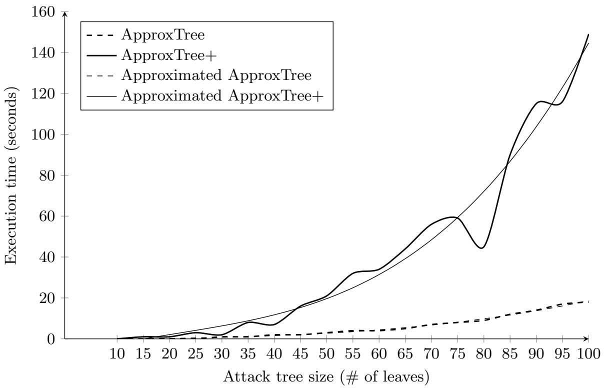


Fig. 3. Execution time.

vergence of ApproxTree+ does not exceed the speed of convergence of ApproxTree.

Benchmarking results have shown that the mutation step has no significant effect on the convergence speed at all. We were unable to find any case where the method would get stuck in the local optimum. Even when the mutation step was excluded entirely (as a phase of the genetic algorithm) – the global optimum was always reached. This may happen because of "good" initial population generation – if the size of the

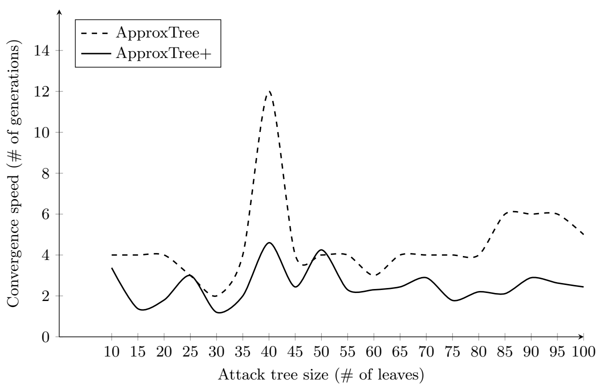


Fig. 4. Convergence speed.

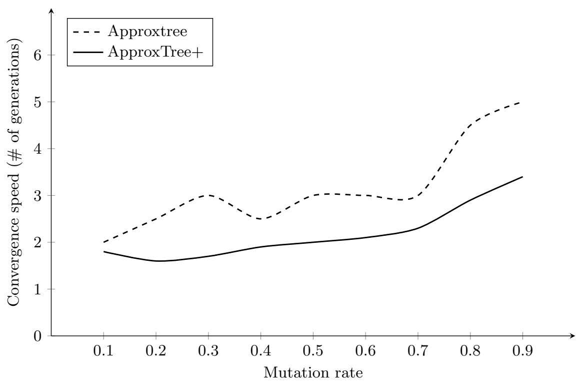


Fig. 5. Convergence speed as a function of the mutation rate.

initial population is rather big compared to the number of satisfying solutions, it is highly likely that the initial population will contain all the solutions (profile satisfying attack suites). In this case the convergence is immediate, which was observed in some cases during benchmarking. If the initial population does not contain all the solutions, still it may be "good enough" so that the crossover step produces the entire domain of solutions.

With the increase in the initial population size (see Fig. 6) the convergence speed increases, stabilising at a value of approximately 1.6 genera-

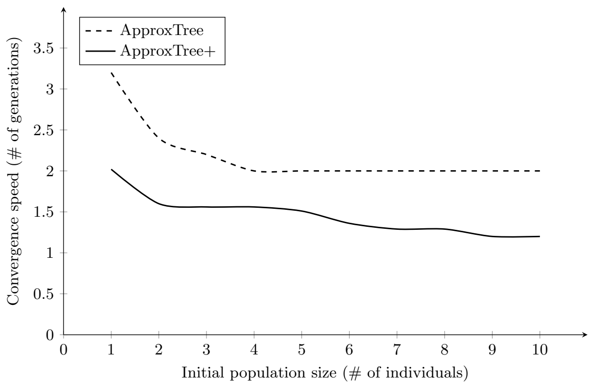


Fig. 6. Convergence speed as a function the size of the initial population.

tions for the initial population size greater than $4n$ (n being the number of leaves in the attack tree) in the case of the ApproxTree approach. In the case of ApproxTree+ we can see slight, but firm decrease in the convergence speed. In some cases when initial population size was less than n the computational method was unable to reach global optimum, which may happen when rather small initial population limits the amount of possible solutions that may be reached and the mutation rate is small enough and does not improve the situation.

The precision assessment shows that in ApproxTree+, as well as in ApproxTree, either the result converges to the global optimum (most profitable attack suite) or the computational method fails to generate the initial population of individuals. In case of profiling, the attacker profile may contain so strict constraints that not a single profile satisfying attack suite may exist. The more strict constraints are used in the considered attacker profiles the higher is the probability that no profile satisfying assignments will be generated. However we are unable to state that the profile satisfying solutions definitely do not exist in this case, as the state when ApproxTree+ is unable to generate the initial population means 2 possible conditions – either no profile satisfying attack suites exist (and thus the considered attack scenario has no profitable solutions), or such attack suites exist, however the attack suite generation procedure failed to generate profile satisfying solutions due to the stochastic nature of the process.

6 Conclusions and Future Research

Attacker profiling is a way to separate infrastructure properties and the properties of the malicious agents who are undertaking strategic decisions in the target infrastructure. This kind of separation allows to estimate and assess these properties independently from one another. This allows to derive meaningful values for the quantitative annotations on the attack steps in complex multi-step attacks from the underlying properties instead of providing joint estimations to these values directly. One can more precisely estimate how would a complex value change in case when some of its underlying components change. In example, how would the likelihood of success in an attack step change if instead of profit-oriented malicious individuals we face organized groups of attackers or a national security agency and the target infrastructure was patched meanwhile and the employees have received an awareness training? Thus, attacker profiling enables more detailed assessment of the impact of the fluctuations in threat and vulnerability landscapes on the values of the quantitative annotations on the attack steps.

Additionally, it adds flexibility to the analysis in general, enabling analysis using different combinations of attacker profiles and infrastructure properties, making comprehensive risk assessment possible. It provides broader and more detailed overview of the risk landscape in a timely manner, following constant changes in the risk environment. It allows to make informed decisions in assessing the cost-effectiveness of the defensive measures and enabling the prediction, prioritization and prevention of emerging attacks in nearly semi-automated way.

We introduced the attacker profiling and demonstrated the application of profiling in the framework of attack tree analysis by introducing the new analysis tool named ApproxTree+ and demonstrating that integrating attacker profiling into an existing analysis method does not introduce any significant performance penalty.

The constraint based approach, outlined in the paper, is only one possible interpretation of attacker profiling. Another possibility is to apply Item Response Theory to represent the relation between various underlying components in the threat and vulnerability landscapes. Such a relation may be represented, in example, in the form of a logistic function in its simplest form indicating that the likelihood of success will be assigned value 0.5 when the skill (β) and difficulty (γ) are equal: $p = \left(e^{\beta-\delta}\right)/(1+e^{\beta-\delta})$. In more complex scenarios the function may be extended to take 3 arguments, as the likelihood of success depends on the invested time parameter as well, and in this case it will take the form of: $p = f(\beta, \delta, \gamma)$ where γ is the time invested into attacking.

We see the way forward in implementing the above mentioned interpretation of profiling, integrating ApproxTree+ in the existing risk assessment frameworks and tools, and validating the approach in real-case risk analysis.

