An Evolutionary Approach of Attack Graphs and Attack Trees: A Survey of Attack Modeling

S. Haque, M. Keffeler, T. Atkison
Computer Science Department, University of Alabama, Tuscaloosa, AL, USA

Abstract— *The advancement of modern day computing has led to an increase of threats and intrusions. As a result, advanced security measures and threat analysis models are necessary to detect these threats and identify protective measures needed to secure a system. The most popular forms of attack modeling today are attack graphs and attack trees. This literature summarizes the different approaches through an extensive survey of the relevant papers and identifies the current challenges, requirements and limitations of efficient attack modeling.*

Keywords: Attack Graphs, Attack Trees, Survey.

1. Introduction

Computer technology has become ubiquitous with the increasing trend of computational capability into devices used in our everyday life. The dependency of these devices on network services and applications marks network security as a demanding research domain. Software bugs, security policy errors, inefficient network configuration can cause security violations any time in a system. A person with a malicious intent can make attempts to gain unauthorized access using these vulnerabilities. These attempts are termed an "attack" in computer security, defined by IETF as "an intentional act by which an entity attempts to evade security services and violate the security policy of a system" [1]. Attack representation models (ARMs) are the most effective means of analysis in these scenarios. Attack graph and attack tree are the most popular forms of representation models.

1.1 Attack Representation Model

Intrusion detection systems or other security components like firewalls generate security alerts if any vulnerable activities are observed by these systems. Alerts generated by these sources are isolated, and efficient detection of an attack scenario requires correlating these isolated alerts. Attack representation modeling is the process of identifying the relations between system alerts and developing an attack scenario recognition system. The purpose of the ARM is to determine the path of an attack and generate reports accordingly rather than describing the attack steps. Cheung et al. identified the necessary steps to develop a model which can recognize cyber attack scenarios successfully [2].

The initial step requires identifying the attacks and dividing them into attack subgoals until each of the logical attacks are identified by the detection systems. In the next phase of the modeling, these attacks need to be attributed based on the observed events, system states and interfaces. In the final stage, the relationship among these attacks needs to be developed based on the temporal relationship (the sequence of identified attacks), attribute-value relationship (attack might be generated from the similar sources) and prerequisite relationship (one attack triggers another attack) [2].

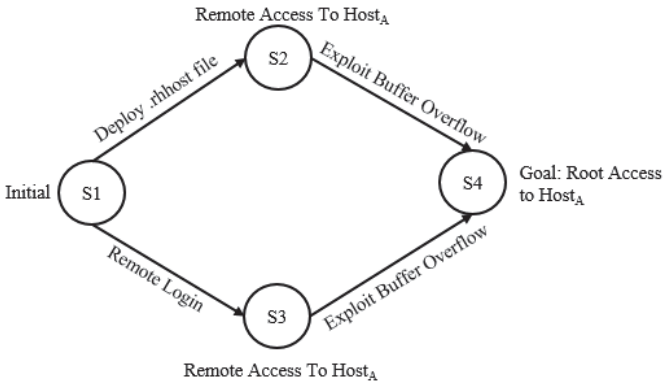

Fig. 1
ATTACK GRAPH BASED ON EXAMPLE NETWORK

1.2 Attack Graph

Attack graphs represent a detailed view of system security by determining if an attacker can reach the final goal state(s) by penetrating the security holes of the system from an initial state. These graphs are composed of nodes and edges where the representation of these components change with the definition of a particular attack graph. Typically, nodes in an attack graph represent states and the edges refer to the transition of the different states defined through various post- and pre-conditions.

For example, let's consider a network where $Host_A$ and $Host_B$ are two workstations connected to the Internet through a switch. $Host_B$ is owned by a malicious user who wants to gain access to $Host_A$. Because $Host_B$ knows the network address of $Host_A$, he can either use the remote login feature that is used by trusted users or deploy a .rhhost file utilizing an FTP vulnerability. In both scenarios, the attacker can create a trusted relationship with the target machine and easily exploit the buffer overflow to gain root access. In this case, the intruder can deploy the required binary codes or create the file locally.

