12-3-2012

HARMs: Hierarchical Attack Representation Models for Network Security Analysis

Jin Hong
University of Canterbury

Dong-Seong Kim
University of Canterbury

DOI: 10.4225/75/57b559a3cd8da
10th Australian Information Security Management Conference, Novotel Langley Hotel, Perth, Western Australia, 3rd-5th December, 2012

HARMS: HIERARCHICAL ATTACK REPRESENTATION MODELS FOR NETWORK SECURITY ANALYSIS

Jin Hong and Dong-Seong Kim
Dept. of Computer Science and Software Engineering, University of Canterbury
Christchurch, New Zealand
jho102@uclive.ac.nz, dongseong.kim@canterbury.ac.nz

Abstract

Attack models can be used to assess network security. Purely graph based attack representation models (e.g., attack graphs) have a state-space explosion problem. Purely tree-based models (e.g., attack trees) cannot capture the path information explicitly. Moreover, the complex relationship between the host and the vulnerability information in attack models create difficulty in adjusting to changes in the network, which is impractical for modern large and dynamic network systems. To deal with these issues, we propose hierarchical attack representation models (HARMs). The main idea is to use two-layer hierarchy to separate the network topology information (in the upper layer) from the vulnerability information of each host (in the lower layer). We compare the HARMs with existing attack models (including attack graph and attack tree) in model complexity in the phase of construction, evaluation and modification.

Keywords
Attack graph, attack tree, complexity analysis, hierarchical model, security assessment, scalability.

INTRODUCTION

All types of network systems, including critical cyber infrastructures down to small home networks, are threatened by the risk of cyber security attacks. To understand and trust the network security, one need not only consider the security mechanisms, but also incorporate the combined effects of various network vulnerabilities and their countermeasures.

To evaluate network security of a network system more efficiently, it is necessary to develop a valid and scalable network security model that illustrates and inherits the security risk properties of the system. There has been research which develops a science of security and security metrics as a major goal from wide range of disciplines (King, 2010). Measuring the security data specifies the cyber attack events, states and consequences, and the security data can be measured by means of testbed measurements (Sharma, Kalbarczyk, Barlow, & Iyer, 2011), emulations (Mirkovic *et al.*, 2010), honeypots (Alata, Nicomette, Kaaniche, Dacier, & Herrb, 2006), and monitoring the network data flow (Yan Zhu, Hu, Ahn, Huang, & Wang, 2012). Security metrics specify the security attributes (qualitative or quantitative), which can be used in the security analysis (Chew, Swanson, Stime, Bartol, Brown & Robinson, 2008). Attack models, such as attack graphs (AGs) (Sheyner, Haines, Jha, & Wing, May, 2002) or attack trees (ATs) (Schneier, 2000) are used for network security analysis.

Existing attack models suffer from scalability and dynamic adjustment problems. Firstly, the scalability problem occurs due to calculation of full attack paths, which has an exponential complexity. Previous researches on AGs show the scalability issues (Ou, Boyer, & McQueen, 2006; Ingols, Chu, Lippmann, Webster, & Boyer, 2009; Xie, Cai, Tang, Hu, & Chen, 2009), and their solutions only consider the subset of full attack paths. ATs (Schneier, 1999; Edge *et al.*, 2007; Edge, 2007; Moore, Ellison, & Linger, 2001; Dawkins & Hale, 2004; Saini, Duan, & Paruchuri, 2008) are non-state space models, but there is no generation method to construct tree-based attack models directly from the network system specifications. Secondly, dynamic adjustment issue occurs when there are changes in the network system, such as network topologies, vulnerabilities and system configurations. Those changes in the network system modify the attack model accordingly.

Both scalability and dynamic adjustment problems are critical to attack models, because they sometime make the practical use of attack models infeasible. To accommodate these problems, we propose two-layer Hierarchical Attack Representation Model (HARMs). The upper layer (or level) captures the network information (e.g., reachability), and the lower layer captures the vulnerability information of hosts in the network in the HARMs. Our contributions are summarised as follows:

- Propose HARMs to enhance scalability and dynamic adjustment problem in existing attack models.
- Compare complexities of the HARMs with AG and AT.

- Demonstrate equivalent security analysis of the HARMs and AG using an illustrative example.

