New Directions in Attack Tree Research: Catching up with Industrial Needs

Olga Gadyatskaya[1] and Rolando Trujillo-Rasua[1]

SnT, University of Luxembourg, Luxembourg
name.surname@uni.lu

Abstract. Attack trees provide a systematic way of characterizing diverse system threats. Their strengths arise from the combination of an intuitive representation of possible attacks and availability of formal mathematical frameworks for analyzing them in a qualitative or a quantitative manner. Indeed, the mathematical frameworks have become a large focus of attack tree research. However, practical applications of attack trees in industry largely remain a tedious and error-prone exercise.
Recent research directions in attack trees, such as attack tree generation, attempt to close this gap and to improve the attack tree state-of-the-practice. In this position paper we outline the recurrent challenges in manual tree design within industry, and we overview the recent research results in attack trees that help the practitioners. For the challenges that have not yet been addressed by the community, we propose new promising research directions.

1 Introduction

Attack trees are one of the most popular graphical models for information security assessment. Proposed originally by Bruce Schneier in 1999 [55], they are intuitive and relatively easy to master, yet they enjoy well-studied formalizations and quantitative analysis means [37,29,4,5]. Security risk assessment at industry has long appreciated attack trees as a means to solve cognitive scalability issues related to securing large systems [44], and as a tool to enable communication among different stakeholders and facilitate brainstorming [15,57].

From the graphical perspective an attack tree resembles a mind map [11,57]: a powerful cognitive tool used often in Psychology and Education. It has a single root node, representing the main attacker's goal, which is subsequently **refined** in sub-goals captured by child nodes. But this is where the analogy ends, as attack trees have a precise mathematical interpretation based on well-defined refinement operators [37]. The most frequently used refinement operators are the conjunctive (AND) and disjunctive (OR) refinements. The AND-refinement sets that all child nodes need to be performed to achieve the parent node, while the OR-refinement states that if the attacker can achieve any of the child nodes, then the parent node will also be achieved [37]. The leaf nodes, i.e., the nodes that do not have any children, represent atomic or self-evident attack steps.

Attack trees are traditionally regarded as both a formal framework and a tool to communicate security risks, but these two aspects are rarely considered together in the literature. For example, research in attack trees strongly focuses on improving expressiveness of the formalism through new refinement operators (e.g., sequential AND [24,3]), proposing new flavors of attack trees (e.g., attack-defense trees [29] or attack-countermeasure trees [53]), and developing various quantitative analysis methods with attack trees [7,37,8,26,18]. Quantitative analysis techniques give rise to optimization problems, such as cost-effective countermeasure selection [53,17,4] and ranking of attacks [18]. In parallel, new attack tree semantics are being developed that translate attack trees into other formalisms, such as timed automata [30,16], stochastic games [5] and Markov chains [23], which offer advanced computational capabilities.

Mostly separately from those efforts, researchers have looked into integration of attack trees with security risk management methodologies [47,45,17] and applying attack trees to model threat scenarios in a large variety of domains, e.g., SCADA systems [39,61], RFID systems [7], and ATM systems [15].

More recently, researchers have started to look into the questions emerging when applying attack trees in industry. Indeed, practical applications of attack trees pose many challenges, including noticeable time investment into the tree design, significant cognitive burden on the analyst when dealing with large trees, and multitude of possible interpretations for even the most basic refinement operators AND and OR that may lead to conceptual misalignments within a single tree created and read by a group of people. This is why attack trees are sometimes mentioned in security threat assessment guidelines as an "advanced" and "alternative" model [57,42,38]. In this position paper we overview the emerging research directions for practical applications of attack trees and identify the gaps that are still to be investigated by the research community interested in attack trees. We argue that there are many exciting research problems that can contribute to better acceptance of attack trees in practice and to a better synergy between the academic and industry worlds.

We start by reviewing some practical challenges with attack trees and lessons we learned while working with attack trees (Section 2). Then we overview the emerging research directions in attack trees that focus on improving the acceptance of the formalism in practice (Section 3). Subsequently, we propose additional research themes that are currently not so active, but will be appreciated by practitioners (Section 4).