Figure 1 shows a simple attack graph which is generated based on the scenario described above. Here S1 denotes the initial state and S4 denotes the goal of the attacker i.e. gaining

root access to $Host_A$. S2 and S3 are intermediary states where the attacker gets remote access to $Host_A$ by deploying the .rhhost file and performing the remote login operation respectively. Edges in this figure describe the transition from one state to the next through actions from the attacker.

1.3 Attack Tree

Attack trees are another powerful approach of modeling the security vulnerabilities of information systems. They analyze different security threats, identify different paths to achieve the goal, and build a tree structure that describes how a threat helps malicious users to reach their goal. In the tree structure, the elementary attacks are placed at the leaf level and the primary attack is placed at the root.

The internal (non-leaf) nodes in the tree represent a combined attack of the elementary nodes or non-leaf nodes located in the next higher level of the tree. In the top-down approach, the internal nodes are actually considered as a refinement of the higher level nodes and can be either conjunctive (aggregation or "AND" nodes) or disjunctive (choice or "OR" nodes). For conjunctive refinement, all the immediate child nodes will need to be in action to achieve the goal. However, for disjunctive refinement, any attack will be sufficient to fulfill the goal [3].

The example described above of $Host_B$ trying to gain root access to $Host_A$ would produce the attack tree shown in Figure 2. The root of the attack tree is the goal of the attacker. To get root access to $Host_A$, the attacker first needs to get remote access to the system; therefore, this will be in the next level of the attack tree. The final level of the attack tree will consist of the actions like deploying .rhhost and remote login operation. As either of these actions allow a user to get remote access, these two nodes form a disjunctive connection with their parent node.

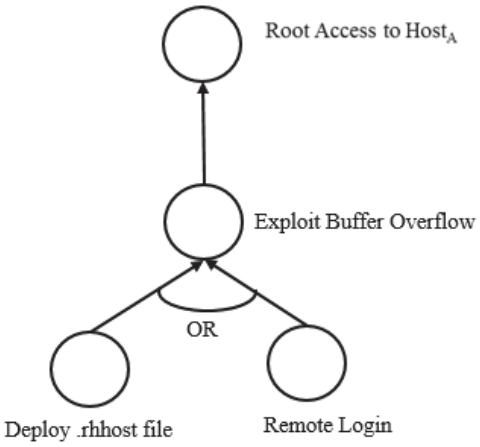

Fig. 2
ATTACK TREE BASED ON EXAMPLE NETWORK

1.4 General Attack Analysis Approaches

Attack trees and graphs are generally used to create a model to analyze attacks that occur in computer networks. Attack and Protection tree, Defense Tree and Attack-Defense tree were a few of such models built upon the regular attack tree and used to analyze attacks from both an attacker's and a defender's point of view [4], [5]. Edge et al. built a Protection tree to identify the possible protection areas calculating the impact, probability and cost using the goal of the attacker analyzing the attack tree. In this scenario, Protection tree and Attack tree are two separate entities and Protection tree is built upon the calculated data found from attack tree [4]. Bistarelli et al. have introduced Defense tree as an extension of attack tree which can be used to measure the return of actions for both attacker and defender [5]. Two indexes, Return on Investment (ROI) and Return on Attack (ROA) defines the success of defender in terms of applying security measure against an attack and success of attacker in terms of successful exploitation of a vulnerability respectively. Kordy et al. have devised their model to overcome some limitations of attack trees. First, attack trees are not capable of identifying the interaction between attacks in a system and defenses implanted against the particular attack. Second, attack tree cannot properly visualize the evolution of security for a system from the action of an attacker and defender. Unlike Protection tree, both Defense tree and Attack-Defense tree accommodate two types of nodes to refer actions regarding attack and defense. These models are also capable of analyzing the economic effectiveness of actions taken either for attack and defense.

Recent research reveals the application of attack trees through game-theoretic analysis of attack scenarios. In a game-theoretic approach, an attacker's best option is deduced based on multi-object optimization, and an administrator or defender improves his information by keeping track of every action of the attacker [6]. Game theory is considered to be an adaptive and faster algorithm to determine an attacker's future action and behavior. Kordy et al. showed the equivalence of both the Attack-Defense tree and binary Zero-Sum extensive game in terms of maintaining outcome and structure while conversion between these two forms [7]. The game theoretic analysis involves actions of attackers followed by the actions of defenders in the form of an attack graph and eventually helps in determining the preventive measures for each attack.