The rest of the paper is as organised as follows. In Section 2, related work is introduced. In Section 3, complexity analysis of AG, AT, and HARMs is presented using an example network system. Section 4 summarises the complexity comparisons, and in Section 5, we conclude the paper.

RELATED WORK

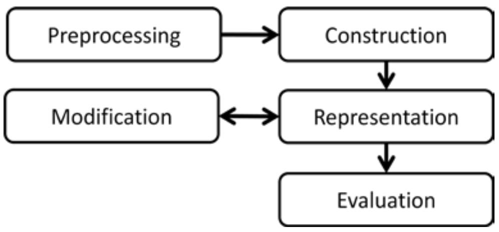


Figure 1. Attack Model Phases

There are different phases for attack models and they are shown in Figure 1. In the preprocessing phase, the network information is preprocessed for construction of an attack model. In the construction phase, the preprocessed network information is used to construct an attack model. In the representation phase, attack model is illustrated and stored. The security analysis is carried out during the evaluation phase, such as calculating all possible attack paths, probability of attack success. The modification phase captures the changes in the network system, and makes modifications to the attack model accordingly.

Attack models have scalability and dynamic adjustment problems in their phases. The scalability problem restricts the use of attack models for large size network systems. The dynamic adjustment problem restricts the real time use of attack models in dynamically changing network systems. There has been research to improve the AG construction and evaluation performance to avoid the exponential complexity (Sheyner *et al.*, May, 2002; Ou *et al.*, 2006; Ingols *et al.*, 2009; Xie *et al.*, 2009). Initially, AGs had a scalability problem in construction with an exponential complexity by using a full AG representation (Sheyner *et al.*, May, 2002). A scalable AG construction has been proposed by K. Ingols *et al.* (Ingols, Lippmann, & Piwowarski, 2006), where they used a Multiple-Prerequisite (MP) graph to construct the attack model in polynomial complexity. But the MP graph representation becomes harder to represent a large size network system, because all network components are presented in a single layer. X. Ou *et al.* (Ou *et al.*, 2006) presented a Logical AG, which has a polynomial construction complexity. K. Ingols *et al.* (Ingols *et al.*, 2009) used a network security analysis tool named Network Security Planning Architecture (NetSPA) to analyse the network security. However, NetSPA could not handle more than 17 hosts using a full AG representation. Predictive graphs and MP graphs are used to avoid the scalability problem in the construction phase and grouping of similar subnet and calculations of likely attacks are used to reduce the evaluation complexity. However, in a worst case, they need to consider full attack paths to assess the overall security. A. Xie *et al.* (Xie *et al.*, 2009) proposed the two-layered AG, where they used an AG representation in two layers but their focus is to use AG in the lower layer to capture attack paths of important edges of the AG in the upper layer. We focus on improving the efficiency of attack models and we use AG or AT in the lower layer to capture hosts information rather than attack paths for edges of AG in the upper layer. Also, our focus of evaluation is to consider all possible attack paths, their evaluation method does not capture them.

ATs are known to express its structure without state space explosion (Schneier, 2000). Previous research mainly discussed the application of ATs (Schneier, 1999; Edge *et al.*, 2007; Edge, 2007; Moore *et al.*, 2001; Dawkins & Hale, 2004; Saini *et al.*, 2008). Edge *et al.* (2007) and Moore *et al.* (2001) stated that the construction of AT is a manual task done by a security analyser (i.e. security expert) in their works. Dawkins *et al.* (2004) provided a general AT construction method using attack chains which are equivalent to AGs. Min-cuts (i.e. minimal cut set, which are considered as attack scenarios) from the attack chain is populated as AT branches. They used heuristic methods to evaluate the attack chains and ATs. Saini *et al.* (2008) described another construction method by decomposing the overall goal into sub-goals, and built the AT using the bottom up approach. This construction method has an exponential complexity, and methods to identify decomposable components are not given.

All previous research on attack models have scalability problem in both construction and evaluation. To avoid the scalability problem in AGs, researchers considered subset of a network system for construction, and most likely attacks for evaluation. The AT was constructed from min-cuts on existing AGs, or by decomposition. Both methods create an exponential number of states in ATs. In our proposed models, the construction phase groups the interaction between independent components in different layers, whereas in existing attack models, same interaction is repeated between independent components.