References

1. Jrgenson, A., Willemson, J.: On Fast and Approximate Attack Tree Computations. In Kwak, J., Deng, R.H., Won, Y., Wang, G., eds.: ISPEC. Volume 6047 of Lecture Notes in Computer Science., Springer (2010) 56–66
2. Vesely, W.E., Goldberg, F.F., Roberts, N.H., Haasl, D.F.: Fault Tree Handbook. U.S. Nuclear Regulatory Commission, Washington, DC (1981)
3. Schneier, B.: Attack trees. Dr. Dobb's Journal of Software Tools **24**(12) (December 1999) 21–22, 24, 26, 28–29
4. Schumacher, M.: Security Engineering with Patterns - Origins, Theoretical Models, and New Applications. Volume 2754 of Lecture Notes in Computer Science. Springer (2003)
5. Miede, A., Nedyalkov, N., Gottron, C., Knig, A., Repp, N., Steinmetz, R.: A Generic Metamodel for IT Security. In: ARES, IEEE Computer Society (2010) 430–437
6. Kishor S. Trivedi, Dong Seong Kim, A.R., Medhi, D.: Dependability and Security Models. In: Proceedings of the 7th IEEE International Workshop on the Design of Reliable Communication Networks (DRCN), Washington, DC (October 2009) 11–20
7. Schneier, B.: Secrets & Lies: Digital Security in a Networked World. 1st edn. John Wiley & Sons, Inc., New York, NY, USA (2000)
8. Barbara Kordy and Sjouke Mauw and Saša Radomirović and Patrick Schweitzer: Attack–Defense Trees. Journal of Logic and Computation **24**(1) (2014) 55–87
9. Buldas, A., Laud, P., Priisalu, J., Saarepera, M., Willemson, J.: Rational Choice of Security Measures Via Multi-parameter Attack Trees. In Lpez, J., ed.: CRITIS. Volume 4347 of Lecture Notes in Computer Science., Springer (2006) 235–248
10. Jrgenson, A., Willemson, J.: Computing Exact Outcomes of Multi-parameter Attack Trees. In Meersman, R., Tari, Z., eds.: OTM Conferences (2). Volume 5332 of Lecture Notes in Computer Science., Springer (2008) 1036–1051
11. Jrgenson, A., Willemson, J.: Serial Model for Attack Tree Computations. In Lee, D., Hong, S., eds.: ICISC. Volume 5984 of Lecture Notes in Computer Science., Springer (2009) 118–128
12. Mauw, S., Oostdijk, M.: Foundations of Attack Trees. In Won, D., Kim, S., eds.: ICISC. Volume 3935 of Lecture Notes in Computer Science., Springer (2005) 186–198
13. Buldas, A., Stepanenko, R.: Upper Bounds for Adversaries' Utility in Attack Trees. In Grossklags, J., Walrand, J.C., eds.: GameSec. Volume 7638 of Lecture Notes in Computer Science., Springer (2012) 98–117
14. Buldas, A., Lenin, A.: New Efficient Utility Upper Bounds for the Fully Adaptive Model of Attack Trees. In Das, S.K., Nita-Rotaru, C., Kantarcioglu, M., eds.: GameSec. Volume 8252 of Lecture Notes in Computer Science., Springer (2013) 192–205

15. Kordy, B., Pietre-Cambacedes, L., Schweitzer, P.: DAG-Based Attack and Defense Modeling: Don't Miss the Forest for the Attack Trees. CoRR **abs/1303.7397** (2013)
16. Phillips, C., Swiler, L.P.: A Graph-based System for Network-vulnerability Analysis. In: Proceedings of the 1998 Workshop on New Security Paradigms. NSPW '98, New York, NY, USA, ACM (1998) 71–79
17. series, K.Y.E.: Honeynet Project. Know Your Enemy The Tools and Methodologies of the Script Kiddie. http://project.honeynet.org (jul 2000)
18. series, K.Y.E.: Honeynet Project. Know Your Enemy II: Tracking the blackhat's moves. http://project.honeynet.org (jun 2001)
19. series, K.Y.E.: Honeynet Project. Know Your Enemy III: They Gain Root. http://project.honeynet.org (mar 2000)
20. Blomquist, A., Arvola, M.: Personas in action: ethnography in an interaction design team. In: Proceedings of the second Nordic conference on Human-computer interaction. NordiCHI '02, New York, NY, USA, ACM (2002) 197–200
21. Castro, J.W., Acua, S.T., Juzgado, N.J.: Integrating the Personas Technique into the Requirements Analysis Activity. In Gelbukh, A.F., Adiba, M.E., eds.: ENC, IEEE Computer Society (2008) 104–112
22. Faily, S., Flechais, I.: Barry is not the weakest link: eliciting secure system requirements with personas. In McEwan, T., McKinnon, L., eds.: BCS HCI, ACM (2010) 124–132
23. Faily, S., Flechais, I.: Persona cases: a technique for grounding personas. In Tan, D.S., Amershi, S., Begole, B., Kellogg, W.A., Tungare, M., eds.: CHI, ACM (2011) 2267–2270
24. Pardue, H., Landry, J., Yasinsac, A.: A Risk Assessment Model for Voting Systems using Threat Trees and Monte Carlo Simulation. In: Requirements Engineering for e-Voting Systems (RE-VOTE), 2009 First International Workshop on. (2009) 55–60
25. Sallhammar, K., Knapskog, S.J., Helvik, B.E.: Building a Stochastic Model for Security and Trust Assessment Evaluation. http://q2s.ntnu.no/publications/open/2005/Mass_media/2005_sallhammar_BSM.pdf (oct 2005)
26. Tipton, H., Baker, P.: Official (ISC)2 guide to the CISSP CBK. In: Official (ISC)2 guide to the CISSP CBK. (2010)

Appendix F

TRE$_S$PASS: Plug-and-Play Attacker Profiles for Security Risk Analysis

TRE$_S$PASS: Plug-and-Play Attacker Profiles for Security Risk Analysis

Wolter Pieters*†, Dina Hadžiosmanović*, Aleksandr Lenin‡, Lorena Montoya†, and Jan Willemson‡
*Delft University of Technology, The Netherlands, {w.pieters, d.hadziosmanovic}@tudelft.nl
†University of Twente, The Netherlands, a.l.montoya@utwente.nl
‡Cybernetica AS, Estonia, {aleksandr.lenin,janwil}@cyber.ee

***Abstract*—Existing methods for security risk analysis typically estimate time, cost, or likelihood of success of attack steps. When the threat environment changes, such values have to be updated as well. However, the estimated values reflect both system properties and attacker properties: the time required for an attack step depends on attacker skill as well as the strength of a particular system component. In the TRE$_S$PASS project, we propose the separation of attacker and system properties. By doing so, we enable "plug-and-play" attacker profiles: profiles of adversaries that are independent of system properties, and thus can be re-used in the same or different organisation to compare risk in case of different attacker profiles. We demonstrate its application in the framework of attack trees, as well as our new concept of attack navigators.**

I. Introduction

Providing meaningful metrics for operational security risk is hard [1]. There are different reasons for this: ever-changing threat landscape, difficulties in validation and updating of risk estimations, inability of performing comprehensive quantification of the threat environment, etc. We believe that the difficulty of dealing with these challenges, among other reasons, is due to properties of the *system* and properties of the *threat* having not been separated in common risk analyses in organisations.

