A Survey of Fault and Attack Tree Modeling and Analysis for Cyber Risk Management

Vidhyashree Nagaraju[1], Lance Fiondella[1], and Thierry Wandji[2]
[1]Electrical and Computer Engineering, University of Massachusetts Dartmouth, MA, USA
[2]Naval Air Systems Command, Patuxent River, MD, USA
Email: {vnagaraju and lfiondella}@umassd.edu, ketchiozo.wandji@navy.mil
Phone: 1(508)999-8596, Fax: 1(508)999-8489

Abstract—**Cyber security is of great concern to the Department of Homeland Security (DHS) and other organizations within government, as cyberspace is the gateway to services and infrastructure, making them vulnerable to a wide range of software-based attacks that could result in physical and cyber threats and hazards. It is extremely difficult to secure these cyber-physical systems (CPS) due to the complexity of their interfaces, which often leaves them exposed to elevated levels of risk to severe disruptions, including information security violations that could threaten national and economic security. Therefore, many researchers have dedicated substantial effort to model and analyze cyber-physical systems through red teaming in order to identify various potential strategies an attacker may take to hack into the system so that they can develop effective countermeasures. Reliability and risk modeling approaches discussed in the literature include fault trees (FT), event trees (ET), binary decision diagrams (BDD), Petri nets (PN), Markov modeling (MM), and attack trees (AT) to systematically characterize the risks latent in cyber-physical systems. This paper provides a survey of the two most popular modeling approaches including fault and attack trees, discussing their benefits and potential limitations. This survey should be beneficial to security professionals who wish to apply techniques from reliability and risk modeling to ensure the cyber security of their systems as well as researchers seeking to identify new modeling opportunities.**

I. Introduction

The widespread growth of software services continue to increase the speed and convenience with which individuals go about their daily lives. However, these services often contain lucrative data sets that if stolen could be used for personal financial gain. Moreover, software services reside in all 16 of the Critical Infrastructure Sectors identified by the Department of Homeland Security. The constant stream of data breaches and malicious cyber attacks have made it clear that their intensity and sophistication will only continue to grow in the future. To overcome widespread vulnerabilities in software systems, researchers have made a concerted effort to apply concepts from risk and reliability modeling to scientifically characterize threats to security so that they can identify weaknesses and prioritize efforts to mitigate vulnerabilities. No single modeling methodology is a complete solution. Typically, multiple complementary approaches can provide a more comprehensive risk assessment. Thus, a systematic survey of methods that have been employed to conduct cyber risk assessment is necessary to inform individuals and organizations of the techniques available as well as their potential benefits and limitations.

Many researchers have proposed several approaches to model the security of cyber-physical systems in order to identify and quantify attacker's attempts to gain access to these systems. Most of the existing studies are tree-based analysis methods, which allow the cyber security expert to enumerate possible loopholes or vulnerabilities that an attacker could exploit to obtain unauthorized access to the system. A common paradigm to conduct cyber risk assessment is to form two adversarial teams consisting of a "red team" whose job is to think like an attacker and a "blue team" that seeks to defend the system by developing countermeasures. In many situations, red team information is used to model the systems with techniques including fault trees [1], [2], event trees [3], [4], decision diagrams such as binary decision diagrams [5] and multi-state decision diagrams [6], Petri nets [7], Markov models [8] and hidden Markov models [9], attack trees [10], and attack-defense trees [11]. Each of these modeling approaches offers a unique perspective of a cyber-physical system. Of these modeling approaches, fault and attack trees are most commonly used to characterize cyber-physical systems. These methods also possess complementary solution methods such as BDD [12], [13], which have been studied intensively to optimize algorithmic efficiency to increase the size and complexity of the problems to which tree-based methods can be applied.

This paper surveys applications of the two predominant techniques to conduct risk assessment on cyber-physical systems, namely fault and attack trees. This survey should be beneficial to cyber security professionals wishing to apply these techniques to their systems as well as researchers wishing to understand the potential applications of risk and reliability modeling to cyber security

The remainder of the paper is organized as follows: Section II presents a survey of fault tree and attack tree modeling techniques. Section III presents policy recommendations and Section IV offers conclusions and future research.

II. Modeling Technique

This section presents a brief survey of the various techniques to model and assess cyber-physical systems, including fault trees [1], [2] and attack trees [10], [14].