There are security solutions to dynamic network changes, such as changing firewall rules, and applying Intrusion Detection/ Protection System (Hwang & Gangadharan, 2001; Theodorakopoulos, Baras, & Le Boudec, 2008). But these security changes must be analysed using attack models to monitor how the network security has been affected. As far as we know, there is no previous research on solutions of dynamic adjustment problems for attack models.

HIERARCHICAL ATTACK REPRESENTATION MODELS

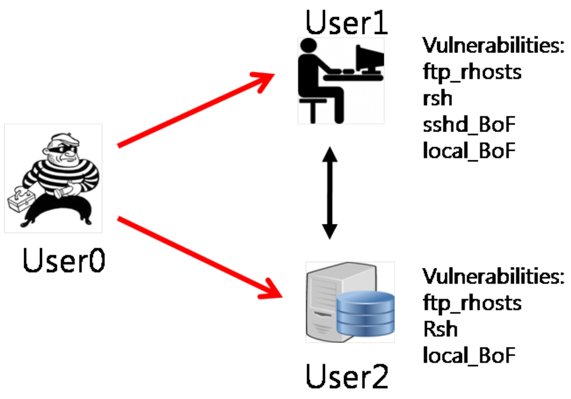
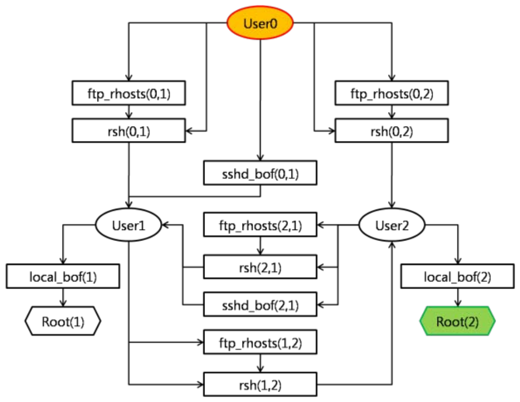


Figure 2. An Example Network System

We use an example network system shown in Figure 2 (Albanese, Jajodia, & Noel, 2012). We assume that User$_0$ is the attacker, and the root access of User$_2$ is the target. The attacker can exploit single or multiple vulnerabilities on each network host to gain the desired privilege. In Figure 2, we can see that the attacker can reach User$_2$ directly, or a route through User$_1$ after gaining a sufficient privilege. A corresponding AG is given in Figure 3, where network hosts and vulnerabilities are viewed in a single layer. In comparison, the AG in both the upper and the lower layers of the HARMs (in particular, HAGAG) is given in Figure 4.



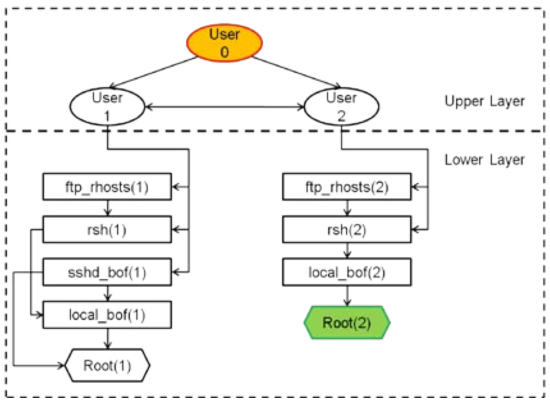


Figure 3. AG of the example network system *Figure 4. HAGAG of the example network system*

Any attack model can be considered in any layer of the HARMs. We consider the HARMs with the upper and the lower layers with AG or AT respectively, which are denoted as H[upper layer][lower layer] (e.g., HAGAT represents the HARMs with AG in the upper layer, and AT in the lower layer respectively). We will compare the complexity of attack models in each phase. The complexity analysis is based on the number of hosts and vulnerabilities. We denote n ($n > 0$) and m ($m > 0$) as the total number of hosts (nodes) and the average number of vulnerability for each host, respectively. We assume that nodes in the network system are fully connected to each other (i.e. a complete graph). In future, we will use other type of several network topologies as the work (Ou, Boyer, & McQueen, 2006) did. The detailed complexity in each phase will be described in the next subsections.

Construction

We consider the number of components and their connections to calculate the construction complexity. First, we consider the construction of the AG. Vulnerabilities in each host may have up to *m*-1 other vulnerabilities as a subsequent exploit. Hence, there are total *m*(*m*-1) number of possible vulnerability connections for each host. For each host, there are up to *n*-1 number of other host(s) to connect. Hence, there are total *n*(*n*-1) number of possible host connections. Therefore, the total construction complexity of the AG is $O(m^2n^2)$.

