Unified Parametrizable Attack Tree

Jie Wang, John N. Whitley, Raphael C.-W. Phan and David J. Parish
High Speed Networks (HSN) Research Group
Electronic and Electrical Engineering
Loughborough University
LE11 3TU, UK

Abstract

Network attacks occur in high proportion on the internet, thus aside from security as a means of defense there is a need for being able to detect attacks as they occur so that measures can be put in place to tackle them. In this paper, we focus on attack trees as a tool to model attacks thereby facilitating attack detection. More precisely, we present a unified parametrizable attack tree; this can be applied for intrusion detection and can be instantiated to produce particular types of attack trees in literature.

1. Introduction

Present day computer networks are constantly under attack (attempts); for instance it was reported even back in 2008 [3] that attack traffic accounts for 20% of America's overall internet traffic, while recently it was revealed [23] that spam accounts for 90.4% of all email traffic. Indeed, the existence of non-stop attack attempts on networks is now a fact of life, the main issue is how to cope with this fact and be resilient in face of all these.

We focus on extending the notion of attack trees proposed by Schneier in 1999 [16]. Attack trees can be used to model potential attacks on a system and corresponding risks associated with each attack path.

Despite the widespread application of the attack tree modelling technique to different kinds of systems, what is lacking is the more foundational treatment of the attack tree as an analysis technique in itself; one nice fundamental work in this direction is [13] and subsequent results that extend it.

In our preliminary previous work [22, 21], we applied augmented attack trees [14] to model different network attacks and presented intrusion detection algorithms that utilize such trees. We also extended, in [20], the notion of attack trees to incorporate the capability to offer quality of detectability (QoD) for intrusion detection schemes.

In this paper, we continue in our direction of attack tree research; we present a unified treatment to the attack tree by casting it into a generic setting parametrized by three parameters: *node attribute*, *edge augmentation* and *connector type*. Indeed, the node, edge and connector are the basic elements of an attack tree, yet they can be defined in different flavours with varying levels of detail. To the best of our knowledge, no previous work has treated the attack tree within a unified and parametrizable viewpoint.

2. Related Work

For ease of description, we shall henceforth refer to the original attack tree [16] as the Conventional Attack Tree (CAT). CAT is defined to model the threats against computer network systems. An alternative formalism [13] investigates the foundation of attack tree with the denotational semantics according to the attack suite mapping information.

As one of the critical security analysis and modelling tools, CAT had been widely applied as the prototype of security related researches in order to provide flexility and adaptability with more precise information. Generally, these extensions can be classified into three main categories: (1) *structure extension*; (2) *computation extension*; and (3) *hybrid*. The main purpose of the first category is to enlarge the modelling capability and augment the modelling functionality with extra information for CAT. The key goal of the second category is to compute and measure the relevant attack decision making from the view of attacker. In the last category, CAT had been extended with other modelling methods together as a hybrid method.

2.1 Attack Tree Structure Based Extension

Since CAT concentrates on the modelling of the nature of attacks and the vulnerability of systems, there are several extensions specialized to provide corresponding countermeasure actions of attack into the modelled tree structure in order to protect the system. Defence Tree [2] provides a set of attack countermeasures on each leaf node. The provided countermeasures represent the possible threat mitigation of the specific vulnerability scenario. Attack Countermeasure Tree (ACT) [15] provides the similar attack countermeasure mechanism as Defence Tree does, but ACT offers countermeasure at any of tree nodes instead of leaf nodes. There are three distinct types of nodes in ACT: attack event node, detection event node and mitigation event node. Attack-Response Tree (ART) [25] defines and analyses the possible vulnerabilities to compromise a system and the possible response actions against attacks. In ART, every leaf node represents a specific vulnerability exploitation attempt by the attacker. While the root node represents the security property. The consequence nodes in ART are tagged by the response box that represent countermeasure actions. Attack-Defense Tree (ADTree) [12] describes the attack actions an attacker can take to compromise the system and the defense actions that a defender can employ to protect the system. The node in ADTree classifies into the attack node and the defense node. Each node may also have one additional child node representing a countermeasure. The node of Protection Tree (PT) [8] contains four metrics: probability, cost, impact and risk. With the run-time obtained metrics value, PT ensures the limited resources are consumed to achieve the highest probability to stop an attack successfully.