978-1-5090-6356-7/17/$31.00 ©2017 IEEE

Authorized licensed use limited to: CNR Area Ricerca Genova. Downloaded on September 09,2025 at 16:49:37 UTC from IEEE Xplore. Restrictions apply.

A. Fault Tree

A fault or logic tree [1] is an acyclic graph, which identifies all branches of events that could contribute to an accident or failure and the frequency with which this undesirable event may occur. A fault tree is a top down logic diagram that is drawn using boolean gates [15] to represent how combinations of component failures can lead to system failures. Fault trees are therefore an appropriate analysis technique when the undesirable event of interest is explicitly given and the underlying potential causes of the event have been enumerated in sufficient detail. Specification of a fault tree thus enables the calculation of various system reliability assessments such as the probability of the existence of the top event and frequency of top event occurrence. Event importance assessment and sensitivity analysis can then be performed to identify where reduction in the probability of the underlying events would most significantly reduce the probability of the top level event and hence be potential candidates for improvement through redundancy allocation or reliability optimization.

Figure 1 shows an example of a fault tree, where the nodes represent different events.

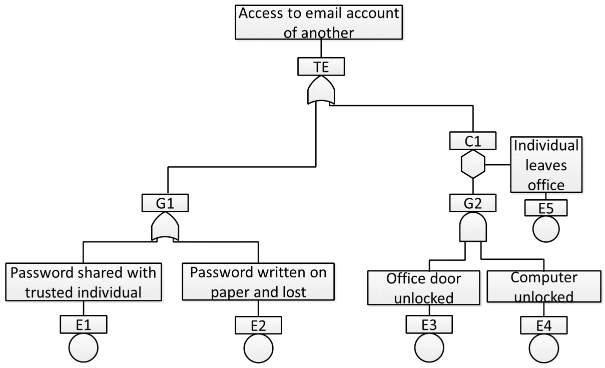
Access to email account of another
TE
C1
Individual leaves office
E5
G1
G2
Password shared with trusted individual
Password written on paper and lost
Office door unlocked
Computer unlocked
E1
E2
E3
E4

Fig. 1. Example fault tree

The basic building blocks [16] of the fault tree shown in Figure 1 include:

- **Top event**: denotes an undesirable event such as a system failure or a vulnerability exploit. Effective fault tree analysis requires careful selection of the top event. If it is too general, the analysis becomes unmanageable, whereas if it is too specific, the analysis does not provide a sufficiently broad view of the system [2]. The top event in Figure 1 access to the email account of another individual and is denoted TE.
- **Basic events**: are the lowest level of detail considered in a fault tree. These can be events such as component failures or events like leaving one's computer unlocked. Examples of basic events in Figure 1 include $E1 - E4$.
- **Gate events**: are logic operators that combine one or more inputs from other nodes such as basic events and other gates in the tree to produce a single binary output. Examples types of gates include AND, OR, and exclusive OR (XOR). Examples of gate events in Figure 1 include $G1$ and $G2$. $G1$ corresponds to the case where an individual shares their password with some one they trust or the password is written on a piece of paper and lost, while $G2$ represent the situation where an individual leaves both their office door and computer unlocked.
- **Condition events**: define a condition that is required in order for a gate event to occur. Example condition events include inhibit, priority AND (PAND), functionality dependency gate (FDEP). Examples of condition events in Figure 1 includes $E5$ which is connected to a condition gate $C1$, which inhibits propagation its output up the tree. In this case, the individual must leave the room before the combination of an unlocked office door and computer can be exploited.
- **Transfer events**: are used to indicate where a node output is used as input to one or more other nodes in the tree, which eliminates the need to specify the subtee at different locations within the tree.

Specifying the triggering events (fault conditions) in terms of events such as those described above combine to form a complete tree.

Fault trees were initially developed in the early 1960s by Watson at Bell Laboratories to facilitate reliability analysis of the Minuteman missile system [1]. Later, in 1975, Barlow and Proschan [17] discussed the importance of system components in terms of the Birnbaum measure, structural importance, minimum cut sets, duality, and Monte Carlo simulation to assess complex systems more easily. Lee *et al.* [18] surveyed subsequent fault tree results reported by researchers including Vesely, Haasl, Barlow and Chatterjee, and Fussell. Earlier research employed only static fault trees consisting of basic logic gates representing boolean functions according to Shannon's method [2] to perform analysis with methods such as minimum cut sets [2], [19], minimum path sets [2], binary decision diagrams [20], and linear time modularization methods [21]. As described by Shannon [22], a Boolean function can take only one of two values: 1 (occurs) and 0 (does not occur). Therefore, given a function with n Boolean variables $X_1, \ldots, X_n$, the function may be expanded about one of its arguments such as X_1 in the following manner [2]:

$$f(X_1, \ldots, X_n) = [X_1 \bullet f(1, \ldots, X_n)] + [X_1 \bullet f(0, \ldots, X_n)]$$

where $\bullet$ and $+$ represent the intersection and union operations.