Second, we consider the construction of the AT. We consider bottom-up approach for an AT construction. The bottom-up approach is to build the AT from vulnerability information up to the host information. There is no generation method to construct ATs using the network configuration, so we consider all possible exploit sequences, and use each sequence as a branch in the AT. For *m* number of vulnerabilities, there are O(*m*!) number of possible exploit sequences for each connection. Similarly, for *n* number of hosts, there are O(*n*!) number of possible attack paths. Therefore, the total construction complexity of the AT is O(*m*!*n*!).

Lastly, we consider the construction of the HARMs, where the construction is independent in each layer. First, we consider the AG in both upper and lower layers (HAGAG). The construction complexity of the upper layer is given by $O(n^2)$, because for each host, there are up to *n*-1 connections for each host, giving *n*(*n*-1) total number of host connections. In the lower layer, the construction complexity is $O(m^2n)$ because for each vulnerability, there are up to *m*-1 number of other subsequent vulnerabilities, and this process is applied to *n* hosts. Therefore, the total construction complexity of the HAGAG is given by $O(m^2n+n^2)$. Secondly, we consider the AT in both upper and the lower layers (HATAT). The construction complexity of the upper layer is given by O(*n*!), because for *n* number of hosts, there are O(*n*!) number of possible attack paths. The lower layer construction complexity is given by O(*m*!*n*) because for *m* number of vulnerabilities, there are O(*m*!) number of possible exploits and this is repeated for each host. Therefore, the total construction complexity of the HATAT is O(*m*!*n*+*n*!). If we consider different attack models in each layer, the construction complexity of each attack model at each layer are combined together as the total construction complexity.

Evaluation

The evaluation complexity measures the total number of components used to calculate full attack paths. First we consider the evaluation complexity of the AG. To calculate full attack paths of the AG, we travel all possible paths from the attacker to the target. The number of paths between two hosts have complexity of O(*m*!), because there are *m*! number of possible paths of length *m*. We will consider the longest path length as the upper bound. The longest sequence is obtained by going through all hosts, meaning O(*m*!) is multiplied by itself *n*-1 times. This only specifies a single path in terms of network hosts, and there are O(*n*!) number of possible paths from the attacker to the target of length *n*. Therefore, the evaluation complexity of the AG is $O(m!^n n!)$.

Second, we consider the evaluation of the AT. The evaluation of AT is given by the total number of branches in the AT. From the construction, the AT have O(*m*!*n*!) number of branches. Therefore, the evaluation complexity of the AT is O(*m*!*n*!).

Lastly, we consider the evaluation of the HARMs. The evaluation of the HARMs is divided into each layer. First, we consider using HAGAG. The evaluation complexity of the upper layer is O(*n*!), similar to AG analysis. The lower layer evaluation complexity is O(*m*!*n*!), because there are O(*m*!) number of exploit sequences, and this is calculated for each attack path. Therefore, the total evaluation complexity using HAGAG is O(*m*!*n*!). We assumed that each host may have different sequences of exploits they can use for different host connections. If all hosts have the same sequences of exploits, then each host's vulnerability evaluation is a repeated event, changing the evaluation complexity to O(*m*!*n*+*n*!). If both *m* and *n* are large, then the overall evaluation complexity is reduced under this assumption. Secondly, we consider using HATAT. The number of states in the upper layer is O(*n*!), and the lower layer is O(*m*!). Therefore the total evaluation complexity using HATAT is O(*m*!*n*!).

Modification

An update event in the network system, such as addition, reconfiguration and deletion of host or vulnerability, causes modification. The complexity of modification measure the number of changes made to existing attack model for the update event. The modification complexity measures a single update event.

First, we consider the modification complexity of the AG. For vulnerability, there are up to *m* number of vulnerability connections between the component and other host. This is repeated to *n*-1 number of hosts. Therefore, the modification complexity of the AG is given by O(*mn*).

Second, we consider the modification complexity of the AT. An update event in the AT requires updated connections for each branch with the update component. There are O(m!(n-1)!) number of affected hosts, and O((m-1)!n!) number of affected vulnerabilities. Both calculations are approximated to O(m!n!), assuming both n and m are large, as the modification complexity of the AT.