Existing approaches for security risk analysis use estimates of time and cost to evaluate attack steps. For example, it is said that a particular attack step costs \$ 10,000 or has a 0.2 likelihood of success. Such annotations can then be used for calculating the properties of complex, multi-step attacks from the values associated with the individual steps (e.g., in the framework of attack trees [2], [3], [4], [5]). For example, if access to sensitive data first requires cracking a password and then exploiting a vulnerability, and both have 0.2 likelihood of success, the likelihood of success of the overall attack is 0.04 (assuming independence and single attempts).

The problem arises when the threat and/or vulnerability landscape changes, which is a realistic scenario in dynamic organisations. The threat environment can change due to agent behaviour (e.g., increase in attacker resources), while the vulnerability landscape can change due to infrastructure updates (e.g., applying patches to decrease system vulnerability, but also unintentional events). In either case, the estimated annotations need to be updated. However, these values reflect jointly both system and agent properties: the time required for an attack step depends on attacker skill as well as difficulty of the step. By using a joint estimation, it is unclear how to update the values if only one of the components changes. For example, one may have assigned a 0.2 likelihood of success assuming a script kiddie as attacker. However, how should this value be updated if one faces a national security agency instead and the system has been patched in the meantime?

In the context of atomic threat events, some standards already acknowledge the distinction between attacker and system properties. For example, the FAIR risk taxonomy [6] distinguishes between Threat Capability and Control Strength to determine likelihood of success. However, FAIR does not consider changing conditions in environments with multi-step attacks. In earlier work, we proposed the use of Item Response Theory to take both attacker and system properties into account in quantitative penetration testing [7], [8].

In the ongoing TRE$_S$PASS project (www.trespass-project.eu), we primarily focus on the risk analysis perspective. In particular, we tackle the challenge of operational security risk metrics by analysing multi-step attacks in the context of complex socio-technical systems. We see our work as a step forward in dealing with challenges of security metrics.

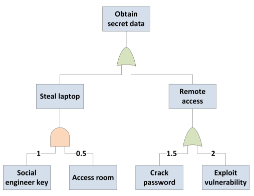
II. Attack navigators

The separation of attacker and system properties enables running risk calculations for different combinations of attacker profiles and system configurations, resulting in comprehensive risk analyses that are further evaluated by the organisation. Thus, we enable "plug-and-play" attacker profiles: profiles of adversaries that are independent of system properties, implying that (1) attacker profiles can be used for different systems, and (2) the risk analysis of a system can be done with a different attacker profile without the need for updates of time, cost or likelihood values. To demonstrate this approach, the TRE$_S$PASS project developed the concept of *attack navigator*, which consists of a *map* of the system components and properties (e.g., by using socio-technical annotations such as system configuration, user policies, network access controls), and an attacker profile which is traversing the *map*. Our tool simulates situations in which different attackers may have different goals, skills and resources, and may therefore prefer different attack paths on the map. Combinations of attacker profiles may be used to reflect the threat environment of a system, and these can be updated when needed. In such a case, a new picture emerges of the risk situation of the organisation based on the new threat environment.

To illustrate the approach, we show how attacker profiles reflect on calculations of the likelihood of success of attacks. For this we use attack trees, an industry standard for adversarial analysis. Our attack navigator tool is able to generate attack trees from maps of the system, based on a chosen target asset. We focus on the situation where attacker properties (e.g., skill) and system properties (e.g., difficulty) together determine the likelihood of success of an attack step. Different functions are possible to denote such a relation:

- a constraint-based approach, indicating that the attacker should have a skill level at least as high as the difficulty (as an extension of [2]);
- a logistic function, indicating that the likelihood of success is 0.5 when skill (β) and difficulty (δ) are equal (as in Item Response Theory [7]): in its simplest form $P = (e^{\beta-\delta})/(1 + e^{\beta-\delta})$.

In attack trees, the system properties (e.g., difficulty) will be annotations on the leaves in the tree. The attacker properties (e.g., skill) will be included in the attacker profiles. When a particular attacker profile is selected, the likelihood of success can be determined for each node based on the combination of difficulty and skill. The resulting likelihood of success can then be used in traditional attack tree calculations, as well as in security risk analysis. Fig. 1 shows in a simplified scenario how to calculate risk properties for an attacker with skill 1. To adapt to changing environments, the same calculations can be performed with different skill levels.



Attacker skill (β): 1
Constraint-based: only left branch (AND-node) is feasible.
Logistic model: the likelihoods of success of the leaf nodes are respectively 0.5; 0.62; 0.38; 0.27. The all-in likelihood of success for the left branch is 0.5 * 0.62 = 0.31. For this attacker, cracking the password would be the most attractive option (i.e., 0.38).

Fig. 1. Example plug-and-play attack tree analysis. The bottom left node is an AND-node; others are OR-nodes. The attack tree is annotated with difficulty (δ) of the steps.

In the project, we work with more complex (multi-parameter) infrastructure maps and attacker profiles. One way to further extend the analysis is to use a three-parameter function to take the dependency between invested time and success into account. One can then derive likelihood of success from (1) attacker skill, (2) step difficulty, and (3) time invested by the attacker. We are currently working on such extensions, based on timed probability distributions [9].

III. Plug-and-Play in TRE$_{\text{S}}$PASS

One of the main bottlenecks of practical security risk analysis is the unclear attribution of properties to the threat environment (attackers) or the system being analysed. In this work, we have presented a way forward based on results from the TRE$_{\text{S}}$PASS project. In particular, we analyse how attacker profiles can be used as plug-and-play components in the risk analysis. In the broader context of the project, we envision that different properties can be used as plug-and-play components: user profiles, system configurations, etc. For example, user profiles could be used for evaluating the likelihood of success of social engineering steps of the attacks, which could differ depending on the cultural environment.

To derive meaningful maps of complex socio-technical systems, we leverage different techniques. Next to scalable formal modelling techniques, several visualisation techniques have been developed in TRE$_{\text{S}}$PASS to support model development and analysis. This ranges from specific representations of importance of branches in attack trees to physical system maps built by stakeholders using *Lego*. Such "thinking tools" are essential in capturing the relevant knowledge about system architecture, potential attackers, and associated parameters.

Acknowledgment

The research leading to these results has received funding from the European Union's Seventh Framework Programme (FP7/2007-2013) under grant agreement ICT-318003 (TRE$_{\text{S}}$-PASS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.

References

[1] V. Verendel, "Quantified security is a weak hypothesis: A critical survey of results and assumptions," in *Proceedings of the 2009 New Security Paradigms Workshop*. New York, NY, USA: ACM, 2009, pp. 37–50. [Online]. Available: http://doi.acm.org/10.1145/1719030.1719036

[2] A. Buldas and A. Lenin, "New efficient utility upper bounds for the fully adaptive model of attack trees," in *Decision and Game Theory for Security*, ser. LNCS, S. K. Das, C. Nita-Rotaru, and M. Kantarcioglu, Eds., vol. 8252. Springer, 2013, pp. 192–205. [Online]. Available: http://dx.doi.org/10.1007/978-3-319-02786-9_12

[3] B. Kordy, P. Kordy, S. Mauw, and P. Schweitzer, "ADTool: Security analysis with attack-defense trees," in *Quantitative Evaluation of Systems*, ser. LNCS, K. Joshi, M. Siegle, M. Stoelinga, and P. D'Argenio, Eds. Springer, 2013, vol. 8054, pp. 173–176. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40196-1_15

[4] S. Mauw and M. Oostdijk, "Foundations of attack trees," in *Proc. 8th Annual International Conference on Information Security and Cryptology, ICISC'05*, ser. LNCS, D. Won and S. Kim, Eds., vol. 3935. Springer, 2006, pp. 186–198. [Online]. Available: http://www.icisc.org/

[5] B. Schneier, "Attack trees: Modeling security threats," *Dr. Dobb's journal*, vol. 24, no. 12, pp. 21–29, 1999.