Real-world system can also depend on the order of component failure, sequence dependent events, spares, redundancy management, and the relative priority of failure events, which cannot be characterized easily by a static fault tree. Therefore, to capture such dynamic behaviors, the dynamic fault tree (DFT) method was proposed [23], which introduces additional elements [24] such as voting (VOT), priority AND (PAND), priority OR (POR), SPARE, sequential logic gate (SEQ), functionality dependency gate (FDEP), and priority dependency gate (PDEP), each of which represent different mechanisms. For example, an FDEP gate expresses situations where one or more preconditions regarding the state of the system must be satisfied in order for the event to occur, whereas a sequential

978-1-5090-6356-7/17/$31.00 ©2017 IEEE
Authorized licensed use limited to: CNR Area Ricerca Genova. Downloaded on September 09,2025 at 16:49:37 UTC from IEEE Xplore. Restrictions apply.

logic gate inhibits one event from being triggered until a predecessor event occurs.

In many cases, only a small portion of a fault tree is dynamic in nature. Therefore, the DIFtree [25] tool implements a modular approach to identify static and dynamic subtrees within the structure, enabling solution of static subtrees with binary decision diagrams (BDD) [13] and dynamic subtrees with Markov chains [26]. The Galileo tool [27] incorporated DIFtree functionality to facilitate cost effective dynamic fault tree analysis with analytical techniques such as cut sets, BDD [13], Markov chains, and Monte Carlo simulation within a graphical interface. Amari *et al.* [28] subsequently incorporated an approach based on conditional probability into commercial tools for DFT analysis.

The complexity of BDD and Markov chain models underlying the solution of DFT limits its scalability, impeding the analysis of complex systems. To simplify, Junges *et al.* [29] interpreted DFT as directed graphs. Other researchers [15], [30] combined static and dynamic features of a system using a method they call static dynamic fault trees (SDFT) in order to improve scalability. Recent developments include a tool to interactively visualize fault tree analysis [31].

Fault trees can be used to model cyber attacks and are often referred to as attack trees when applied to this domain [32]. Hong *et al.* [33] applied dynamic fault trees to model an autonomic computing framework for cyber-physical systems. Patil *et al.* [34] modeled security events arising from employee error, including accidental loss of data when outbound email is sent or other forms of accidental disclosure of sensitive information by insiders. Hashimoto *et al.* [35] proposed a scheme to evaluate the effects of manipulation and concealment by cyber terrorists in the context of industrial control systems. Kornecki and Liu [36] performed fault tree analysis of an Avionics Simulation Network (ASN) to protect against a cyber attack that could affect the security of an aircraft's communications system and the related safety implications. Kuhn *et al.* [37] applied fault tree analysis with extended semantics to efficiently improve the safety of a cyber-medical system for patient treatment despite dependencies among multiple safety measures. Sabaliauskaite and Mathur [38] proposed the Failure-Attack-Countermeasure (FACT) graph modeling approach to simultaneously consider fault trees for safety and attack trees for security, which are two closely related requirements of cyber-physical systems.

The primary benefits of fault trees are that they provide a straightforward visualization of a system to explicitly express the logical relationships between events and causes that can lead to system failure, facilitating the systematic identification potential improvements to improve safety and reliability or lower vulnerability and risk. General disadvantages or difficulties that may be encountered when applying fault tree methods include the need to enumerate all possible sequences of failure dependencies, which is one of several limitations that impedes expressive modeling. Moreover, algorithmic techniques to solve fault trees require time complexity that grows exponentially with the size of the system [26].

B. Attack Trees

Attack trees are tree-structured conceptual diagrams that provide a formal way to describe systems security as a function of all possible conceivable attacks, where the root of the tree denotes a exploit and the leaves different actions to achieve that goal. An attack graphs is *exhaustive*, if it covers all possible attack vectors, and *succinct* if it contains only those network states from which the intruder can succeed in exploiting the system [14]. The primary distinction [32] between attack and fault trees is that a fault tree are used to assess the reliability and safety of components or a system, whereas an attack tree enumerates the possible attack vectors to exploit a system.