Except the extensions with the appropriate defense and countermeasure mechanisms, couple of works extend the original elements of CAT with sophisticated modelling capability. OWA trees [24] provides a class of aggregation operators called Ordered Weighted Averaging (OWA) decomposition to model the sophisticated relation between a parent node and its child node(s). Augmented Attack Tree (AAT) [14] contains the attack signature of the corresponding atomic attack in a multi-step attack procedure. In previous work, we proposed an AAT-based intrusion detection algorithm in [21] to detect network attacks. We also proposed in [20] to extend attack trees with attack detection quality (so-called QoD) metrics.

2.2 Attack Tree Computation Based Extension

The attack tree computation based extension is a substantial progress in the attack tree research field. Several extensions have been proposed to compute attacker's decision making process from the view of attacker. The multi-parameter attack tree computation [5] has introduced the idea of game-theoretic modelling associated with different elementary attacks to estimate the cost and the success probability of attacks. The serial attack tree model [10] extends the classic parallel attack tree model with the temporal order of the elementary attacks. The main advantage is to provide the flexibility to model the attacker's behaviour more accurately and reality, i.e. the elementary attacks skipping. In addition, it computes the better expected outcomes of the attacker.

2.3 Hybrid: Attack Tree et al.

ATiki is the collaborative method by applying CAT and Petri Net [17]. It utilizes two major Petri Net elements *condition* and *transition* into the structure of CAT. The *condition* is utilized as the node, whereas the *transition* is utilized as the edge. The advantages of this hybrid method are that it allows pieces of information to be sorted in and combines easily.

Misuse cases, derived from UML use cases, describe the actions that may harm the system. In [18], misuse cases were applied with attack tree to model the security activity and analysis the system security requirements in software development lifecycle of a software project.

3. Advanced Attack Tree Parameters

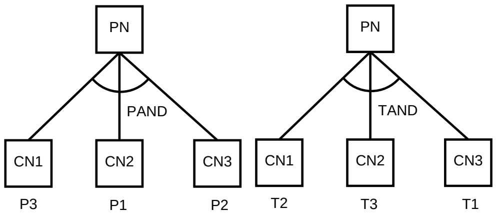
In this section, we discuss the different advanced ways in which an attack tree's parameters i.e. connector type and node attribute, have been extended in literature.

3.1 Advanced Node Connectors

3.1.1 Order Based Connector

Priority Based Order Connector (PBOC) represents the parent node can be achieved only if all the child nodes are accomplished in a priority order, from the highest priority one to the lowest priority one. Priority-AND (PAND) [4, 11] obeys this kind of order. Note that, it is possible to assign multi nodes with exactly same priority. In this kind of situation, those nodes cannot be achieved simultaneously. Instead, they should be accomplished in term. Therefore, we apply the first come, first achieve mechanism on those equal-priority nodes. Figure 1(a) shows a parent node PN (attack goal) that can be achieved if the attacker achieves each of child nodes (sub-goals) CN1, CN2 and CN3 with the priority order. According to the priority sequence P1 → P2 → P3, the child nodes are accomplished with the CN2 → CN3 → CN1 sequence.

Time Based Order Connector (TBOC) represents the parent node can be achieved only if all the child nodes are accomplished in a pre-defined time sequence manner. Ordered-AND (O-AND) [6] and Sequence Enforcing (SEQ) [11] applied this kind of time based order. Figure 1(b) illustrates a parent node PN (attack goal) that can be achieved if the attacker achieves each of child nodes (sub-goals) CN1, CN2 and CN3 with the defined time order. According to the time sequence T1 → T2 → T3, the child nodes are accomplished with the CN3 → CN1 → CN2 sequence.