Attack tree computation with multiple parameters illustrates the complete analysis associated with an attacker's action. Initially, attack trees were analyzed only with cost. Buldas et al. have included success probability and penalty both in the case of failure and success in their analysis [8]. Jürgenson et al. later introduced the concept of *Gains* and *Expanses* calculated from attack cost and achievement, probability and penalties in their analysis which successfully helped in analyzing the possible attack path with more realistic approach [9].

2. Analysis of Core Papers

This section will present a survey of core papers related to alert correlation, attack graph generation and attack tree reduction. Motivations, contributions, and limitations of each paper will be discussed.

2.1 Statistical causality analysis of INFOSEC alert data [10]

A statistical approach of identifying correlated alerts can be a vital part of an attack graph generation process. Previous alert correlation systems depended on prior knowledge and consequences of alerts [2], [11], [12], and lacked the ability to detect a new attack. Qin et al. were motivated by these limitations to create a new approach to alert correlation. They created a four step system that used time series and statistical analysis to combine the low-level alerts using their attribute information, thus ensuring a reduction of the high volume of alerts.

The first step was alert aggregation and clustering. Clustering changed low level alerts into aggregated alerts. Then, the alerts were prioritized depending on the relationship to the networks, hosts and goals of the attacks. The third step was to create an alert time series variable for the aggregated alerts. Finally, attack scenarios were constructed in the alert correlation step based on the correlated alerts generated through causality analysis. Since Qin's approach does not depend on previous knowledge for pattern matching, it is capable of discovering new attacks.

There are several limitations to Qin's approach. First of all, it lacks the ability to generate scenarios without the intervention of a system analyst, and advanced knowledge is required to inspect the alert candidates produced from the Granger Causality Test during the alert correlation step. Qin's approach also suffers from false causality alert scenarios when the volume of background alerts is large. Control probability tables in the attack prioritization phase are not adaptive and updated according to the mission goals, but rather developed based on prior experience or domain knowledge.

2.2 A scalable approach to attack graph generation [13]

Research on attack graphs have been around for decades, but most efforts suffer from proper scalability and lack of logical formulation. Furthermore, attack graph tools require additional input in a specialized data format and often produce complex, unclear attack graphs. Ou [14] proposes a logical attack graph that produces a derivation trace and generates an attack graph using the trace in quadratic time. It also clearly specifies the configuration information of the system and the potential privileges of the attacker.

Ou's approach is designed after Sheynar's MulVAL system [15], but modifies it to accommodate attack simulation traces. These traces, which record the trace of the evaluation performed by XSB, are passed to the graph builder which generates a directed graph.

Ou's method shows a definite superiority over the other models in terms of efficiency. It can generate an attack graph polynomial to the network size. However, there are some limitations. To create a complete attack graph, attack conditions must be expressed in propositional formulas; otherwise, this method overlooks that particular attack condition. Also, an attack graph generated in this way contains loops which restrict it from converting a graph to a tree. Although a possible solution to this problem is presented, it is not explained how this solution affects the proposed algorithm.

2.3 Alert correlation for extracting attack strategies [14]

Alert correlation is an integral part of designing an efficient intrusion detection and response system. Identifying the attack strategy and analyzing a large volume of alerts generated by the Intrusion Detection System (IDS) is the primary goal of alert correlation. Zhu was motivated by the fact that previous alert correlation approaches, that were based on feature similarity, known scenarios or cause and effect relationships, lacked features to identify the relationship between alerts. Some of the approaches cannot identify causal relationships, some are applicable only in familiar situation, and some require predefined rules and expected consequences [15], [16], [17]. Therefore, Zhu et al. developed a correlation method which enables automatic extraction of attack strategy from intrusion alerts and without prior knowledge of the alerts.