Figure 2 shows an example of an attack tree [39].

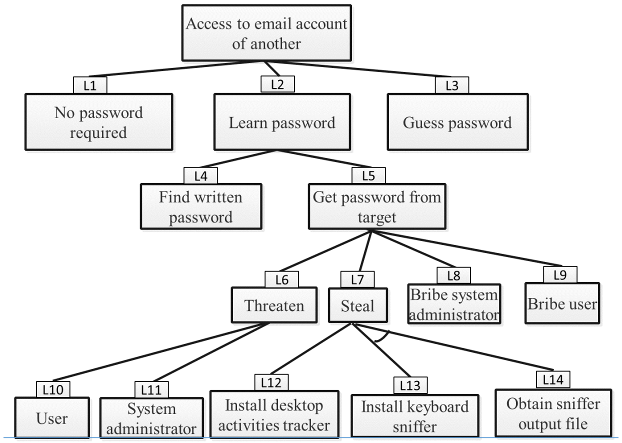

Fig. 2. Example attack tree

The basic building blocks of the attack tree shown in Figure 2 include:

- **Root node**: represents the goal of the attacker. The root node of Figure 2 is to gain access to the email account of an individual.
- **Leaf nodes**: represent basic actions which an attacker can perform to attack the system. Combinations of these leaf nodes identify all possible strategies that the attacker might consider, where low cost and complexity actions may be preferred. In many cases, these leaf nodes are determined from Red team activities. Examples of leaf nodes in Figure 2 include $L1 - L14$.
- **Intermediate nodes**: are non-leaf nodes that denote intermediate goals achieved by performing one or more of the actions given in the children of that node, which are connected to their parent through AND and OR gates. An AND gate indicates that the attacker must perform all of the sub-actions in the child nodes beneath the AND gate, while only one of the sub-actions of an OR gate need to be performed to reach the parent node. An example of an interior node in Figure 2, is 'Steal' where the arc between 'Install keyboard sniffer' and 'obtain sniffer output file' indicates an AND gate, whereas 'Install desktop activities tracker' is an OR gate possessing no arc. A successful

Authorized licensed use limited to: CNR Area Ricerca Genova. Downloaded on September 09,2025 at 16:49:37 UTC from IEEE Xplore. Restrictions apply.

attack is one that reaches the root node from the actions at the leaf nodes.

In general, attacks on computer systems are coordinated but very difficult to model and analyze because of the complexity of these systems. Some of the earliest work to introduce concepts related to attack trees include the work of Salter *et al.* [40] who discussed modeling of adversaries and vulnerabilities to characterize attacks against information systems. These attack tree models can be used to identify and prioritize countermeasures in order to minimize risk. Schneier [41] proposed attack tree modeling and analysis considering cost, illustrating their approach in the context of a payment system. Tidwell *et al.* [42] took a goal-oriented approach by using attack trees to model the effects of exploits in order to facilitate the characterization of multistage attacks. However, it is difficult to manually analyze an attack tree for a system consisting of more than 100 nodes. Therefore, Sheyner *et al.* [14] developed a computer-aided tool to construct and analyze the attack trees.

The review of papers on attack graphs by Lippmann and Ingols [43] identifies several limitations, including scalability, enumerating the most important details of an attack, determining possible attack paths within a network, and making inferences to generate recommendations from these attack trees. To simplify the enumeration process, Mauw and Oostdijk [44] specified denotational semantics to dynamically manipulate attack trees during construction and analysis. To reduce complexity and simplify the modeling of coordinated attacks, Noel *et al.* [45] describe attack graph visualization techniques to interactively navigate a hierarchical attack graph.

Opel [46] extended a simple attack tree tool with AND and OR nodes by adding NAND, NOR, and XOR nodes as well as the ability to model leaf nodes with more than one parent. Camtepe and Yener [47] proposed an automatic method to detect partial attacks and verify the completeness of an attack tree. Eom *et al.* [48] developed a cyber attack model which simulates a network to expose vulnerabilities and identify suitable defenses. Jurgenson and Willemson [49] extended a multi-parameter attack tree model considering cost, feasibility of the attack, and skill level required by the attacker to the case where parameters are interval estimates rather than point values. Saini *et al.* [50] illustrate concepts from attack tree modeling on an online credential repository with an online certificate authority. In reality, an attacker may try alternative scenarios if some other subsets fail or may consider stop trying if a critical subset of attacks are not successful. To address such scenarios, Willemson *et al.* [51] proposed a serial model for attack trees which can model the temporal order of a sequence of attacks. To achieve computational consistency, they generalize the attack tree approach to accommodate arbitrary rooted directed acyclic graphs. Khand [52] illustrated the application of several nodes in the context of attack trees, including PAND (priority AND) node, k/n node, SEQ (sequential) node, CSUB (conditional subordination) node, and housing nodes which are used to characterize time varying changes to a configuration. Morais *et al.* [53] proposed a method to systematically derive scenarios for attack injection in order to test the security properties of the protocol.