Lastly, we consider the modification complexity of the HARMs. The modification complexity of the HARMs is divided by each layer. If we consider HAGAG, the modification complexity is O(n) and O(m) for the network layer and the vulnerability layer respectively. If we consider HATAT, the modification complexity is O(n!) and O(m!) for the network layer and the vulnerability layer respectively. Update events in each layer are independent in the HARMs.

COMPLEXITY COMPARISONS

The complexity analysis shows the efficiency of the HARMs compared with AG and AT in attack model phases. The complexity comparisons are shown in Table 1. Attack models have practical implications, and they can capture many security metrics for real networks (Albanese, Jajodia, & Noel, 2012; Roy, Kim & Trivedi, 2012; Ou, Boyer, & McQueen, 2006; Ingols, Chu, Lippmann, Webster, & Boyer, 2009). Here, our focus is to provide a scalable model for a large size network, and compare its complexities with existing attack models.

Tasks	Construction	Evaluation	Modification
AG	O(m^2n^2)	O($m!^n n!$)	O(mn)
AT	O($m!n!$)	O($m!n!$)	O($m!n!$)

HAGAG	O($m^2n+ n^2$)	O($m!n!$)	O(n) (Upper)
			O(m) (Lower)

HAGAT	O($m!n+ n^2$)	O($m!n!$)	O(n) (Upper)
			O($m!$) (Lower)

HATAG	O($m^2n!+ n!$)	O($m!n!$)	O($n!$) (Upper)
			O(m) (Lower)

HATAT	O($m!n!+ n!$)	O($m!n!$)	O($n!$) (Upper)
			O($m!$) (Lower)

Table 1: Attack model complexity comparisons

Construction Comparisons

Network properties are used to construct attack models. In existing attack models, the network topology information is passed onto vulnerability components to make connections. As a result, each vulnerability components in the AG makes independent connection to the network hosts. ATs also make independent connections depending on the construction approach. In the HARMs, the network topology information is not required at the lower layer. Hence, vulnerability components do not consider any network connection. From Table 1, the construction complexity between AG and HAGAG shows that HAGAG has lower construction complexity. The AT construction complexity remains the same because the construction method for the AT was based on the bottom-up approach, meaning the construction built the vulnerability information first, and then built the network information. If a top-down approach is used for the AT, the construction complexity will be equivalent to the evaluation complexity of the AG.

Evaluation Comparisons

The evaluation complexity of the AG is much greater than any other attack models. Logical simplification is illustrated using an example security analysis. We consider each exploit with its successful attack probability given in Table 2, using the example network system. The probabilities of each variable are reasonably and randomly chosen for a numerical security analysis. We will use real probability values from experiments in our future work, such as using testbeds, emulations, honeypots and data flow measurements (Sharma *et al.*, 2011; Mirkovic *et al.*, 2010; Alata *et al.*, 2006; Yan Zhu *et al.*, 2012). The probability of a successful attack is calculated by adding up all attack path probabilities. We have assumed that each attack path are equally likely be taken by the attacker. Hence, we divide by the total number of attack paths. The logical expression for calculating the successful attack using the AG is given in Equation 1. The total number of attack paths is the number of components in the equation joined by *OR* gates, or addition components. Equation 2 calculates the probability of a successful attack using the HAGAG, and we can verify that both results are equal. The total number of attack paths for the HAGAG can be calculated from the upper layer by substituting the number of components that each host contains. The number of states and calculations in Equation 2 is much less than Equation 1.

The AT components are linked by logical expressions using logical links, such as *AND* and *OR* gates. The sequence of attack is lost from the commutable property of logical expression, but we can express ATs as state space models. We expect the optimal size complexity will have a linear complexity in respect to the number of hosts and vulnerabilities. Hence, ATs will improve the evaluation phase to linear complexity.