2 Challenges with attack tree design in practice

Attack trees allow to structure quite diverse threat scenarios (e.g., attacks occurring at physical, digital or social levels, or a combination of those) and to reason about these scenarios at different levels of abstraction. Yet, this strength comes at the cost of **substantial time and effort investment** into the design of a comprehensive tree. Like any other security assessment methodology, a thorough attack tree model requires a diverse set of skills from its authors. Typically, an

attack tree design exercise requires domain expertise (i.e., stakeholders providing in-depth knowledge of the system) and security expertise (persons providing security knowledge and experience in the methodology). For example, the attack-defense tree for the relatively small ATM case study reported in [15] required a team of two domain experts and two security analysts working jointly for 6 days. Furthermore, one of the stakeholders has spent 10 days before the case study meeting on preparing the documentation, reviewing ATM crime reports, outlining the scenario, and collecting statistical data for quantitative analysis. This amount of effort can be prohibitive for small organizations.

A delaying factor in designing attack trees is that, although they may seem to be quite simple and intuitive [55,56,57], there is still a **space for misconceptions and a multitude of interpretations** about the meaning of refinement operators or tree semantics. These discrepancies are often neglected in the literature, but they become apparent when a group of people with diverse backgrounds starts working together on an attack tree. For example, in case of attack-defense trees applied in [15], the meaning of defense nodes was problematic. For the stakeholders, the defense nodes represented anything that is a security control, irrespectively of its type (preventive, detective or reactive). However, the attack-defense tree formalism provides a uniform meaning to defense nodes through semantics and attribute domain specification [29], and it does not allow to specify explicitly different countermeasure types.

Attribute domains can introduce confusion even for pure attack trees without countermeasure nodes. For example, the traditional AND operator specifies that all children nodes need to be fulfilled to fulfil the parent node. Its interpretation in the minimal attack time attribute domain depends whether the children node actions can be done in parallel or they are sequential. The ADTool software, for example, includes both options, and so the practitioners have to choose which one to work with [28]. This means, they need to be aware of this choice, and they need to interpret it correctly and consistently for the whole tree.

In fact, attack tree guidelines existing today in the literature are quite vague, and they usually operate within the top-down approach. The team needs to start with the top node representing the attacker's main goal, that can be refined into subgoals and more concrete steps until very precise attack actions are found [1,57,60,37,55]. The guidelines do not specify what is the best way to structure the tree, how to deal with repeating nodes, how to label the nodes in the best way, or how to arrange the work on the attack tree so that everybody has the same understanding of the attack tree elements meaning. This means that, in the best case, these choices are strategically made by the most experienced team members, who, however, do not share them with the wider community, or they are made ad-hoc or even post-hoc. In the worst case, these aspects are not agreed upon at all, and, therefore, the resulting tree can be inconsistent. Furthermore, this tree will likely be less comprehensible, due to the **absence of empirically founded best practices** in tree structure and comprehensible tree design.

Absence of **errors** is another big concern for practitioners when designing attack trees [57,55]. These errors can be on both sides, and, therefore, optimally,

the designed tree should be both complete (no attacks are omitted) and sound (does not contain attacks that do not exist in the actual system). Practitioners can apply some tree validation techniques to ensure this. For example, in the ATM case study semantics-based validation (checking that attack bundles in the multiset semantics [29] represented meaningful attacks), data-based validation (investigating any discrepancies between the expected attribute value and the value computed in the quantitative analysis), and catalogue-based validation (ensuring that all attacks collected by an industry specific catalogue are captured) were applied [15]. However, these validation techniques are limited when applied by human analysts, because it is impractical to check by hand all possible attack bundles or data value discrepancies.

Certainly the attack tree construction process is an excellent opportunity for brainstorming about potential security threats and cost-effective countermeasures. But its main value comes from post-analysis and subsequent communication with other stakeholders. We observe that in practice **analysis and comprehensibility of attack trees are in conflict**. On the one hand, fine-grained analysis benefits from large trees describing all attacks on a concrete system. But on the other hand, large models can strain analysts' cognitive capabilities, and the practitioners may find it difficult to comprehend all described attack scenarios, especially after a certain time.