Modeling cyber attacks should also consider defense strategies and system security improvements. Otherwise, inferences may not be accurate. To address this limitation of attack trees, Ekstedt and Sommestad [11] proposed attack-defense (AD) trees as an extension of attack trees, which also includes countermeasures. Kordy *et al.* [54], [55] showed that AD trees possess equivalent expressive power to two-player binary zero-sum extensive form games. This method encodes the strategy of a game as a satisfiability problem, which provides an alternative algebraic representation of the AD tree to facilitate the identification of a winning defense strategy. Kordy *et al.* also extended attack trees semantics to AD tree semantics and introduced the notion of attributes to enable quantitative analysis of AD trees [56] and discussed how AD trees can be used to answer intuitive questions that arise during practical security analysis [57]. To address limitations of their previous research, Kordy *et al.* [58] combined AD trees with Bayesian networks to evaluate probabilistic measures on AD scenarios involving dependent actions.

Ten *et al.* [59] proposed a supervisory control and data acquisition (SCADA) [60] security system for real-time monitoring, anomaly detection, impact analysis, and mitigation strategies. Wang *et al.* [61] proposed an improved AD tree (iADTree) that models AD trees based on financial costs such as return on attack (ROA) and defense cost to select countermeasures for a strategy to defend each attack path. Ji *et al.* [62] discussed the AD tree model for cyber physical systems considering both attack and defense cost with metrics such as probability of success, attack cost, revised cost, impact, revised impact, and defense cost while the tree is evaluated based on return on investment (ROI) and return on attack (ROA). Roy *et al.* [63] proposed attack countermeasure trees (ACT) where defenses can be applied to any node of the tree, qualitative and probabilistic analysis conducted, and select an optimal set of countermeasures. Whitley *et al.* [64] proposed a novel attribution approach that determines how values of child nodes are aggregated to form the attribute of a parent node in a manner that enables better understanding of interactions between weak and strong links in system security. Bagnato *et al.* [65] created an AD tree for an RFID-based goods management system and explored how to use attributes for AD nodes to assign and aggregate values to obtain performance indicators of security. Dewri *et al.* [66] developed a cost-benefit analysis for optimal security hardening, modeling AD interaction as an arms race and multi-objective optimization considering the cost of implementation and cost of residual damage.

The primary benefits of attack trees are that they allow the defender to identify potential attacks and suitable countermeasures. Moreover, attack trees are "self-documenting" in nature, which facilitates easy interpretation by the modeler. Disadvantages of this approach include the difficulty to enumerate all actions of an attacker and lack of expressivity for modeling attacks involving concurrent actions.

Authorized licensed use limited to: CNR Area Ricerca Genova. Downloaded on September 09,2025 at 16:49:37 UTC from IEEE Xplore. Restrictions apply.

III. Policy Recommendations

Tools and standards are needed to ensure cyber risk management is conducted in a consistent manner. Organizations should assess how their present practices correspond to modeling approaches. This would enable them to identify what existing tools that closely parallel their practices. From this baseline, they could then enhance their modeling practices by identify shortcomings in their existing practices and additional areas that are deemed necessary according to their systems and organization.

IV. Conclusion and Future Research

This paper provided a survey of fault and attack tree modeling and analysis for cyber risk management. The basic elements of these modeling approaches were reviewed. A variety of modeling and applications were given as well as a discussion of limitations of the approaches.

Future research will review additional cyber risk modeling paradigms such as event trees, binary decision diagrams, Petri nets, and Markov models to identify the limitations of their expressivity and ability to quantify and mitigate risk. We will then develop theoretical generalizations to these mathematical modeling techniques in order to overcome these limitations and apply them to systems to demonstrate these enhancements.

Acknowledgment

This work was supported by the Naval Air Warfare Center (NAVAIR) under contract N00421-16-T-0373.