Intrusion or anomaly detection not only depends on the detail information of a system, proper understanding about the anomalous behavior of the system that occurred because of different attacks also plays an important role. Building an exact profile of the general behavior of the system meets the first requirement in this case, while discovering attack strategy from an IDS generated alert fulfill the second criteria. The proposed approach of this paper is developed based on the concept of neural networks. A knowledge base is first built through supervised learning and stored relationship attributes between alerts, such as correlation strength and average time difference between two alerts. The correlation engine uses Multilayer Perceptions (MLP) and Support Vector Machines (SVM) to assign correlation probability to an alert. This correlation engine is then used to generate hyper-alert graphs and attack graphs that represent real attack scenario.

Zhu et al. proposed a different technique for alert correlation by using a combination of both MLP and SVM to determine the relationship between alerts. Like other proposed methods, it also has some limitations. As this method follows a supervised learning approach, both MLP and SVM need manually generated and labeled training. This methodology requires additional effort and might produce an erroneous result. Also, the attack graph contains loops, but no approaches to eliminate the loops are discussed.

2.4 Scalable attack representation model using logic reduction techniques [18]

Attack graphs and attack trees are the most popular attack modeling techniques, and various models have been developed to use them. These models include automatic construction of attack scenarios in both graph and tree forms, but none of these approaches are proven to be efficient [19], [20], [21]. Analyzing an attack graph for seemingly larger networks suffers from the state-explosion problem while attack tree lacks the feature of covering all the attack scenarios [13], [18]. Efforts to generate attack graphs and then transform them into a tree structure also failed due to the lack of scalability as the methods generate nodes at an exponential rate. These limitations motivated Hong et al. to introduce two logic reduction techniques to enable an automatic transformation of an attack tree as well as ensure a reduction of the size of the tree: Full Path Calculation (FPC) approach, and Incremental Path Calculation (IPC) approach.

In the FPC approach, logic reduction initially requires the full attack tree to be represented in a logical expression. The reduction process eliminates the attack sequence and groups similar nodes together. In this case, an element from the attack tree is selected first; then, the selected node is factorized from the complete logical expression. This process iteratively runs with selecting a common element from the rest of the nodes of the graph. Paths with similar nodes are evaluated as a container of similar information in this approach. However, FPC of the complete attack tree suffers from the lack of efficiency when all the paths are included in the computation.

On the other hand, the IPC approach minimizes repetition of the node through recursive expansion of attack path. IPC considers reachability information from every node and thus overcomes the problem of full path calculation. Reachability information against each node is separately maintained while calculating the attack path. The IPC approach follows this information while evaluating a particular node and includes the next attack path. Eventually, the process stops when all the possible attack paths are included.

Although the evaluation and complexity analysis study of the logical reduction approaches show the efficient utilization of the attack tree after transformation, there are limitations to Hong's method. These proposed algorithms perform well in small attack trees, but larger attack tree FPC is affected by the exponential number of nodes that it requires to process and IPC suffers from inefficient memory allocation.

2.5 Efficient Attack Graph Analysis through Approximate Inference [22]

Protecting networks through effective vulnerability identification and prevention is often affected by the lack of expertise and requires interruption to the system. Therefore, optimized resources for protection need to be determined by risk-driven security measures. Dependencies between the attacks are mostly ignored in these. Muñoz-González et al. provides an elaborate discussion on the Bayesian Attack Graph (BAG) which maintains a rigid relationship among random variables and also enables modeling uncertainty about an intruder's intention and capabilities [25]. Variable elimination (VE) and Junction Tree (JT) also allow exact inference in a BAG by computing unconditional probabilities for each node. However, exact inference can only be applied to a smaller graph and computation is marked as an NP-Hard problem. Approximate inference through Loopy Belief Propagation (LBP) is introduced to overcome the issue with network size.