(a) PBOC (b) TBOC

Figure 1. Conditional Certainty Connectors

3.1.2 Threshold Based Connector

Threshold Based Connector (TBC) determines the minimum number of accomplished child nodes for success of the parent node, e.g., K-out-of-N (K/N) [11]. TBC is an intermediate connector which describes the uncertainty with the needed number of child nodes. Normally, it only concentrates on the number of node satisfaction, but ignores to determine which node need to achieve. Note that TBC has two special cases. TBC equivalents to OR decomposition if the determined minimum number is one. In addition, TBC equivalents to AND decomposition if the determined minimum number equals to the number of child nodes.

Figure 2(a) shows the parent node can be achieved if the attacker achieves at least 2 child nodes out of 3 child nodes. In this model, there are three child nodes and the threshold value is set as 2. Therefore, either two child nodes achievement accomplish the parent node. The truth table of TC with 2-out-of-3 is given as Table 1 in columns CN1, CN2, CN3 and PN(TBC).

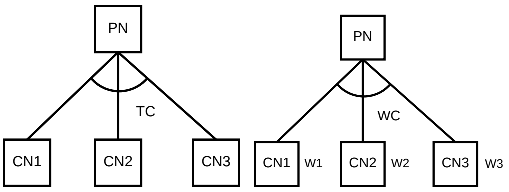
3.1.3 Weighted Based Connector

Weighted Based Connector (WC) determines the probabilistic uncertainty as to the number of child nodes that need be satisfied for the parent node accomplishment. For this kind of weight based connector, weight components W_i, where i is the index of child nodes, must satisfy the following two conditions: (1) $0 \leq W_i \leq 1$ and $(2)\sum_{i=1}^{n} W_i=1$, where n is the total number of child nodes.

As each child node assigned one weight individually, the number of child node uncertainty is determined by accumulating the achieved child nodes' weight together. Once the accumulated value exceeds the threshold, the number of child nodes is determined. Figure 2(b) illustrates the attack tree model for FWC. Each node has assigned with their own weight, W1 of CN1 is 0.2, W2 of CN2 is 0.3 and W3 of CN3 is 0.5. The threshold value is set as 0.6. The columns CN1, CN2, CN3 and PN(WBC) in Table 1 shows the truth values of this scenario.

Table 1. Truth Table For UUC

CN1	CN2	CN3	PN(FBC)	PN(WBC)
F	F	F	F	F
F	F	T	F	F
F	T	F	F	F
T	F	F	F	F
F	T	T	T	T
T	F	T	T	T
T	T	F	T	F
T	T	T	T	T



(a) TBC (b) WBC

Figure 2. Unconditional Uncertainty Connectors

3.2 Advanced Node Attributes

Attack trees can not only capture the steps of an attack and their interdependencies, but also used to represent and calculate probabilities, risks, cost or weighting [17]. In the conventional attack tree, Schneier [16] had defined that attributes can be assigned with either boolean or continuous values. The raw value of those attributes are assigned at the leaf nodes. The attributes of upper nodes are obtained through the propagation of pre-defined mathematical calculation. With the assistance of attributes, attack tree provides the ability to describe the full complexity of the attacker's decision-making process. Besides those original node attributes, there are several extra node attributes applied in the attack tree modeling. In the terms of usage, we classify those attributes into two main categories, *the attack attribute* and *the victim system attribute*. Meanwhile, we subdivide the attack attribute attributes into two sub-categories, *the attack accomplishment attribute* and *the attack evalua-*

tion attribute.

3.2.1 Attack Accomplishment Attributes

Attack Accomplishment Attributes (AAA) examine the atomic attack procedure information of current node. It describes the temporal dependencies between components and expiration of an attack, even the attack success probability.

Time-To-Live (TTL) [6] defines the lifetime for attack actions at nodes of the attack tree. With the regulation of TTL, the attacker need to finalize the attack within TTL. If more than TTL time has elapsed since the node accomplished, it must be expired.