[6] The Open Group, "Risk taxonomy," The Open Group, Tech. Rep. C081, 2009. [Online]. Available: www.opengroup.org/pubs/catalog/c081.htm

[7] W. Pieters, S. H. G. Van der Ven, and C. W. Probst, "A move in the security measurement stalemate: Elo-style ratings to quantify vulnerability," in *Proceedings of the 2012 workshop on New security paradigms*, ser. NSPW '12. New York, NY, USA: ACM, 2012, pp. 1–14. [Online]. Available: http://doi.acm.org/10.1145/2413296.2413298

[8] F. Arnold, W. Pieters, and M. Stoelinga, "Quantitative penetration testing with item response theory," in *Proceedings of Information Assurance and Security (IAS) 2013*. IEEE, 2013.

[9] F. Arnold, A. Belinfante, F. Berg, D. Guck, and M. Stoelinga, "Dftcalc: A tool for efficient fault tree analysis," in *Computer Safety, Reliability, and Security*, ser. LNCS, F. Bitsch, J. Guiochet, and M. Kaâniche, Eds. Springer, 2013, vol. 8153, pp. 293–301. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40793-2_27

Appendix G

Genetic Approximations for the Failure-Free Security Games

Genetic Approximations for the Failure-Free Security Games

Aleksandr Lenin[1,2], Jan Willemson[1] and Anton Charnamord[1,2] ⋆

[1] Cybernetica AS, Mäealuse 2/1, Tallinn, Estonia
[2] Tallinn University of Technology, Ehitajate tee 5, Tallinn, Estonia

Abstract. This paper deals with computational aspects of attack trees, more precisely, evaluating the expected adversarial utility in the failure-free game, where the adversary is allowed to re-run failed atomic attacks an unlimited number of times. It has been shown by Buldas and Lenin that exact evaluation of this utility is an NP-complete problem, so a computationally feasible approximation is needed. In this paper we consider a genetic approach for this challenge. Since genetic algorithms depend on a number of non-trivial parameters, we face a multi-objective optimization problem and we consider several heuristic criteria to solve it.

1 Introduction

Hierarchical methods for security assessment have been used for several decades already. Called *fault trees* and applied to analyze general security-critical systems in early 1980-s [1], they were adjusted for information systems and called *threat logic trees* by Weiss in 1991 [2]. In the late 1990-s, the method was popularized by Schneier under the name *attack trees* [3].

There are several ways attack trees can be used in security assessment. The simplest way is purely descriptional. Such an approach is limited only to qualitative assessment of security. Based on such an assessment, it is difficult to talk about optimal level of security or return of security investments. Already the first descriptions of attack trees introduced computational aspects [2, 3]. The framework for a sound formal model for such computations was introduced in 2005 by Mauw and Oostdijk [4].

Most of the earlier studies focus on the analysis of a single parameter only. A substantial step forward was taken by Buldas *et al.* [5] who

⋆ This research was supported by the European Regional Development Fund through Centre of Excellence in Computer Science (EXCS), the Estonian Research Council under Institutional Research Grant IUT27-1 and the European Union Seventh Framework Programme (FP7/2007-2013) under grant agreement ICT-318003 (TREsPASS). This publication reflects only the authors' views and the Union is not liable for any use that may be made of the information contained herein.

introduced the idea of game-theoretic modeling of the adversarial decision making process based on several interconnected parameters like the cost, risks and penalties associated with different atomic attacks. Their approach was later refined by Jürgenson and Willemson [6] to achieve compliance with Mauw-Oostdijk framework [7]. However, increase in the model precision was accompanied by significant drop in computational efficiency. To compensate for that, a genetic algorithm approach was proposed by Jürgenson and Willemson [8]. It was later shown by Lenin, Willemson and Sari that this approach is flexible enough to allow extensions like attacker models [9].

Buldas and Stepanenko [10] introduced the upper bound ideology by pointing out that in order to verify the security of the system, it is not necessary to compute the exact adversarial utility but only upper bounds. Buldas and Lenin further improved the fully adaptive model by eliminating the force failure states and suggested the new model called the failure-free model [11]. The model more closely followed the upper bounds ideology originally introduced by Buldas *et al.* [10] and turned out to be computationally somewhat easier to analyze. It has been shown that finding the optimal strategy is (still) an NP-complete problem, hence looking for a good heuristic approximation is an important goal. Additionally, one of the goals of the paper is to find empirical evidence for the rational choice of the parameters of the genetic algorithm.

The paper has the following structure. First, Section 2 defines the required terms. Section 3 presents and evaluates our genetic algorithms. These algorithms are improved with adaptiveness in Section 4. Finally, Section 5 draws some conclusions.

2 Definitions

Let $\mathcal{X} = \{\mathcal{X}_1, \mathcal{X}_2, \ldots, \mathcal{X}_n\}$ be the set of all possible atomic attacks and $\mathcal{F}$ be a monotone Boolean function corresponding to the considered attack tree.

Definition 1 (Attack Suite). *Attack suite $\sigma \subseteq \mathcal{X}$ is a set of atomic attacks which have been chosen by the adversary to be launched and used to try to achieve the attacker's goal. Also known as individual.*

Definition 2 (Satisfying attack suite). *A satisfying attack suite σ evaluates $\mathcal{F}$ to* true *when all the atomic attacks from the attack suite σ have been evaluated to* true*. Also known as live individual.*

Definition 3 (Satisfiability game). *By a satisfiability game we mean a single-player game in which the player's goal is to satisfy a monotone Boolean function $\mathcal{F}(x_1, x_2, \ldots, x_k)$ by picking variables x_i one at a time and assigning $x_i = 1$. Each time the player picks the variable x_i he pays some amount of expenses $\mathcal{E}_i$, which is modeled as a random variable. With a certain probability p_i the move x_i succeeds. The game ends when the condition $\mathcal{F} \equiv 1$ is satisfied and the player wins the prize $\mathcal{P} \in \mathbb{R}$, or when the condition $\mathcal{F} \equiv 0$ is satisfied, meaning the loss of the game, or when the player stops playing. Thus we can define three common types of games:*

1. *SAT Game Without Repetitions - the type of a game where a player can perform a move only once.*
2. *SAT Game With Repetitions - the type of a game where a player can re-run failed moves an arbitrary number of times.*
3. *Failure-Free SAT Game - the type of a game in which all success probabilities are equal to 1. It has been shown that any game with repetitions is equivalent to a failure-free game [11, Thm. 5].*

3 Genetic Approximations for the Failure-Free Satisfiability Games

The whole family of satisfiability games tries to maximize expected adversarial profit by solving an optimization problem: given a monotone Boolean function $\mathcal{F}(x_1, x_2, \ldots, x_n)$ optimize the utility function $\mathcal{U}(x_{i_1}, x_{i_2}, \ldots, x_{i_n})$ over the set of all satisfying assignments fulfilling a set of model-specific conditions (in some specific cases). The models for the SAT games without move repetitions and the failure-free SAT games differ only by their corresponding utility functions, as in both cases the order in which atomic attacks are launched by an adversary is irrelevant. On the contrary, models for SAT games with repetitions (e.g. [12]) consider strategic adversarial behavior in the case of which the order in which the atomic attacks are launched does matter. In this paper we focus on the genetic approximations suitable to be applied to the SAT games without repetitions, as well as the failure-free SAT games. The suggested algorithm is practically validated by the example of the computational model for the failure-free SAT game.