Muñoz-González combined static and dynamic risk analysis based on the BAG. In both models, the BAG is first built based on either the network topology or the analysis on alerts. The static analysis model, then, computes the conditional probability tables of the nodes based on the Common Vulnerability Scoring System score. LBP is used as an approximate inference technique to identify the vulnerable points of the network. Muñoz-González develop two different types of LBP: Serial Loopy Belief Propagation (S-LBP) and Parallel Loopy Belief Propagation (P-LBP). In the case of S-LBP, messages are computed iteratively for each node until it reaches a maximum range of iteration or the value converges. A scheduling technique is applied for favorable convergence and better efficiency of this approach. P-LBP allows simultaneous updating of messages for all variables and factor nodes based on the value achieved in the earlier iterations. In the dynamic model, the conditional probabilities are recomputed based on the detected attack in a network at any point in time. In this case, the state of the variable of the compromised node is set to 1. This updated state of one node eventually affects the posterior probabilities of rest of the nodes.

Muñoz-González showed an experimental result of the efficiency of the proposed S-LBP and P-LBP in terms of accuracy, convergence and execution time. However, Loopy Belief Propagation (LBP) cannot always guarantee convergence [34], and the convergence of LBP does not always ensure correctness of the probability estimations. The alternative approaches to ensure convergence are found to be inefficient in most of the scenarios [29].

3. Synthesis of core papers

The approach for building an attack graph to attack tree conversion technique requires identifying paths to construct the attack graph first and then eventually convert the attack graph into an attack tree. In a general attack tree modeling approach, attack goals are decomposed into sub-goals until the sub-goals become atomic attacks [23]. Attack trees also can be constructed manually or using already developed attack tree designing tools [24], [25], [26]. Five core papers discussed in Section 2 highlight the various models of developing an attack graph based on the information acquired from different alerts and network configurations [13], [10], [14], the process of analyzing the attack path [27], and scalable representation of attack trees [18].

3.1 Attack Graph Modeling

Three of the five selected papers discuss different approaches to attack scenario modeling and generating attack graphs. These approaches are considered to be the initial step to formulate attack trees from the general alerts and system configuration. In attack graph generation, though system configuration plays an important role, determining the model of an attack (identifying attack relevant alerts and correlating those alerts based on their attributes) is also vital. Proposed methods of Qin et al. and Zhu et al. concentrate determining relationship between alerts through alert correlation approach while model-checking approach is used by Ou et al. to develop attack graph in a scalable manner.

Both Qin et al. and Zhu et al. developed their model based on the dataset of Grand Challenge Problem (GCP) from DARPA. From this large volume of dataset, they extracted some important attributes which mostly described an alert and tried to find the relationship between these alerts based on those attribute values. Qin et al. defined the relationship between alerts based on the similarity of attribute values, while Zhu et al. selected six primary features which in some cases considered similar values in different attributes (e.g., the similarity of source IP of one alert with destination IP of another alert). Qin et al. thus reduced the number of alerts generated by different IDS or firewalls and eventually grouped the alerts. This is an essential approach in determining the particular set of information that must be analyzed from a large volume of data. Qin et al. then allowed system administrator's input to determine the priority of hyper-alerts generated through the fusion of alerts. The System administrator uses configuration information of the host and network to determine the rank of the alerts. Prioritized alerts were further arranged based on the time sequence and provided as the input of the Granger Casualty Test (GCT) to determine the ultimate relationship [10]. Zhu et al. considered the time series based relationship as a temporary measure and determined the causal relation based on forward and backward correlation strength. The temporal and causal relationship eventually built the cell weight of Alert Correlation Matrix (ACM). Zhu et al. also used Multilayer Perception(MLP) and Support Vector Machine(SVM) to build the relation between the new alert and the hyper-alert and also updated the ACM with new relationships and correlation weights [14]. Zhu et al. used a stronger measure to correlate different alerts and also generated a new attack scenario determining the relationship with the new alert and existing hyper-alert from the ACM. Alert aggregation approach used by Qin et al. can complement the proposed method of Zhu et al. to reduce the number of alerts and eventually can increase efficiency. Both these approaches require human intervention either for prioritizing alert or for supervised learning.

The logic-based approach proposed by Ou et al. mostly depended on the system configuration and defined the attack path based on the rule determined by the configuration [13]. The rule also included the possible vulnerabilities associated with a host and network. These features were already provided by MulVAL and Ou et al. additionally added the attack simulation trace to generate the final attack graph [28], [13]. Though this approach determined the attack path by iteratively looking into a large set of rules and associated attack traces - it lacked the feature of dynamically analyzing the data generated by various IDS and was highly dependent on rules and vulnerabilities set by the system administrator.