State variable	Vulnerability	P_a(*attack success*)
F_1	ftp_rhosts$_1$	0.2
R_1	rsh$_1$	0.1
S_1	sshd_BoF$_1$	0.3
L_1	local_BoF$_1$	0.25
F_2	ftp_rhosts$_2$	0.2
R_2	rsh$_2$	0.1
L_2	local_BoF$_2$	0.25

Table 2: Probability of attack success for vulnerabilities

$$P_a(attack\ success) = \frac{\begin{pmatrix} F_1R_1F_2R_2L_2+F_1R_1R_2L_2+R_1F_2R_2L_2+ \\ R_1R_2L_2+S_1F_2R_2L_2+S_1R_2L_2+F_2R_2L_2+R_2L_2 \end{pmatrix}}{8} \quad (1)$$

$$\begin{aligned} P_a(attack\ success) &= \frac{P_a(User_1)P_a(User_2) + P_a(User_2)}{Paths(User_1)Paths(User_2) + Paths(User_2)} \\ &= \frac{(F_1R_1 + R_1 + S_1)(F_2R_2L_2 + R_2L_2) + (F_2R_2L_2 + R_2L_2)}{6+2} \end{aligned} \quad (2)$$

Modification Comparisons

All HARMs have better modification complexity compared to its equivalent attack models, as shown in Table 1. In existing attack models (Sheyner *et al.*, May, 2002; Schneier, 2000; Ou *et al.*, 2006; Ingols *et al.*, 2009; Xie *et al.*, 2009; Schneier, 1999; Edge *et al.*, 2007; Edge, 2007; Moore *et al.*, 2001; Dawkins & Hale, 2004; Saini *et al.*, 2008), each component in the attack model must make independent connections to other components. Hence, the update event affects a large proportion of the attack model. In contrast, the independent layered structure of the HARMs allows modifications to be performed on a single layer only, without affecting other layers.

Structural Advantages of the HARMs

Independent layers in the HARMs allow parallel construction of each layer. Using distributed systems, such as cloud, we can subdivide the HARMs by each layer, and join them without any complex process. Existing attack models can also be constructed in parallel, by dividing the model into sub-models, but dividing and joining processes require additional preprocessing on the attack model to find independent components.

We can also apply layer-centric analysis. For example, computing characteristics of the network system only requires network layer information (Ahn, Han, Kwak, Moon, & Jeong, 2007). Security analysis, such as impact of an attack, probability of successful attack, and network hardening optimizations, are considered by existing attack models. In addition, the HARMs can perform additional analysis applicable in different layers, such as performance analysis on the network layer. Therefore, we can consider wide range of constraints and other requirements when we analyse the network security.

CONCLUSION AND FUTURE WORK

Existing attack models have scalability and dynamic adjustment issues for large network systems. The HARMs are proposed to improve these problems. The HARMs divide independent components in the network systems into two layers. The HARMs are composed with other well known attack models (i.e., AGs, ATs) in the phase of construction, evaluation and modification. The HARMs show improvements in all phases compared with AG and AT. Further, structural advantages of the HARMs allow better construction performance, and layer-centric analysis to suite wider range of user and system requirements. The complexity comparisons verify that the HARMs perform better than existing attack models in all phases.

In the future work, we will construct analytical and simulation models with several network topologies to validate our complexity comparisons, and also taking into account dynamic changes (e.g., hosts join/removal, vulnerabilities patch) in network systems. Also, the practical use of the HARMs will be considered with security metrics and qualitative/quantitative security analysis as the work.

REFERENCES

Chew, E., Swanson, M., Stine, K., Bartol. N,. Brown, A., & Robinson, W. (2008) *Security metrics guide for information technology systems.* 800-55, N. S. P.

Ahn, Y., Han, S., Kwak, H., Moon, S., & Jeong, H. (2007). Analysis of topological characteristics of huge online social networking services. In *Proc. of International conference on world wide web (WWW 2007)* (pp. 835–844). New York, NY, USA: ACM. Retrieved from http://doi.ACM.org/10.1145/1242572.1242685 doi: 10.1145/1242572.1242685

Alata, E., Nicomette, V., Kaaniche, M., Dacier, M., & Herrb, M. (2006). Lessons learned from the deployment of a high-interaction honeypot. In *Proc. of European dependable computing conference (EDCC 2006)* (p. 39 - 46). doi: 0.1109/EDCC.2006.17

Albanese, M., Jajodia, S., & Noel, S. (2012). Time efficient and cost-effective network hardening using attack graphs. In *Proc. of IEEE/IFIP International Conference on Dependable Systems and Networks (DSN 2012).* doi: http://doi.IEEEcomputersociety.org/10.1109/DSN.2012.6263942

Dawkins, J., & Hale, J. (2004). A systematic approach to multi-stage network attack analysis. In *Proc. of IEEE international information assurance workshop (IWIA 2004)* (p. 48 - 56). doi: 10.1109/IWIA.2004.1288037

Edge, K. (2007). *A framework for analyzing and mitigating the vulnerabilities of complex systems via attack and protection trees.* Doctoral dissertation, Air Force Institute of Technology.