References

[1] H. Watson, "Bell telephone laboratories. launch control safety study," *Bell Telephone Laboratories, Murray Hill, NJ USA*, 1961.

[2] W. Vesely, F. Goldberg, N. Roberts, and D. Haasl, "Fault Tree Handbook," DTIC Document, Tech. Rep., 1981.

[3] H. Kravitz, G. Driessen, R. Gomberg, and A. Korach, "Accidental falls from elevated surfaces in infants from birth to one year of age," *Pediatrics*, vol. 44, no. 5, pp. 869–876, 1969.

[4] M. Roth and P. Liggesmeyer, "Modeling and analysis of safety-critical cyber physical systems using state/event fault trees," in *International Conference on Computer Safety, Reliability and Security*, 2013.

[5] C. Lee, "Representation of switching circuits by binary-decision programs," *Bell System Technical Journal*, vol. 38, no. 4, pp. 985–999, 1959.

[6] T. W. Manikas, M. A. Thornton, and D. Y. Feinstein, "Modeling system threat probabilities using mixed-radix multiple-valued logic decision diagrams." *Multiple-Valued Logic and Soft Computing*, vol. 24, no. 1-4, pp. 135–149, 2015.

[7] C. Petri and W. Reisig, "Petri net," *Scholarpedia*, vol. 3, no. 4, p. 6477, 2008.

[8] S. Bernstein, "Sur l'extension du théorème limite du calcul des probabilités aux sommes de quantités dépendantes," *Mathematische Annalen*, vol. 97, no. 1, pp. 1–59, 1927.

[9] D. Kim, T. Lee, S. Jung, H. In, and H. Lee, "Cyber threat trend analysis model using HMM," in *International Symposium on Information Assurance and Security*. IEEE, 2007, pp. 177–182.

[10] B. Schneier, "Attack trees," *Dr. Dobbs journal*, vol. 24, no. 12, pp. 21–29, 1999.

[11] M. Ekstedt and T. Sommestad, "Enterprise architecture models for cyber security analysis," in *Power Systems Conference and Exposition*, 2009, pp. 1–6.

[12] B. Harpel, J. Dugan, I. Walker, and J. Cavallaro, "Analysis of robots for hazardous environments," in *IEEE Annual Reliability and Maintainability Symposium*, 1997, pp. 111–116.

[13] K. Reay and J. Andrews, "A fault tree analysis strategy using binary decision diagrams," *Reliability engineering & system safety*, vol. 78, no. 1, pp. 45–56, 2002.

[14] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing, "Automated generation and analysis of attack graphs," in *IEEE Symposium on Security and privacy*, 2002, pp. 273–284.

[15] E. Ruijters and M. Stoelinga, "Fault tree analysis: A survey of the state-of-the-art in modeling, analysis and tools," *Computer science review*, vol. 15, pp. 29–62, 2015.

[16] C. Ericson and C. Ll, "Fault tree analysis," in *System Safety Conference, Orlando, Florida*, 1999, pp. 1–9.

[17] R. Barlow and F. Proschan, "Importance of system components and fault tree events," *Stochastic Processes and their Applications*, vol. 3, no. 2, pp. 153–173, 1975.

[18] W. Lee, D. Grosh, F. Tillman, and C. Lie, "Fault tree analysis, methods, and applications - a review," *IEEE Transactions on Reliability*, vol. 34, no. 3, pp. 194–203, 1985.

[19] M. Locks, "Modularizing, minimizing, and interpreting the K&H fault-tree," *IEEE Transactions on Reliability*, vol. 30, no. 5, pp. 411–415, 1981.

[20] O. Coudert and J. Madre, "Metaprime: An interactive fault-tree analyzer," *IEEE Transactions on Reliability*, vol. 43, no. 1, pp. 121–127, 1994.

[21] Y. Dutuit and A. Rauzy, "A linear-time algorithm to find modules of fault trees," *IEEE Transactions on Reliability*, vol. 45, no. 3, pp. 422–425, 1996.

[22] C. Shannon, "The synthesis of two-terminal switching circuits," *Bell System Technical Journal*, vol. 28, no. 1, pp. 59–98, 1949.

[23] J. Dugan, S. Bavuso, and M. Boyd, "Dynamic fault-tree models for fault-tolerant computer systems," *IEEE Transactions on Reliability*, vol. 41, no. 3, pp. 363–377, 1992.