3.2 Attack Graph Analysis

Attack graph analysis is another important phase of constructing the proper model of attacks. It facilitates the process of assessing the risk and identifies the actual vulnerable point on the attack path statically or dynamically. Researchers proposed different matrices for risk assessment over the years - rate at which asset can be acquired, measure of risk based on weakest path, measurement based on number of attack, length of shortest path and standard deviation, normalized mean, median and mode of length of paths [29], [30], [31], [32]. Muñoz-González et al. developed their approach by analyzing an attack tree based on Bayesian Attack Graph(BAG) and used Common Vulnerability Scoring System(CVSS) values as a standard of measurement [22]. The Proposed approach identified attack path based on the pre-recorded information through static analysis and detected possible threat through dynamic analysis. The inclusion of Loopy Belief Propagation(LBP) to measure approximate inference enabled the algorithm to be applicable in the larger network. Attack graph analysis designed with this approach basically helps in devising a concrete attack path removing the less vulnerable nodes from an attack graph and can help in reducing the false positive alerts using both static and dynamic analysis.

3.3 Logic Reduction

Nodes in attack trees are built with various forms of boolean algebra where the status of a particular network or host plays the role of a boolean variable. These forms of boolean logic can often be further reduced to a simplified structure for better understanding and faster analysis. Hong et al. proposed two logic reduction algorithms in their literature to simplify the structure of the attack trees [18]. These approaches can be applied partially or fully in an already constructed attack tree and can improve the performance of attack tree evaluation. In their first algorithm, all vulnerable nodes in an attack path are included with groups containing same nodes to define the simpler attack path. In their second approach, vulnerable nodes are iteratively included in a path based on the reachability of the initial attack node. Though both these approach suffers from memory and performance efficiency while deriving the simplified version of attack tree, proposed algorithms showed their superiority by constructing less-complex attack trees.

4. Conclusion

Throughout this literature study, we have explored different attack representation models, discussed their opportunities, as well as techniques to develop these models from raw alerts, and network and host configurations. We have also discussed

alert analysis methodologies and attack tree and attack graph techniques. Additionally, we have discussed the recent research conducted around these models. From our analysis, we have found that attack trees are more computationally efficient model for attack analysis. An advanced attack tree construction approach makes attack analysis more convenient in various environments and eventually make the real-time intrusion detection more efficient.

References

[1] R. W. Shirey, "Internet security glossary, version 2," 2007.

[2] S. Cheung, U. Lindqvist, and M. W. Fong, "Modeling multistep cyber attacks for scenario recognition," in *DARPA information survivability conference and exposition, 2003. Proceedings*, vol. 1. IEEE, 2003, pp. 284–292.

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

[4] K. S. Edge, G. C. Dalton, R. A. Raines, and R. F. Mills, "Using attack and protection trees to analyze threats and defenses to homeland security," in *MILCOM 2006-2006 IEEE Military Communications conference*. IEEE, 2006, pp. 1–7.

[5] S. Bistarelli, F. Fioravanti, and P. Peretti, "Defense trees for economic evaluation of security investments," in *First International Conference on Availability, Reliability and Security (ARES'06)*. IEEE, 2006, pp. 8–pp.

[6] Y. Luo, F. Szidarovszky, Y. Al-Nashif, and S. Hariri, "Game theory based network security," *Journal of Information Security*, vol. 1, no. 1, p. 41, 2010.

[7] 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.

[8] A. Buldas, P. Laud, J. Priisalu, M. Saarepera, and J. Willemson, "Rational choice of security measures via multi-parameter attack trees," in *International Workshop on Critical Information Infrastructures Security*. Springer, 2006, pp. 235–248.

[9] A. Jürgenson and J. Willemson, "Computing exact outcomes of multi-parameter attack trees," in *OTM Confederated International Conferences" On the Move to Meaningful Internet Systems"*. Springer, 2008, pp. 1036–1051.