Acquisition of input data is a challenge by itself in risk assessment methodologies [65], and attack trees magnify it due to a large number of leaf nodes [43,7]. The standard approach for attack trees, when only leaf nodes are annotated with values can be too restrictive, as often data for intermediate nodes can be more readily available than data for leaves. This observation is further reinforced if we consider the costs of data collection in an organization (in terms of effort, time, etc.). Sometimes, more generic data than expected is available, e.g. from historical databases, multiple surveys and empirical results [15]. In this case, correlation and normalization of data to fit the attack tree methodology can become a challenge.

We notice that there exists a **conceptual mismatch** between research on attack trees and practical applications of attack trees. As we mentioned, in practice, attack trees are constructed by following the top-down approach. Yet, academic papers on attack trees define semantics and quantitative analysis techniques for these models via the leaf nodes, i.e., the lowest-level events (bottom-up approach). This makes it hard to implement a consistent feedback loop between design and analysis.

3 Research trends in attack tree applications

Given the challenges summarized in Section 2, several promising research directions have emerged recently to address the needs of practitioners.

Attack tree generation. Manual design is the state-of-practice for attack trees [54,57]. However, this exercise is time-consuming and error-prone. Automated tree generation techniques have emerged very recently, and there are few

approaches reported in the literature yet [64,22,48]. These works provide means to generate attack trees from some system model, under assumption that it is easier for the team to design a good system model than a good attack tree. However, there is still some way to go before generated attack trees can be used in the security risk assessment practice. First of all, the techniques reported in [64] and [22] generate refinement-unaware trees, i.e., trees that do not support the user in understanding the various levels of abstraction. In tradition with the propositional semantics of attack trees [37], but in contrast with the expectations of a security analyst, [64] interprets each intermediate node as a combination of child nodes, without offering any higher-level meaning. [22] offers trees with meaningful intermediate nodes, yet they still lack a proper refinement structure, when more abstract subgoals are refined into more precise attack steps. ATSyRA is today the only approach that is capable to generate refinement-aware trees [49,48]. It extracts the refinement structure from a hierarchy of actions in the model defined by the expert. Still, all these approaches [64,22,48] output huge attack trees that can be incomprehensible to the analyst team.

Other attack tree generation approaches work with established security catalogues and knowledge bases, and attempt to construct attack trees from them. Knowledge bases and catalogues that systematize information on attacks, vulnerabilities and countermeasures are a trusted source of information for security risk practitioners, and many security risk management techniques include one or more catalogues [2,9,40]. Suggestions to apply established catalogues of threats to facilitate manual tree design have been voiced in [15,57]. Furthermore, for some knowledge bases, it is straightforward to transform certain attack scenarios described in those into attack trees. Then an analyst can manually produce more complex threat scenarios from these attack trees [19,62]. Techniques to automate attack tree generation using security catalogues and libraries have been reported in [44,50]. The TREsPASS project [51,63] has applied a security knowledge base to attack trees generated from a system model in order to refine leaf nodes of particular types, mainly for precise attacks on human agents and processes, such as social engineering or hacking.

Attack tree generation allows to reason on formal properties of obtained models. Indeed, for manually designed trees, it is understood that these models are as complete as the knowledge and experience of experts who designed them [57,55]. When attack trees are produced from an underlying system model or a knowledge base, it is possible to define the notion of **completeness** with respect to the model, and one can investigate whether an approach generates complete trees. For example, completeness with respect to a knowledge base is established as a desired property in [44], and completeness of a generated tree with respect to a system model is established in [22].

Another interesting property of generated attack trees is **soundness** with respect to a system model, i.e., whether all attack scenarios captured by a tree are valid attacks in the model. Audinot and Pinchinat defined the soundness property for generated attack trees [6]. Soundness is very important when one wants to generate refinement-aware trees. Indeed, refinement establishes how

abstract actions can be represented as combinations of more precise ones. Yet, not all combinations of precise actions can result in a valid attack in the system model.