3.1 Genetic algorithm (GA)

A genetic algorithm is typically characterized by the set of the following parameters: a genetic representation of chromosomes or individuals (feasible solutions for the optimization problem), a population of encoded solutions, fitness function which evaluates the optimality of the solutions, genetic operators (selection, crossover, mutation) that generate a new population from the existing one, and control parameters (population size, crossover rate, mutation rate, condition under which the reproduction process terminates).

The reproduction process, as well as the condition, under which reproduction terminates is identical to the one described in [9]. We refer the readers to this paper for further details. An individual is any feasible solution to the considered optimization problem. Thus, for the SAT games a solution is any of the *satisfying attack suites*. We have chosen linear binary representation of individuals to facilitate the robustness of the crossover and mutation operations. The algorithm used to generate individuals is shown in Algorithm 1.

Algorithm 1: Recursive individual generation algorithm

```
Data: The root of a propositional directed acyclic graph (PDAG) representing
      a monotone Boolean function. An empty individual with all bits set to 0.
Result: Live individual.
if the root is a leaf then
    get the index of the leaf;
    set corresponding individual's bit to 1;
end
else if the root is an AND node then
    forall the children of the root do
        recursive call: child considered as root parameter;
    end
end
else if the root is an OR node then
    choose at least one child;
    forall the chosen children do
        recursive call: child considered as root parameter;
    end
end
```

We allow duplicate entries to be present in the population for the sake of maintaining genetic variation and keep the population size constant throughout the reproduction process. It is well known in the field of genetic algorithms that genetic variation directly influences the chances of premature convergence – thus increasing genetic variation in the population is one of the design goals.

The choice of the population size is important – too small population does not contain enough genetic variation to maintain the exploration capabilities, too big population already contains enough genetic variation to efficiently explore the search space and only results in the performance overhead in the crossover operator. This means that there exists an optimal population size corresponding to the minimal population size capable of producing the best result. Thus the optimal size of the population sets the lower bound of reasonable choice for the population size and the upper bound is solely based on performance considerations – what is the reasonable time the analysts would agree to wait for the analysis to produce the result. If the population size is suboptimal, there is a high risk to converge to suboptimal solutions and if the population is bigger than the optimal size it does not add anything, except for the increase in the time required to run the analysis. If the optimal population in some certain case is $k\%$ of the size of the attack tree (the number of leaves in an attack tree), then any population size greater than $k\%$ and capable of producing the result in reasonable time, would suit to be used for analysis.

All the following computations were made with PC/Intel Core i5-4590 CPU @ 3.30 GHz, 8 GB RAM, Windows 8.1 (64 bit) operating system. Fig. 1 on the left demonstrates the effect of the population size on the result in the case of a single attack tree. Measurements were taken for the attack tree with 100 leaves using uniform crossover operator and mutation rate 0.1.

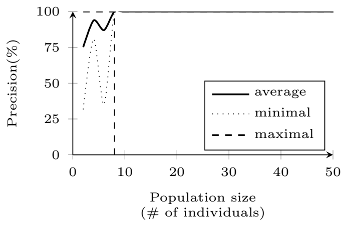


Fig. 1: Optimal population size

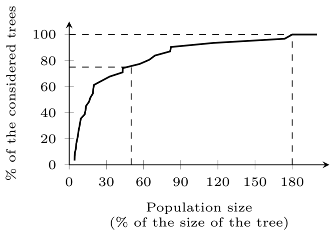


Fig. 2: Reasonable choice for population size

We have conducted experiments on the set of attack trees of different sizes (ranging from 10 to 100 leaves with steps of size 3) and observed that there is no obvious relation between the size of the analyzed tree and the optimal population size. Apart from the size of the tree, the optimal population size might depend on, at the very least, the structure of the tree itself. Measurements were taken with the same crossover operator and mutation rate. Fig. 2 shows how many trees (%) from the conducted experiment the considered population size would fit. It can be seen that, in general, the population size equal to 180% of the size of the tree would fit every considered attack tree. The population size 200%, chosen by Jürgenson and Willemson in [8] for their ApproxTree model, was a reasonable choice.

Lenin, Willemson and Sari have shown that the crossover operations take 90-99% of the time required to run the analysis [9]. Fig. 3 shows the time measurement for the suggested GA, depending on the size of the population.

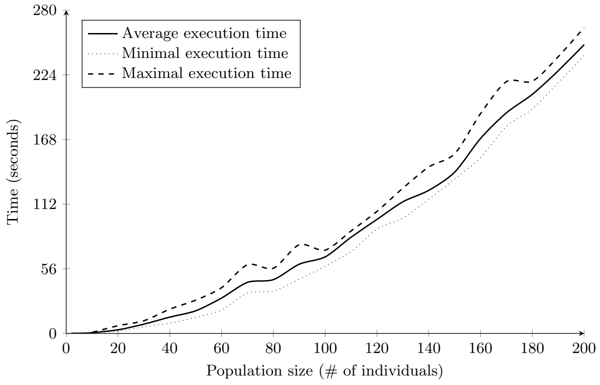


Fig. 3: Population size effect on GA execution time.

The *fitness function* is the model-specific utility function for the corresponding type of the security game. For further details we refer the reader to the detailed descriptions of the security games [7–11].

The power of GA arises from crossover which causes randomized but still structured exchange of genetic material between individuals in assumption that 'good' individuals will produce even better ones. The *crossover rate* controls the probability at which individuals are subjected to crossover. Individuals, not subjected to crossover, remain unmodified. The higher the crossover rate is, the quicker the new solutions get introduced into the population. At the same time, chances increase for the solutions to get disrupted faster than selection can exploit them. The selection operator selects individuals for crossing and its role is to direct the search towards promising solutions. We have chosen to disable parent selection entirely thus defaulting to crossing every individual with every other individual in the population (crossover rate equal to 1), as scalable selection pressure comes along with the selection mechanisms after reproduction.

Notable crossover techniques include the single-point, the two-point, and the uniform crossover types. Figures 4 and 5 demonstrate the differences between the convergence speeds resulting from using various crossover operators. It can be seen that the considered crossover operators do not have any major differences nor effect on the convergence speed of the GA.

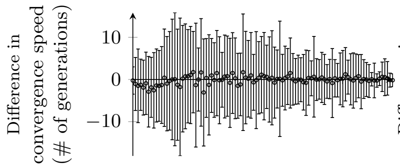


Fig. 4: Uniform crossover compared to single point crossover

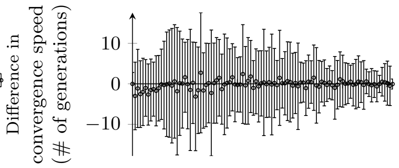


Fig. 5: Uniform crossover compared to two point crossover

Our choice fell upon using the uniform crossover – this enables a more exploratory approach to crossover than the traditional exploitative approach, resulting in a more complete exploration of the search space with maintaining the exchange of good information. The algorithm for the crossover operator is shown in Algorithm 2.