[10] X. Qin and W. Lee, "Statistical causality analysis of infosec alert data," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2003, pp. 73–93.

[11] A. Valdes and K. Skinner, "Probabilistic alert correlation," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2001, pp. 54–68.

[12] P. A. Porras, M. W. Fong, and A. Valdes, "A mission-impact-based approach to infosec alarm correlation," in *International Workshop on Recent Advances in Intrusion Detection*. Springer, 2002, pp. 95–114.

[13] X. Ou, W. F. Boyer, and M. A. McQueen, "A scalable approach to attack graph generation," in *Proceedings of the 13th ACM conference on Computer and communications security*. ACM, 2006, pp. 336–345.

[14] B. Zhu and A. A. Ghorbani, "Alert correlation for extracting attack strategies," *IJ Network Security*, vol. 3, no. 3, pp. 244–258, 2006.

[15] O. Dain and R. K. Cunningham, "Fusing a heterogeneous alert stream into scenarios," in *Proceedings of the 2001 ACM workshop on Data Mining for Security Applications*, vol. 13. Citeseer, 2001.

[16] S. T. Eckmann, G. Vigna, and R. A. Kemmerer, "Statl: An attack language for state-based intrusion detection," *Journal of computer security*, vol. 10, no. 1, 2, pp. 71–103, 2002.

[17] P. Ning, Y. Cui, and D. S. Reeves, "Constructing attack scenarios through correlation of intrusion alerts," in *Proceedings of the 9th ACM conference on Computer and communications security*. ACM, 2002, pp. 245–254.

[18] J. B. Hong, D. S. Kim, and T. Takaoka, "Scalable attack representation model using logic reduction techniques," in *2013 12th IEEE International Conference on Trust, Security and Privacy in Computing and Communications*. IEEE, 2013, pp. 404–411.

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

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

[21] B. Schneier, *Secrets and lies: digital security in a networked world*. John Wiley & Sons, 2011.

[22] L. Muñoz-González, D. Sgandurra, A. Paudice, and E. C. Lupu, "Efficient attack graph analysis through approximate inference," *arXiv preprint arXiv:1606.07025*, 2016.

[23] 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.

[24] K. S. Edge, "A framework for analyzing and mitigating the vulnerabilities of complex systems via attack and protection trees," DTIC Document, Tech. Rep., 2007.

[25] K. Edge, R. Raines, M. Grimaila, R. Baldwin, R. Bennington, and C. Reuter, "The use of attack and protection trees to analyze security for an online banking system," in *System Sciences, 2007. HICSS 2007. 40th Annual Hawaii International Conference on*. IEEE, 2007, pp. 144b–144b.

[26] A. T. Ltd., "Securitree," http://www.amenaza.com/, [Accessed : 12-31-2016].

[27] L. Muñoz-González, D. Sgandurra, M. Barrere, and E. C. Lupu, "Exact inference techniques for the analysis of bayesian attack graphs," *arXiv preprint arXiv:1510.02427*, 2015.

[28] X. Ou, S. Govindavajhala, and A. W. Appel, "Mulval: A logic-based network security analyzer." in *USENIX security*, 2005.

[29] R. Lippmann, K. Ingols, C. Scott, K. Piwowarski, K. Kratkiewicz, M. Artz, and R. Cunningham, "Validating and restoring defense in depth using attack graphs," in *MILCOM 2006-2006 IEEE Military Communications conference*. IEEE, 2006, pp. 1–10.

[30] J. Pamula, S. Jajodia, P. Ammann, and V. Swarup, "A weakest-adversary security metric for network configuration security analysis," in *Proceedings of the 2nd ACM workshop on Quality of protection*. ACM, 2006, pp. 31–38.

[31] N. Idika and B. Bhargava, "Extending attack graph-based security metrics and aggregating their application," *IEEE Transactions on dependable and secure computing*, vol. 9, no. 1, pp. 75–85, 2012.

[32] C. Phillips and L. P. Swiler, "A graph-based system for network-vulnerability analysis," in *Proceedings of the 1998 workshop on New security paradigms*. ACM, 1998, pp. 71–79.