Attack tree visualization. The TREsPASS project has proposed means for visualizing large attack trees [35,20,46]. This visualisation portfolio strives to hide away complexity of the tree by removing the node labels, arranging the tree circularly, linearizing complex attack scenarios, and supporting zooming-in and out (at the visualization level). These methods lead to reduction of the cognitive effort needed to process a complex tree, yet they contrast with the traditional manually designed attack trees, where meaningful node labels are essential, trees are arranged vertically to allow label readability, and non-linearism of attack scenarios gives an opportunity to reason about complex attacks [57,56].

Empirical studies with attack trees. To the best of our knowledge, Opdahl and Sindre [41] and Karpati et al. [25] have been the only ones reporting on empirical studies with attack trees. These studies compared attack trees with misuse cases in the context of threat assessment, and they have reported that attack trees allowed the participants to find more threats.

4 Next Steps and Conclusions

Comparing the challenges enumerated in Sec. 2 and research results summarized in Sec. 3, we can see that some challenges are addressed by an ongoing or past research effort. Indeed, *generation techniques* strive to reduce the *time and effort required to produce attack trees*, and to provide a framework for guaranteeing *absence of errors* in the obtained model. *Visualization approaches* are helping the analysts to better *comprehend* attack trees and to improve the *cognitive scalability* of the method. Yet, these results can still be strengthened and extended towards more user-friendly models.

In particular, the generation techniques can be improved by working on the refinement-awareness for the produced models. To achieve this, we propose to establish a new refinement-aware semantics for attack trees that will allow to assign meaning to intermediate nodes independently of their children nodes. The generated trees will need to be correct with respect to this semantics. Refinement relationship can be either defined by an expert as in [48], or it can be extracted from an appropriate knowledge base. Furthermore, the generated trees can be transformed into semantically-equivalent forms that have less nodes [27,37], what could potentially improve the comprehensibility of these smaller trees.

Comprehensibility and readability of graphs and the limits of human cognitive capabilities while reading and analizying graphical data have been explored in, e.g., [13,52]. Information visualization challenges related to usability and scalability were highlighted in [12]. It will be interesting to see the findings of these works applied in the attack tree domain.

In the security risk assessment area, comprehensibility studies of visual and textual security risk models were reported in, e.g., [33,36,21]. A classification of scenarios for empirical studies in information visualization was proposed by Lam

et al. [34], and visualization evaluation for cyber security was discussed by Staheli et al. [59]. To the best of our knowledge, there have been yet no empirical studies of attack tree comprehensibility, and this could be a promising research direction. Indeed, outside the attack trees topic, there is a rich empirical research literature on security modeling and assessment [31,32,10], software engineering [66], and requirements engineering [58]. This literature can be used by the attack trees community to build upon.

The challenges *acquisition of input data*, *absence of empirically grounded best practices*, *the trade-off between analysis and comprehensibility*, and the *conceptual mismatch between the top-down manual tree design process and the bottom-up formal semantics* have not yet been addressed in the attack trees community.

The data issues for quantitative analysis is a complex problem, because the quality and quantity of available data strongly depend on the application domain. In the quantitative risk analysis domain data-related challenges are known, and there exist methodologies for validating the data [65]. The attack tree community may thus strive to devise new methodologies for data validation and data-based tree validation.

We observe that the tension between detailed analysis, which requires large-scale trees, and comprehensibility, which tends to drop with the size of the tree, can be mitigated by means of model transformation techniques. Model transformation is fundamental in Computer Science and key in Model-driven software development [14], as it provides models at different levels of abstraction in a synchronized way. In that regards, attack tree generation can be seen as a model transformation approach; from a system model to an attack tree model. It would be interesting to see other types of transformations, e.g., from an attack tree to an attack tree, that could yield to more condensed yet human-readable trees.

We argue that the misconceptions and multitude of interpretations of attack trees can be addressed by establishing a more rigorous methodology for practical application of attack trees that will include an initial phase when interpretations of the tree semantics and refinement operators are agreed upon. This methodology needs to be grounded in empirical studies with attack tree practitioners, in which they could report on what are the most frequent communication pitfalls they face, and how do they interpret different attack tree-related aspects, such as operators, semantics, etc.