Algorithm 2: The uniform crossover operation

```
Data: The population of individuals represented as a sorted set.
Result: The population with new added individuals, created during the
        crossover operation.
initialize a new set of individuals;
forall the individual i in the population do
    forall the individual j different from i do
        new individual := the result of cross operation between individuals i
        and j;
        if new individual is alive then
            add the new individual to the set of new individuals;
        end
    end
end
add the set of new individuals to the population;
```

The role of the mutation operator is to restore lost or unexplored genetic material into the population thus increasing the genetic variance and preventing premature convergence to suboptimal solutions. The mutation rate controls the rate at which 'genes' are subjected to mutation. High levels of mutation rate turn GA into a random search algorithm, while too low levels of mutation rates are unable to restore genetic material efficiently enough, thus the algorithm risks converging to suboptimal solutions. Typically the mutation rate is kept rather small, in the range $0.005 - 0.05$.

In our implementation of the genetic algorithm, the mutation operator is a part of the crossover operation, mutating the genes, having same value in the corresponding positions in both parent individuals. The uniform crossover randomly picks corresponding bits in the parent individuals to be used in the new individual, and thus in the case bits are different, this already provides sufficient genetic variation. However, in the case when bits have the same value this yields just a single choice and in order to increase the genetic variation (compared to its parents) we mutate just these bits. Fig. 6 demonstrates the mutation rate effect on the utility function for the case of a specific attack tree with 100-leaves with initial population of 50 individuals. It shows that when the mutation rate exceeds value 0.1 GA turns into a random search algorithm, thus it is reasonable to keep the mutation rate rather small. We have conducted similar experiments on a larger set of attack trees and the results have shown that the optimal value for the mutation rate is not necessarily small – in some cases the optimal mutation rate was 0.6 or even higher. This means that the optimal value for the mutation rate cannot be set from the very beginning – it highly depends on the structure of the fitness landscape. However, it is still reasonable to follow the general rule of thumb to keep the mutation rate small, assuming that this should work for the majority of the cases.

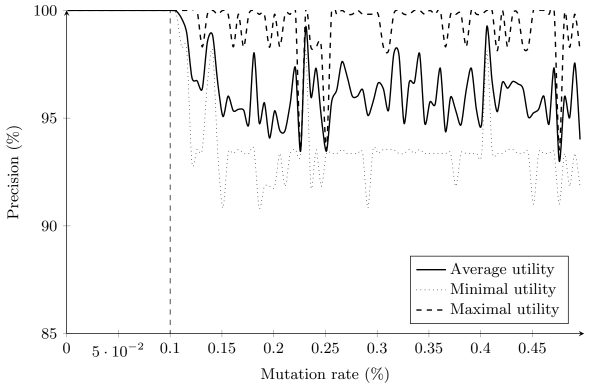


Fig. 6: GA mutation rate effect

It is important to determine the practical applicability boundaries for the suggested method. By practical applicability we mean the maximal size of the attack tree, which the computational method is capable of analyzing in reasonable time set to two hours. Extrapolating the time consumption curve in Fig. 7 we have come to a conclusion that theoretically the suggested GA is capable of analyzing attack trees containing up to 800 leaves in reasonable time. This is a major advancement compared to the ApproxTree model [8] which would take more than 900 hours to complete such a task.

The execution time complexity estimations for GA are outlined in Table 1.

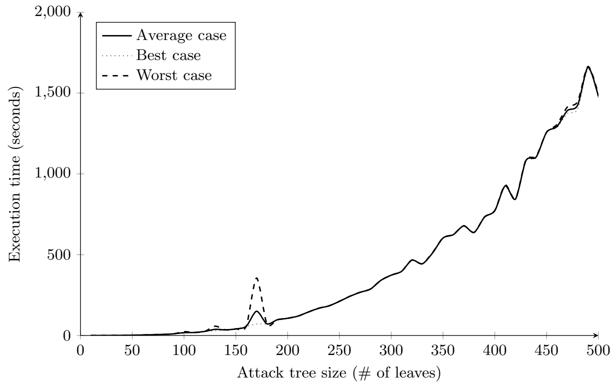


Fig. 7: GA execution time

Table 1: GA execution time complexity estimations

Case	Approximation polynomial	R^2 coefficient
Worst	$1.68 \cdot 10^{-5}n^3 - 0.003n^2 + 0.7015n - 23.03$	0.99
Average	$1.41 \cdot 10^{-5}n^3 - 0.001n^2 + 0.25n - 8.81$	0.99
Best	$1.26 \cdot 10^{-5}n^3 + 1.62 \cdot 10^{-5}n^2 + 0.047n - 2.55$	0.99

For comparison, the execution time complexity of the ApproxTree model [8] was estimated to be $\mathcal{O}(n^4)$, where n is the number of leaves in the attack tree. This difference comes from the fact that ApproxTree runs for a fixed number of generations, whereas the computations presented in this paper run until local convergence, as well as the fact that the utility function used in ApproxTree is considerably more complex compared to the corresponding utility function used in the Failure-Free model.

4 Adaptive Genetic Algorithm (AGA)

We compare the genetic algorithm suggested in Section 3 to the adaptive genetic approach described in [13]. The authors suggest to adaptively vary the values of crossover and mutation rates, depending on the fitness values of the solutions in the population. High fitness solutions are 'protected' and solutions with subaverage fitness are totally disrupted. It was suggested to detect whether the algorithm is converging to an optimum by evaluating the difference between the maximal and the average fitness values in the population $f_{\max} - \bar{f}$ which is likely to be less for the population which is converging to an optimum solution than for a population scattered across the solution space. Thus the corresponding values of the mutation and crossover rates are increased when the algorithm is converging to an optimum and decreased when the population gets too scattered. The authors concluded that the performance of AGA is in general superior to the performance of GA but varies considerably from problem to problem. In this paper we apply the suggested method to the problem of the security games.

In the case of the adaptive genetic algorithm, the crossover and mutation rate parameters are assigned their initial values and are changed adaptively during the runtime of the algorithm and the only parameter which remains fixed is the population size. Similarly to the GA there exists an optimal population size corresponding to the minimal population size capable of producing the maximal result. Fig. 8 shows the result corresponding to the computations using various population sizes in the experiment setup similar to the one for GA. In the case of GA the maximal value was stable with the increase in the population size, however in the case of AGA some fluctuations are present. Fig. 9 shows how many trees (%) from the conducted experiment the considered population size would fit. It can be seen that, in general, the population size equal to 200% of the size of the tree would fit every considered attack tree. Based on these observations we can say that AGA seems to be more robust, but less stable, compared to GA and requires bigger population sizes in order to produce optimal results for the majority of the cases.

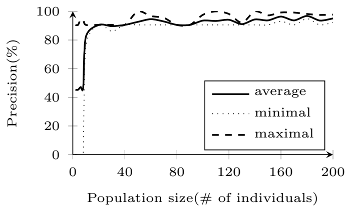


Fig. 8: Optimal population size

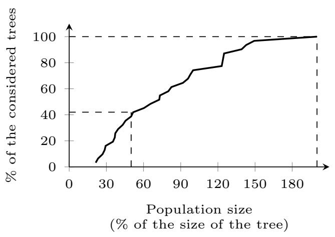


Fig. 9: Reasonable choice for population size

Similarly to the GA, we estimate the maximal size of the attack tree which AGA is capable of analyzing within reasonable timeframe set to two hours. Extrapolating the time consumption curve with the most extreme values trimmed out in Fig. 10 we have come to a conclusion that theoretically AGA is capable of analyzing attack trees containing up to

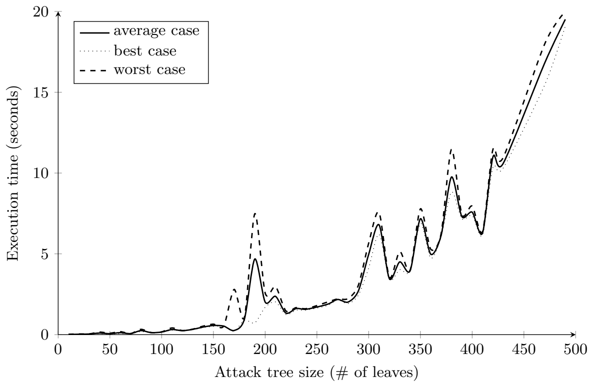


Fig. 10: AGA execution time

26000 leaves in reasonable timeframe, which is approximately 32 times more efficient compared to GA.