Attack Level (AL) and *Attack Probability* (AP) attributes [6] define the amount or the level of attack completed when a node is accomplished. As attack tree usually models the multi-step attack process, AL and AP assistant the measurement of the distance between the current node and the ultimate root node by examining the latest accomplished node. AL calculation for a node assumes each attack action with equal difficulty and weight, whereas AP calculation for a node defines each attack action with different difficulty and weight according to the statistical analysis. Through the statistical analysis, the actions generated more often are assumed as easier to be accomplished. Both AL and AP can be used to establish an early warning system.

Attack Success Probability (ASP) [5, 8, 9] indicates the probability of attack to achieve one particular node.

3.2.2 Attack Evaluation Attributes

Attack Evaluation Attributes (AEA) are associated with each of the nodes in the attack tree to analyse and evaluate attacks.

Cost to Attack (C) [5, 8, 9, 19] indicates the money cost of man-hour to achieve the subgoal during the whole attack procedure.

Impact to the System (I) [8] indicates how serious of damage being given to the system.

Risk (R) [8] is calculated by using a logarithmic scale to easily compare the risk values between nodes. In addition, the exploitability, the dependency and the damage potential [1] been utilized to assess the risk. The exploitability, which is associated to each vulnerability to measure the likelihood that the vulnerability may be successfully used to the security of the system. The dependency is identified between the enabling vulnerabilities of the former and the latter attacks. The damage potential measures the ability to damage as the number of the affected users times the average number of days the affected service is unavailable.

Four extra attributes [5, 9] used in attack tree to evaluate the attack. Two probabilities of catching the attacker, if the attack was successful q_S, and, if the attack was unsuccessful q_F. In addition, two penalties of the attacker, if the attack was successful and attacker was caught k_S, and, if the attack was not successful and attacker was caught k_F. With the assistance of those node attributes, a much more accurate attack tree model can be obtained.

3.2.3 Victim System Attributes

Victim System Attributes describe the attack information of the attack tree node from the view of victim system. Because different properties of the network system effectuate different ways for an attacker to compromise a victim [7]. Several network properties such as System Vulnerability (SV), Network Configuration (NC), System Configuration (SC) and Access Privilege (AP) had been applied as the attack tree node attributes. SVs are already reported vulnerabilities from some well known security database. NC is the related information about open port, unsafe firewall configuration, etc. SC such as data accessibility, unsafe default configuration, or read-write permission in file structures. AP includes user account, guest account and root account. With the assistance of those metrics, attack potential damage can be evaluated and can be assigned into node as one of cost metrics.

4. Unified Parametrizable Attack Tree

In this section, we present the unification of different ways in which parameters of an attack tree may be extended, notably: *node attribute*, *edge augmentation*, and *connector type*. In formally describing the elements of the tree, an overview of the relationship between the component elements is helpful.

The purpose of the attack tree elements, and the entities the attack tree refers to, can be equated to English grammatical elements within a sentence: the *subject–noun* is the adversarial entity, and is not directly represented within the attack tree; the *object–noun determiner* pair, or *determined–object*, is the system element that is, or could be, under attack, and is not directly represented within the attack tree; the node of the tree represents the coupling of an *adjective* with the determined–object, describing the state of an attack; attributes of a node are *prepositional phrases* further describing the relationship of the determined–object–adjective (node); edges of the tree represent the *verbs* that explain the process that is needed to move from one determined–object adjective state to another; augmentations of a verb (edge) are *adverbs* and *adverbial phrases*; and the connector type is the logical construction of verbs built up using *conjunctions* to allow multiple determined–object adjective verb combinations to be used within a sentence.

An example scenario could be: the vulnerability is defined as an adversary *(subject–noun)* invading *(adjective)* the room *(determined–object)* and being left alone for at least five minutes *(prepositional phrases)*. Currently the adversary is not present in the room, but is at the end of the corridor. To invade the room, the adversary must either: walk *(verb)* down the corridor, where walking may be qualified by speed *(adverb)* and direction *(adverbial phrase)*; or *(conjunction)* break in *(verb)* through the window.

4.1 Parameters

We will first define the input parameters to the unified attack tree.