Overall, we can conclude that the attack tree research community has made a substantial progress in developing the formal framework underpinning the model. We as researchers have a huge choice of attack tree semantics, quantitative analysis techniques, software tools, and means to apply attack trees in security assessment case studies. It is also exciting to see that the research community has started to focus on the practical needs of security analysts working with manually designed attack trees in organizational threat modeling and security risk management. We believe that this synergy between research and industry can further enhance the attack tree formalism and it will open new horizons in the attack tree research.

References

1. Amenaza: Creating Secure Systems through Attack Tree Modeling. http://www.amenaza.com/ (2003)
2. ANSSI: EBIOS – Expression des Besoins et Identification des Objectifs de Securite (2010)
3. Arnold, F., Guck, D., Kumar, R., Stoelinga, M.: Sequential and parallel attack tree modelling. In: Proc. of SAFECOMP and Workshops. pp. 291–299 (2015)
4. Aslanyan, Z., Nielson, F.: Pareto efficient solutions of attack-defence trees. In: Proc. of POST. Springer (2015)
5. Aslanyan, Z., Nielson, F., Parker, D.: Quantitative verification and synthesis of attack-defence scenarios. In: Proc. of CSF. IEEE (2016)
6. Audinot, M., Pinchinat, S.: On the soundness of attack trees. In: Proc. of GraMSec. pp. 25–38. Springer (2016)
7. Bagnato, A., Kordy, B., Meland, P.H., Schweitzer, P.: Attribute decoration of attack–defense trees. International Journal of Secure Software Engineering (IJSSE) 3(2) (2012)
8. 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, pp. 235–248. Springer (2006)
9. Bundesamt fur Sicherheit in der Informationstechnik: IT-Grundschutz-Catalogues, 13th version (2013)
10. Buyens, K., De Win, B., Joosen, W.: Empirical and statistical analysis of risk analysis-driven techniques for threat management. In: Proc. of ARES. IEEE (2007)
11. Buzan, T., Buzan, B.: The Mind Map Book: How to Use Radiant Thinking to Maximize Your Brain's Untapped Potential. Plume, reprint edn. (Mar 1996), http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0452273226
12. Chen, C.: Top 10 unsolved information visualization problems. IEEE computer graphics and applications 25(4) (2005)
13. Cleveland, W.: The elements of graphing data. AT&T Bell Laboratories (1994)
14. Czarnecki, K., Helsen, S.: Feature-based survey of model transformation approaches. IBM Systems Journal 45(3), 621–645 (2006)
15. Fraile, M., Ford, M., Gadyatskaya, O., Kumar, R., Stoellinga, M., Trujillo-Rasua, R.: Using attack-defense trees to analyze threats and countermeasures in an ATM: A case study. In: Proc. of PoEM. Springer (2016)
16. Gadyatskaya, O., Hansen, R.R., Larsen, K.G., Legay, A., Olesen, M.C., Poulsen, D.B.: Modelling attack-defense trees using timed automata. In: Proc. of FORMATS. LNCS, vol. 9884. Springer (2016)
17. Gadyatskaya, O., Harpes, C., Mauw, S., Muller, C., Muller, S.: Bridging two worlds: Reconciling practical risk assessment methodologies with theory of attack trees. In: Proc. of GraMSec. LNCS, vol. 9987. Springer (2016)
18. Gadyatskaya, O., Jhawar, R., Kordy, P., Lounis, K., Mauw, S., Trujillo-Rasua, R.: Attack trees for practical security assessment: Ranking of attack scenarios with ADTool 2.0. In: Proc. of QEST. LNCS, vol. 9826. Springer (2016)
19. Ghani, H., Luna, J., Petkov, I., Suri, N.: User-centric security assessment of software configurations: A case study. In: Proc. of ESSoS. pp. 196–212. Springer (2014)
20. Hall, P., Heath, C., Coles-Kemp, L., Tanner, A.: Examining the contribution of critical visualisation to information security. In: Proc. of NSPW. ACM (2015)