[24] S. Junges, D. Guck, J. Katoen, and M. Stoelinga, "Uncovering dynamic fault trees," in *Proceedings of Dependable Systems and Networks*, 2016.

[25] J. Dugan, B. Venkataraman, and R. Gulati, "DIFTree: A software package for the analysis of dynamic fault tree models," in *IEEE Annual Reliability and Maintainability Symposium*, 1997, pp. 64–70.

[26] R. Gulati and J. Dugan, "A modular approach for analyzing static and dynamic fault trees," in *Annual Reliability and Maintainability Symposium*, 1997, pp. 57–63.

[27] J. Dugan, K. Sullivan, and D. Coppit, "Developing a low-cost high-quality software tool for dynamic fault-tree analysis," *IEEE Transactions on Reliability*, vol. 49, no. 1, pp. 49–59, 2000.

[28] S. Amari, G. Dill, and E. Howald, "A new approach to solve dynamic fault trees," in *Annual Reliability and Maintainability Symposium*, 2003, pp. 374–379.

[29] S. Junges, D. Guck, J. Katoen, A. Rensink, and M. Stoelinga, "Fault trees on a diet," in *International Symposium on Dependable Software Engineering: Theories, Tools, and Applications*. Springer, 2015, pp. 3–18.

[30] J. Krcál and P. Krcál, "Scalable analysis of fault trees with dynamic features," in *IEEE International Conference on Dependable Systems and Networks*, 2015, pp. 89–100.

[31] R. Maaskant, "Interactive visualization of fault trees," 2016.

[32] I. Fovino, M. Masera, and A. De Cian, "Integrating cyber attacks within fault trees," *Reliability Engineering & System Safety*, vol. 94, no. 9, pp. 1394–1402, 2009.

[33] I. Hong, H. Youn, I. Chun, and E. Lee, "Autonomic computing framework for cyber-physical systems," in *International Conference on Advances in Computing, Control, and Telecommunication Technologies*, 2011, pp. 148–151.

[34] P. Patil, P. Zavarsky, D. Lindskog, and R. Ruhl, "Fault tree analysis of accidental insider security events," in *International Conference on Cyber Security*. IEEE, 2012, pp. 113–118.

[35] Y. Hashimoto, T. Toyoshima, S. Yogo, M. Koike, T. Hamaguchi, S. Jing, and I. Koshijima, "Safety securing approach against cyber-attacks for process control system," *Computers & Chemical Engineering*, vol. 57, pp. 181–186, 2013.

[36] A. Kornecki and M. Liu, "Fault tree analysis for safety/security verification in aviation software," *Electronics*, vol. 2, no. 1, pp. 41–56, 2013.

[37] J. Kühn, P. Schoonbrood, A. Stollenwerk, C. Brendle, N. Wardeh, M. Walter, R. Rossaint, S. Leonhardt, S. Kowalewski, and R. Kopp, "Safety conflict analysis in medical cyber-physical systems using an smt-solver." in *Software Engineering*, 2015, pp. 19–23.

Authorized licensed use limited to: CNR Area Ricerca Genova. Downloaded on September 09,2025 at 16:49:37 UTC from IEEE Xplore. Restrictions apply.

[38] G. Sabaliauskaite and A. Mathur, "Aligning cyber-physical system safety and security," in *Complex Systems Design & Management Asia*. Springer, 2015, pp. 41–53.

[39] M. S. Pallos, "Attack trees: It's jungle out there," in *Business Forum*, 2003.

[40] C. Salter, O. S. Saydjari, B. Schneier, and J. Wallner, "Toward a secure system engineering methodolgy," in *Proceedings of the 1998 workshop on New security paradigms*. ACM, 1998, pp. 2–10.

[41] B. Schneier, "Attack trees," *Dr. Dobb's Journal*, 2001.

[42] T. Tidwell, R. Larson, K. Fitch, and J. Hale, "Modeling internet attacks," in *IEEE Proceedings of Workshop on Information Assurance and security*, vol. 59, 2001.

[43] R. Lippmann and K. Ingols, "An annotated review of past papers on attack graphs," DTIC Document, Tech. Rep., 2005.

[44] S. Mauw and M. Oostdijk, "Foundations of attack trees," in *International Conference on Information Security and Cryptology*. Springer, 2005, pp. 186–198.

[45] S. Noel, M. Jacobs, P. Kalapa, and S. Jajodia, "Multiple coordinated views for network attack graphs," in *IEEE Workshop on Visualization for Computer Security*. IEEE, 2005, pp. 99–106.