Definition 1. $EdgeParam$ is assigned a boolean value, i.e. $EdgeParam \in \{0, 1\}$ to determine whether to augment the edge within the attack tree. $EdgeParam = 0$ denotes that the edge is devoid of augmentation, while $EdgeParam = 1$ denotes that the edge is augmented with the tuples $\langle Label, SIG_{u,v} \rangle$ as per Definition 4

Definition 2. $ConnectParam$ is assigned with an integer value, i.e. $ConnectParam \in \{1, 2, \ldots, p\}$ where: each integer denotes an index to a particular connector type, and p is the maximum index number of connectors to select a particular connector on one node.

Definition 3. $NodeAttrParam$ is assigned with the tuples $\langle q, Attr_{ID} \rangle$, where: q denotes the number of desired node attributes and $Attr_{ID}$ denotes the index of a particular attribute type.

4.2 Formalization

UPAT uses similar notations as AAT. The formalization of UPAT is as follows:

Definition 4. A unified parametrizable attack tree (UPAT) is a rooted labelled tree given by $UPAT = \langle N, E, C, A, G \rangle$, where

- *N is a finite set of nodes in the tree representing the different states of partial compromise or sub-goals that an attacker needs to move through in order to fully compromise a system. $\nu \in N$ is a special node, distinguished from others, that forms the root of the tree. It represents the ultimate goal of the attacker, namely system compromise. The set N can be partitioned into two subsets, leaf_nodes and internal_nodes, such that*
  - $*$ *leaf_nodes $\bigcup$ internal_nodes=N,*
  - $*$ *leaf_nodes $\bigcap$ internal_nodes=$\emptyset$,*
  - $*$ *$\nu \in$ internal_nodes.*
- *$E \subseteq N \times N$ constitutes the set of edges in the attack detection tree. An **edge** $\langle u, v \rangle \in E$ defines an **atomic attack** and represents the state transition from a child node v to a parent node u, for $u, v \in N$. An atomic attack is a sequence of **incidents**. The edge $\langle u, v \rangle$ is said to be emergent from v and proceeding to u.*
- *C is a set of node connector tuples of the form $\langle v, Decomposition \rangle$ such that*
  - $*$ *$v \in$ internal_nodes and*
  - $*$ *Decomposition $\in$ {O-AND, U-AND, OR}, where O-AND, U-AND and OR are given in Definitions 5, 6 and 7.*

  The particular decomposition choice of one node is based on the input parameter $ConnectParam$, which is defined as Definition 2.
- *A is a set of node attributes tuples of the form $\langle v, Attr \rangle$, Attr can be partitioned into two subsets: QoD_attr, which represents the real-time measured Quality of Detectability (QoD) metrics of the attack detector [20]; and attack_attr, which represents the characteristics of the modelled attack state, such that*
  - $*$ *$v \in N$ and*
  - $*$ *Attr $\in$ {QoD_attr, attack_attr}, where*
    - ◇ *QoD_attr $\bigcup$ attack_attrs=Attr,*
    - ◇ *QoD_attr $\bigcap$ attack_attr=$\emptyset$.*

  The particular attribute choice is based on the input parameter $NodeAttrParam$, which is defined as Definition 3.
- *G is a set of edge augmentation structures of the form $\langle u, v, Label, SIG_{u,v} \rangle$, determined based on the input parameter $EdgeParam$, which is defined as Definition 1. For $EdgeParam = 0$, G is $\emptyset$. With an $EdgeParam$ being equal to 1, the edge $\langle u, v \rangle$ becomes analogous to an attack signature, and we have $Label$ denotes the name of the exploit associated with the edge and $SIG_{u,v}$ represents the attack signature defined as Definition 9.*

Definition 5. Given a node v of a unified parametrizable attack tree such that $v \in internal\_nodes$, the node is an O-AND-decomposition if all edges incident to the node are connected by the AND operation but with either the priority based order or the time based order.