21. Hogganvik Grøndahl, I., Lund, M.S., Stølen, K.: Reducing the effort to comprehend risk models: Text labels are often preferred over graphical means. Risk Analysis 31(11) (2011)
22. Ivanova, M.G., Probst, C.W., Hansen, R.R., Kammuller, F.: Transforming graphical system models to graphical attack models. In: Proc. of GraMSec. LNCS, vol. 9390. Springer (2015)
23. Jhawar, R., Lounis, K., Mauw, S.: A stochastic framework for quantitative analysis of attack-defense trees. In: Proc. of STM. Springer (2016)
24. Jhawar, R., Kordy, B., Mauw, S., Radomirović, S., Trujillo-Rasua, R.: Attack trees with sequential conjunction. In: Proc. of IFIP SEC. Springer (2015)
25. Karpati, P., Redda, Y., Opdahl, A., Sindre, G.: Comparing attack trees and misuse cases in an industrial setting. Inform. Software Tech. 56(3), 294–308 (2014)
26. Kordy, B., Mauw, S., Schweitzer, P.: Quantitative questions on attack-defense trees. In: Proc. of ICISC. LNCS, vol. 7839. Springer (2013)
27. Kordy, B., Kordy, P., van den Boom, Y.: Sptool–equivalence checker for\mathtt {SAND} attack trees. In: Proc. of CRISiS. pp. 105–113. Springer (2016)
28. Kordy, B., Kordy, P., Mauw, S., Schweitzer, P.: ADTool: Security analysis with attack-defense trees. In: Proc. of QEST (2013)
29. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Attack–Defense Trees. Journal of Logic and Computation 24(1), 55–87 (2014), http://people.rennes.inria.fr/Barbara.Kordy/papers/ADT12.pdf
30. Kumar, R., Ruijters, E., Stoelinga, M.: Quantitative attack tree analysis via priced timed automata. In: Proc. of FORMATS. Springer (2015)
31. Labunets, K., Massacci, F., Paci, F.: On the equivalence between graphical and tabular representations for security risk assessment. In: Proc. of REFSQ. Springer (2017)
32. Labunets, K., Massacci, F., Paci, F., L., T.: An experimental comparison of two risk-based security methods. In: Proc. of ESEM. pp. 163–172. IEEE (2013)
33. Labunets, K., Massacci, F., Paci, F., Marczak, S., de Oliveira, F.: Model comprehension for security risk assessment: an empirical comparison of tabular vs. graphical representations. Empirical Software Engineering (2017)
34. Lam, H., Bertini, E., Isenberg, P., Plaisant, C., Carpendale, S.: Empirical studies in information visualization: Seven scenarios. IEEE trans. on visualization and computer graphics 18(9) (2012)
35. Li, E., Barendse, J., Brodbeck, F., Tanner, A.: From a to z: developing a visual vocabulary for information security threat visualisation. In: Proc. of GraMSec. Springer (2016)
36. Matulevičius, R.: Model comprehension and stakeholder appropriateness of security risk-oriented modelling languages. In: Proc. of BPMDS. Springer (2014)
37. Mauw, S., Oostdijk, M.: Foundations of Attack Trees. In: ICISC'05. LNCS, vol. 3935, pp. 186–198. Springer (2006)
38. Microsoft: Threat modeling https://msdn.microsoft.com/en-us/library/ff648644.aspx (2003)
39. Nielsen, J.: Evaluating information assurance control effectiveness on an air force supervisory control and data acquisition (SCADA) system. Tech. rep., DTIC Document (2011)
40. NIST: Special Publication 800-53 Revision 4. Security and privacy controls for federal information systems and organizations http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf (2013)
41. Opdahl, A.L., Sindre, G.: Experimental comparison of attack trees and misuse cases for security threat identification. Inform. Software Tech. 51(5), 916–932 (2009)