[46] A. Opel, "Design and implementation of a support tool for attack trees," *Internship Thesis, Otto-von-Guericke University Magdeburg*, 2005.

[47] S. Camtepe and B. Yener, "A formal method for attack modeling and detection," *SA Camtepe, B. Yener*, 2006.

[48] J. Eom, Y. Han, S. Park, and T. Chung, "Active cyber attack model for network system's vulnerability assessment," in *IEEE International Conference on Information Science and Security*, 2008, pp. 153–158.

[49] A. Jürgenson and J. Willemson, "Processing multi-parameter attacktrees with estimated parameter values," in *International Workshop on Security*. Springer, 2007, pp. 308–319.

[50] V. Saini, Q. Duan, and V. Paruchuri, "Threat modeling using attack trees," *Journal of Computing Sciences in Colleges*, vol. 23, no. 4, pp. 124–131, 2008.

[51] A. Jürgenson and J. Willemson, "Serial model for attack tree computations," in *International Conference on Information Security and Cryptology*. Springer, 2009, pp. 118–128.

[52] P. Khand, "System level security modeling using attack trees," in *IEEE International Conference on Computer, Control and Communication*, 2009, pp. 1–6.

[53] A. Morais, E. Martins, A. Cavalli, and W. Jimenez, "Security protocol testing using attack trees," in *IEEE International Conference on Computational Science and Engineering*, vol. 2, 2009, pp. 690–697.

[54] B. Kordy, S. Mauw, M. Melissen, and P. Schweitzer, "Attack-defense trees and two-player binary zero-sum extensive form games are equivalent," in *International Conference on Decision and Game Theory for Security*. Springer, 2010, pp. 245–256.

[55] B. Kordy, S. Mauw, S. Radomirović, and P. Schweitzer, "Foundations of attack–defense trees," in *International Workshop on Formal Aspects in Security and Trust*. Springer, 2010, pp. 80–95.

[56] ——, "Attack-defense trees," *Journal of Logic and Computation*, 2012.

[57] B. Kordy, S. Mauw, and P. Schweitzer, "Quantitative questions on attack-defense trees," in *International Conference on Information Security and Cryptology*. Springer, 2012, pp. 49–64.

[58] B. Kordy, M. Pouly, and P. Schweitzer, "Probabilistic reasoning with graphical security models," *Information Sciences*, vol. 342, pp. 111–131, 2016.

[59] C. Ten, G. Manimaran, and C. Liu, "Cybersecurity for critical infrastructures: attack and defense modeling," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 40, no. 4, pp. 853–865, 2010.

[60] I. Fovino, M. Masera, L. Guidi, and G. Carpi, "An experimental platform for assessing scada vulnerabilities and countermeasures in power plants," in *IEEE International Conference on Human System Interaction*, 2010, pp. 679–686.

[61] P. Wang and J. Liu, "Improvements of attack-defense trees for threat analysis," in *Advances in Intelligent Systems and Applications-Volume 2*. Springer, 2013, pp. 91–100.

[62] X. Ji, H. Yu, G. Fan, and W. Fu, "Attack-defense trees based cyber security analysis for cpss," in *IEEE International Conference on Software Engineering, Artificial Intelligence, Networking and Parallel/Distributed Computing*, 2016, pp. 693–698.

[63] A. Roy, D. Kim, and K. Trivedi, "Cyber security analysis using attack countermeasure trees," in *Proceedings of the Sixth Annual Workshop on Cyber Security and Information Intelligence Research*. ACM, 2010, p. 28.

[64] J. Whitley, R. Phan, J. Wang, and D. Parish, "Attribution of attack trees," *Computers & Electrical Engineering*, vol. 37, no. 4, pp. 624–628, 2011.

[65] A. Bagnato, B. Kordy, P. H. Meland, and P. Schweitzer, "Attribute decoration of attack-defense trees," *International Journal of Secure Software Engineering*, vol. 3, no. 2, pp. 1–35, 2012.

[66] R. Dewri, I. Ray, N. Poolsappasit, and D. Whitley, "Optimal security hardening on attack tree models of networks: A cost-benefit analysis," *International Journal of Information Security*, vol. 11, no. 3, pp. 167–188, 2012.

978-1-5090-6356-7/17/$31.00 ©2017 IEEE

Authorized licensed use limited to: CNR Area Ricerca Genova. Downloaded on September 09,2025 at 16:49:37 UTC from IEEE Xplore. Restrictions apply.