The execution time complexity estimations for AGA are outlined in Table 2.

Table 2: AGA execution time complexity estimations

Case	Approximation polynomial	R^2 coefficient
Worst	$3.985x^3 - 0.0001x^2 + 0.0358x - 1.1970$	0.90
Average	$3.5731x^3 - 0.0001x^2 + 0.0267x - 0.8786$	0.94
Best	$3.1892x^3 - 0.0001x^2 + 0.0192x - 0.6115$	0.96

5 Conclusions

This paper addressed the problem of efficient approximation of attack tree evaluation of the failure-free game. We considered the genetic approach to approximation, since it is known to have worked on similar problems previously. However, genetic algorithms depend on various loosely connected parameters (e.g. crossover and mutation operators and their corresponding rates). Selecting them all simultaneously is a non-trivial task requiring a dedicated assessment effort for each particular problem type. The current paper presents the first systematic study of GA parameter optimization for the attack tree evaluation. We have conducted a series of experiments and collected heuristic evidence for optimal parameter selection.

The second contribution of the paper is the application of adaptive genetic algorithms (AGA) to the problem domain of attack tree computations. It turns out that AGA converges generally faster than GA and provides similar level of accuracy, but with the price of potentially larger population sizes. Since usually there are no major technical obstacles to increasing the population, we conclude that AGA should be preferred to plain GA in the considered application domain.

References

1. Vesely, W., Goldberg, F., Roberts, N., Haasl, D.: Fault Tree Handbook. US Government Printing Office (January 1981) Systems and Reliability Research, Office of Nuclear Regulatory Research, U.S. Nuclear Regulatory Commission.
2. Weiss, J.D.: A system security engineering process. In: Proceedings of the 14th National Computer Security Conference. (1991) 572–581
3. Schneier, B.: Attack trees: Modeling security threats. Dr. Dobb's Journal **24**(12) (December 1999) 21–29
4. Mauw, S., Oostdijk, M.: Foundations of attack trees. In Won, D., Kim, S., eds.: International Conference on Information Security and Cryptology – ICISC 2005. Volume 3935 of LNCS., Springer (2005) 186–198
5. Buldas, A., Laud, P., Priisalu, J., Saarepera, M., Willemson, J.: Rational Choice of Security Measures via Multi-Parameter Attack Trees. In: Critical Information Infrastructures Security. First International Workshop, CRITIS 2006. Volume 4347 of LNCS., Springer (2006) 235–248
6. Jürgenson, A., Willemson, J.: Serial Model for Attack Tree Computations. In Lee, D., Hong, S., eds.: ICISC. Volume 5984 of Lecture Notes in Computer Science., Springer (2009) 118–128
7. Jürgenson, A., Willemson, J.: Computing Exact Outcomes of Multi-parameter Attack Trees. In Meersman, R., Tari, Z., eds.: OTM Conferences (2). Volume 5332 of Lecture Notes in Computer Science., Springer (2008) 1036–1051
8. Jürgenson, A., Willemson, J.: On Fast and Approximate Attack Tree Computations. In Kwak, J., Deng, R.H., Won, Y., Wang, G., eds.: ISPEC. Volume 6047 of Lecture Notes in Computer Science., Springer (2010) 56–66
9. Lenin, A., Willemson, J., Sari, D.P.: Attacker profiling in quantitative security assessment based on attack trees. In Bernsmed, K., Fischer-Hübner, S., eds.: Secure IT Systems - 19th Nordic Conference, NordSec 2014, Tromsø, Norway, October 15-17, 2014, Proceedings. Volume 8788 of Lecture Notes in Computer Science., Springer (2014) 199–212
10. Buldas, A., Stepanenko, R.: Upper bounds for adversaries' utility in attack trees. In Grossklags, J., Walrand, J.C., eds.: Decision and Game Theory for Security - Third International Conference, GameSec 2012, Budapest, Hungary, November 5-6, 2012. Proceedings. Volume 7638 of Lecture Notes in Computer Science., Springer (2012) 98–117
11. Buldas, A., Lenin, A.: New efficient utility upper bounds for the fully adaptive model of attack trees. In Das, S.K., Nita-Rotaru, C., Kantarcioglu, M., eds.: Decision and Game Theory for Security - 4th International Conference, GameSec 2013, Fort Worth, TX, USA, November 11-12, 2013. Proceedings. Volume 8252 of Lecture Notes in Computer Science., Springer (2013) 192–205
12. Lenin, A., Buldas, A.: Limiting adversarial budget in quantitative security assessment. In Poovendran, R., Saad, W., eds.: Decision and Game Theory for Security - 5th International Conference, GameSec 2014, Los Angeles, CA, USA, November 6-7, 2014. Proceedings. Volume 8840 of Lecture Notes in Computer Science., Springer (2014) 153–172
13. Srinivas, M., Patnaik, L.M.: Adaptive probabilities of crossover and mutation in genetic algorithms. IEEE Transactions on Systems, Man, and Cybernetics **24**(4) (1994) 656–667

Index

DISSERTATIONS DEFENDED AT TALLINN UNIVERSITY OF TECHNOLOGY ON INFORMATICS AND SYSTEM ENGINEERING

1. Lea Elmik. Informational Modelling of a Communication Office. 1992.
2. Kalle Tammemäe. Control Intensive Digital System Synthesis. 1997.
3. Eerik Lossmann. Complex Signal Classification Algorithms, Based on the Third-Order Statistical Models. 1999.
4. Kaido Kikkas. Using the Internet in Rehabilitation of People with Mobility Impairments – Case Studies and Views from Estonia. 1999.
5. Nazmun Nahar. Global Electronic Commerce Process: Business-to-Business. 1999.
6. Jevgeni Riipulk. Microwave Radiometry for Medical Applications. 2000.
7. Alar Kuusik. Compact Smart Home Systems: Design and Verification of Cost Effective Hardware Solutions. 2001.
8. Jaan Raik. Hierarchical Test Generation for Digital Circuits Represented by Decision Diagrams. 2001.
9. Andri Riid. Transparent Fuzzy Systems: Model and Control. 2002.
10. Marina Brik. Investigation and Development of Test Generation Methods for Control Part of Digital Systems. 2002.
11. Raul Land. Synchronous Approximation and Processing of Sampled Data Signals. 2002.