Edge, K., Raines, R., Grimaila, M., Baldwin, R., Bennington, R., & Reuter, C. (2007). The use of attack and protection trees to analyze security for an online banking system. In *Proc. of Hawaii international conference on system sciences (HICSS2007)* (pp. 144–151).

Hwang, K., & Gangadharan, M. (2001). Micro-firewalls for dynamic network security with distributed intrusion detection. In *Proc. of IEEE international symposium on network computing and applications (NCA 2001)* (pp. 68–79). doi: 10.1109/NCA.2001.962517

Ingols, K., Chu, M., Lippmann, R., Webster, S., & Boyer, S. (2009). Modeling modern network attacks and countermeasures using attack graphs. In *Proc. of annual computer security applications conference (ACSAC 2009)* (pp. 117–126). IEEE.

Ingols, K., Lippmann, R., & Piwowarski, K. (2006). Practical attack graph generation for network defense. In *Proc. of annual computer security applications conference (ACSAC 2006)* (p. 121 -130). doi: 10.1109/ACSAC.2006.39

Mirkovic, J., Benzel, T., Faber, T., Braden, R., Wroclawski, J., & Schwab, S. (2010). The deter project: Advancing the science of cyber security experimentation and test. In *Proc. of IEEE international conference on technologies for homeland security (HST 2010)* (p. 1 -7). doi: 10.1109/THS.2010.5655108

Moore, A., Ellison, R., & Linger, R. (2001). Attack modeling for information security and survivability. *CMU/SEI-2001-TN-001.*

Ou, X., Boyer, W., & McQueen, M. (2006). A scalable approach to attack graph generation. In *Proc. of ACM conference on computer and communications security (CCS 2006)* (pp. 336–345). ACM.

Saini, V., Duan, Q., & Paruchuri, V. (2008). Threat modeling using attack trees. *J. Comput. Sci. Coll., 23(4),* 124–131. Retrieved from http://dl.ACM.org/citation.cfm?id=1352079.1352100

Schneier, B. (1999). Modeling security threats. *Dr. Dobb's journal, 24(12).*

Schneier, B. (2000). *Secrets and lies: Digital security in a networked world.* John Wiley and Sons Inc., Indianapolis, Indiana

King, S. (2010). *Science of cybersecurity*. Retrieved from http://www.fas.org/irp/agency/dod/jason/cyber.pdf

Sharma, A., Kalbarczyk, Z., Barlow, J., & Iyer, R. (2011). Analysis of security data from a large computing organization. In *Proc. of IEEE/IFIP International Conference on dependable systems networks (DSN 2011)* (p. 506 -517). doi: 10.1109/DSN.2011.5958263

Sheyner, O., Haines, J., Jha, S., & Wing, R. L. J. (2002). *Automated generation and analysis of attack graphs* (Tech. Rep.). CMU.

Theodorakopoulos, G., Baras, J., & Le Boudec, J. (2008). Dynamic network security deployment under partial information. In *Proc. of annual allerton conference on communication, control, and computing (CCC 2008)* (pp. 261–267). doi: 10.1109/ALLERTON.2008.4797565

Xie, A., Cai, Z., Tang, C., Hu, J., & Chen, Z. (2009). Evaluating network security with two-layer attack graphs. In *Proc. of annual computer security applications conference (ACSAC 2009)* (p. 127 -136). doi: 10.1109/ACSAC.2009.22

Zhu, Y., Hu, H., Ahn, G., Huang, D., & Wang, S. (2012). Towards temporal access control in cloud computing. In *Proc. of annual IEEE international conference on computer communications (INFOCOM 2012)* (p. 2576-2580). doi: 10.1109/INFCOM.2012.6195656

Roy, A., Kim, D. S., Trivedi, K. (2012). Scalable optimal countermeasure selection using implicit enumeration on attack countermeasure trees. In *Proc. of IEEE/IFIP International Conference on Dependable Systems and Networks (DNS 2012)* doi: http://doi.ieeecomputersociety.org/10.1109/DSN.2012.6263940