42. OWASP: CISO AppSec Guide: Criteria for managing application security risks (2013)
43. P. Schweitzer: Attack–Defense Trees. Ph.D. thesis, University of Luxembourg (2013)
44. Paul, S.: Towards automating the construction & maintenance of attack trees: a feasibility study. In: Proc. of GraMSec (2014)
45. Paul, S., Vignon-Davillier, R.: Unifying traditional risk assessment approaches with attack trees. Journal of Information Security and Applications 19(3), 165–181 (2014)
46. Pieters, W., Barendse, J., Ford, M., Heath, C., Probst, C.W., Verbij, R.: The navigation metaphor in security economics. IEEE Security & Privacy 14(3), 14–21 (2016)
47. Pieters, W., Davarynejad, M.: Calculating adversarial risk from attack trees: Control strength and probabilistic attackers. In: Proc. of DPM/SETOP/QASA, pp. 201–215. Springer (2015)
48. Pinchinat, S., Acher, M., Vojtisek, D.: Towards synthesis of attack trees for supporting computer-aided risk analysis. In: Proc. of SEFM and Workshops. LNCS, vol. 8938, pp. 363–375 (2014)
49. Pinchinat, S., Acher, M., Vojtisek, D.: Atsyra: an integrated environment for synthesizing attack trees. In: Proc. of GraMSec. pp. 97–101. Springer (2015)
50. Popp Fredslund, M.: Automated synthesis of attack-defense trees using a library of component attacks. Master thesis, University of Luxembourg (2015)
51. Probst, C.W., Willemson, J., Pieters, W.: The attack navigator. In: Proc. of GraMSec. pp. 1–17 (2015)
52. Purchase, H.C., Cohen, R.F., James, M.I.: An experimental study of the basis for graph drawing algorithms. Journal of Experimental Algorithmics (JEA) 2, 4 (1997)
53. Roy, A., Kim, D.S., Trivedi, K.: Scalable optimal countermeasure selection using implicit enumeration on attack countermeasure trees. In: Proc. of DSN. IEEE (2012)
54. Saini, V., Duan, Q., Paruchuri, V.: Threat modeling using attack trees. J. of Computing Sciences in Colleges 23(4), 124–131 (2008)
55. Schneier, B.: Attack Trees. Dr. Dobb's Journal of Software Tools 24(12), 21–29 (1999), http://www.ddj.com/security/184414879
56. Schneier, B.: Secrets and lies: digital security in a networked world. John Wiley & Sons (2011)
57. Shostack, A.: Threat modeling: Designing for security. John Wiley & Sons (2014)
58. Sommerville, I., Ransom, J.: An empirical study of industrial requirements engineering process assessment and improvement. ACM Trans. on Software Engineering and Methodology 14(1) (2005)
59. Staheli, D., Yu, T., Crouser, R.J., Damodaran, S., Nam, K., O'Gwynn, D., McKenna, S., Harrison, L.: Visualization evaluation for cyber security: Trends and future directions. In: Proc. of VizSec. ACM (2014)
60. Synopsis: How mapping the Ocean's Eleven heist can make you better at application security testing https://www.synopsys.com/blogs/software-security/oceans-eleven-make-you-better-at-application-security-testing/ (2015)
61. Ten, C.W., Liu, C.C., Govindarasu, M.: Vulnerability assessment of cybersecurity for scada systems using attack trees. In: Power Engineering Society General Meeting, 2007. IEEE. IEEE (2007)
62. Tøndel, I.A., Jensen, J., Røstad, L.: Combining misuse cases with attack trees and security activity models. In: Proc. of ARES. pp. 438–445. IEEE (2010)

63. TREsPASS: Technology-supported Risk Estimation by Predictive Assessment of Socio-technical Security, FP7 project, grant agreement 318003 (2012–2016), http://www.trespass-project.eu/
64. Vigo, R., Nielson, F., Nielson., H.R.: Automated generation of attack trees. In: Proc. of CSF. IEEE (2014)
65. Vose, D.: Risk analysis: a quantitative guide. John Wiley & Sons (2008)
66. Wohlin, C., Runeson, P., Höst, M., Ohlsson, M., Regnell, B., Wesslén, A.: Experimentation in software engineering. Springer (2012)