12. Ants Ronk. An Extended Block-Adaptive Fourier Analyser for Analysis and Reproduction of Periodic Components of Band-Limited Discrete-Time Signals. 2002.
13. Toivo Paavle. System Level Modeling of the Phase Locked Loops: Behavioral Analysis and Parameterization. 2003.
14. Irina Astrova. On Integration of Object-Oriented Applications with Relational Databases. 2003.
15. Kuldar Taveter. A Multi-Perspective Methodology for Agent-Oriented Business Modelling and Simulation. 2004.
16. Taivo Kangilaski. Eesti Energia käiduhaldussüsteem. 2004.
17. Artur Jutman. Selected Issues of Modeling, Verification and Testing of Digital Systems. 2004.
18. Ander Tenno. Simulation and Estimation of Electro-Chemical Processes in Maintenance-Free Batteries with Fixed Electrolyte. 2004.
19. Oleg Korolkov. Formation of Diffusion Welded Al Contacts to Semiconductor Silicon. 2004.
20. Risto Vaarandi. Tools and Techniques for Event Log Analysis. 2005.
21. Marko Koort. Transmitter Power Control in Wireless Communication Systems. 2005.
22. Raul Savimaa. Modelling Emergent Behaviour of Organizations. Time-Aware, UML and Agent Based Approach. 2005.
23. Raido Kurel. Investigation of Electrical Characteristics of SiC Based Complementary JBS Structures. 2005.
24. Rainer Taniloo. Ökonoomsete negatiivse diferentsiaaltakistusega astmete ja elementide disainimine ja optimeerimine. 2005.
25. Pauli Lallo. Adaptive Secure Data Transmission Method for OSI Level I. 2005.
26. Deniss Kumlander. Some Practical Algorithms to Solve the Maximum Clique Problem. 2005.
27. Tarmo Veskioja. Stable Marriage Problem and College Admission. 2005.
28. Elena Fomina. Low Power Finite State Machine Synthesis. 2005.
29. Eero Ivask. Digital Test in WEB-Based Environment 2006.

30. Виктор Войтович. Разработка технологий выращивания из жидкой фазы эпитаксиальных структур арсенида галлия с высоковольтным p-n переходом и изготовления диодов на их основе. 2006.
31. Tanel Alumäe. Methods for Estonian Large Vocabulary Speech Recognition. 2006.
32. Erki Eessaar. Relational and Object-Relational Database Management Systems as Platforms for Managing Softwareengineering Artefacts. 2006.
33. Rauno Gordon. Modelling of Cardiac Dynamics and Intracardiac Bioimpedance. 2007.
34. Madis Listak. A Task-Oriented Design of a Biologically Inspired Underwater Robot. 2007.
35. Elmet Orasson. Hybrid Built-in Self-Test. Methods and Tools for Analysis and Optimization of BIST. 2007.
36. Eduard Petlenkov. Neural Networks Based Identification and Control of Nonlinear Systems: ANARX Model Based Approach. 2007.
37. Toomas Kirt. Concept Formation in Exploratory Data Analysis: Case Studies of Linguistic and Banking Data. 2007.
38. Juhan-Peep Ernits. Two State Space Reduction Techniques for Explicit State Model Checking. 2007.
39. Innar Liiv. Pattern Discovery Using Seriation and Matrix Reordering: A Unified View, Extensions and an Application to Inventory Management. 2008.
40. Andrei Pokatilov. Development of National Standard for Voltage Unit Based on Solid-State References. 2008.
41. Karin Lindroos. Mapping Social Structures by Formal Non-Linear Information Processing Methods: Case Studies of Estonian Islands Environments. 2008.
42. Maksim Jenihhin. Simulation-Based Hardware Verification with High-Level Decision Diagrams. 2008.
43. Ando Saabas. Logics for Low-Level Code and Proof-Preserving Program Transformations. 2008.
44. Ilja Tšahhirov. Security Protocols Analysis in the Computational Model – Dependency Flow Graphs-Based Approach. 2008.
45. Toomas Ruuben. Wideband Digital Beamforming in Sonar

Systems. 2009.

46. Sergei Devadze. Fault Simulation of Digital Systems. 2009.
47. Andrei Krivošei. Model Based Method for Adaptive Decomposition of the Thoracic Bio-Impedance Variations into Cardiac and Respiratory Components. 2009.
48. Vineeth Govind. DfT-Based External Test and Diagnosis of Mesh-like Networks on Chips. 2009.
49. Andres Kull. Model-Based Testing of Reactive Systems. 2009.
50. Ants Torim. Formal Concepts in the Theory of Monotone Systems. 2009.
51. Erika Matsak. Discovering Logical Constructs from Estonian Children Language. 2009.
52. Paul Annus. Multichannel Bioimpedance Spectroscopy: Instrumentation Methods and Design Principles. 2009.
53. Maris Tõnso. Computer Algebra Tools for Modelling, Analysis and Synthesis for Nonlinear Control Systems. 2010.
54. Aivo Jürgenson. Efficient Semantics of Parallel and Serial Models of Attack Trees. 2010.
55. Erkki Joasoon. The Tactile Feedback Device for Multi-Touch User Interfaces. 2010.
56. Jürgo-Sören Preden. Enhancing Situation – Awareness Cognition and Reasoning of Ad-Hoc Network Agents. 2010.
57. Pavel Grigorenko. Higher-Order Attribute Semantics of Flat Languages. 2010.
58. Anna Rannaste. Hierarcical Test Pattern Generation and Untestability Identification Techniques for Synchronous Sequential Circuits. 2010.
59. Sergei Strik. Battery Charging and Full-Featured Battery Charger Integrated Circuit for Portable Applications. 2011.
60. Rain Ottis. A Systematic Approach to Offensive Volunteer Cyber Militia. 2011.
61. Natalja Sleptšuk. Investigation of the Intermediate Layer in the Metal-Silicon Carbide Contact Obtained by Diffusion Welding. 2011.
62. Martin Jaanus. The Interactive Learning Environment for Mobile Laboratories. 2011.
63. Argo Kasemaa. Analog Front End Components for Bio-Impe-

Sensor Systems. 2012.

81. Jaan Ojarand. Wideband Excitation Signals for Fast Impedance Spectroscopy of Biological Objects. 2012.
82. Igor Aleksejev. FPGA-based Embedded Virtual Instrumentation. 2013.
83. Juri Mihhailov. Accurate Flexible Current Measurement Method and its Realization in Power and Battery Management Integrated Circuits for Portable Applications. 2013.
84. Tõnis Saar. The Piezo-Electric Impedance Spectroscopy: Solutions and Applications. 2013.
85. Ermo Täks. An Automated Legal Content Capture and Visualisation Method. 2013.
86. Uljana Reinsalu. Fault Simulation and Code Coverage Analysis of RTL Designs Using High-Level Decision Diagrams. 2013.
87. Anton Tšepurov. Hardware Modeling for Design Verification and Debug. 2013.
88. Ivo Müürsepp. Robust Detectors for Cognitive Radio. 2013.
89. Jaas Ježov. Pressure sensitive lateral line for underwater robot. 2013.
90. Vadim Kaparin. Transformation of Nonlinear State Equations into Observer Form. 2013.
91. Reeno Reeder. Development and Optimisation of Modelling Methods and Algorithms for Terahertz Range Radiation Sources Based on Quantum Well Heterostructures. 2014.
92. Ants Koel. GaAs and SiC Semiconductor Materials Based Power Structures: Static and Dynamic Behavior Analysis. 2014.
93. Jaan Übi. Methods for Coopetition and Retention Analysis: An Application to University Management. 2014.
94. Innokenti Sobolev. Hyperspectral Data Processing and Interpretation in Remote Sensing Based on Laser-Induced Fluorescence Method. 2014.
95. Jana Toompuu. Investigation of the Specific Deep Levels in p-, i- and n- Regions of GaAs p+-pin-n+ Structures. 2014.
96. Taavi Salumäe. Flow-Sensitive Robotic Fish: From Concept to Experiments. 2015.
97. Yar Muhammad. A Parametric Framework for Modelling of