Definition 6. Given a node v of a unified parametrizable attack tree such that $v \in internal\_nodes$, the node is an U-AND-decomposition if all edges incident to the node are connected by the AND operation but contain uncertainty with either the threshold condition or the weight condition.

Definition 7. Given a node v of a unified parametrizable attack tree such that $v \in internal\_nodes$, the node is an OR-decomposition if all edges incident to the node are connected by the OR operation.

Definition 8. An incident-choice is a group of related incidents, the occurrence of any one of which can contribute towards the state transition in the attack tree.

Definition 9. An attack signature $SIG_{u,v}$, is associated with an edge $\langle u, v\rangle$ and is a sequence of incident-choices $(incident\_choice_1, incident\_choice_2, \ldots, incident\_choice_n)$ such that the sequence constitutes an atomic attack that contributes towards the state transition in the attack tree.

4.3 Applying UPAT

UPAT can be applied for intrusion detection. For instance, consider the intrusion detection algorithm based on AAT in [21]. Although the AAT has extra edge augmentation compared to conventional attack trees, there is less richness to be represented by such trees e.g. only AND and OR connector types are considered, and there are no attributes. Using UPAT instead allows advanced connectors to provide the flexibility to model more complicated attacks, instead of simply AND-decomposition and OR-decomposition, and additional node attribute information can identify related attack information and provide the related risk warning information to the intrusion detection system. In another direction, one can see that UPAT can be instantiated to produce particular restricted attack trees, for instance by instantiating the input parameter $ConnectParam$ so that there are only two connector types AND and OR, $EdgeParam = 1$ and $nodeAttrParam = \langle 0, \perp\rangle$, we obtain the augmented attack tree (AAT).

References

[1] M. Benini and S. Sicari. Risk assessment in practice: a real case study. *Journal of Computer Communications*, 31(15):3691–3699, 2008.

[2] S. Bistarelli, F. Fioravanti, and P. Peretti. Defense trees for economic evaluation of security investments. In *Proceedings of The First International Conference on Availability, Reliability and Security*, April 2006.

[3] K. Blodget. China leads in global Internet attack traffic. In *Akamai Reports*, 17 November 2008.

[4] P. Brooke and R. Paige. Fault trees for security system design and analysis. *Journal of Computers & Security*, 22(3):256–264, 2003.

[5] A. Buldas, P. Laud, J. Priisalu, M. Saarepera, and J. Willemson. Rational choice of security measures via multiparameter attack trees. *Journal of Critical Information Infrastructures Security*, 4347:235–248, 2006.

[6] S. Camtepe and B. Yener. Modeling and detection of complex attacks. In *Proceedings of Third International Conference on Security and Privacy in Communications Networks and the Workshops*, pages 234–243, September 2007.

[7] R. Dewri, N. Poolsappasit, I. Ray, and D. Whitley. Optimal security hardening using multi-objective optimization on attack tree models of networks. In *Proceedings of the 14th ACM Conference on Computer and communications Security*, pages 204–213, 2007.

[8] 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 *Proceedings of 40th Annual Hawaii International Conference on System Sciences*, pages 144b–144b, January 2007.

[9] A. Jürgenson and J. Willemson. Processing multi-Parameter attacktrees with estimated parameter values. *Advances in Information and Computer Security, Second International Workshop on Security*, 4752:308–319, 2007.

[10] A. Jürgenson and J. Willemson. Serial model for attack tree computations. *Information, Security and Cryptology-ICISC 2009*, 5984:118–128, 2010.

[11] P. Khand. System level security modeling using attack trees. In *Proceedings of 2nd International Conference on Computer, Control and Communication*, pages 1–6, February 2009.

[12] B. Kordy, S. Mauw, S. Radomirović, and P. Schweitzer. Foundations of attack-defense trees. *LNCS FAST*, 2010.

[13] S. Mauw and M. Oostdijk. Foundations of attack trees. *Journal of Information Security and Cryptology-ICISC 2005*, 3935:186–198, 2006.

[14] N. Poolsapassit and I. Ray. Investigating computer attacks using attack trees. *IFIP, Advances in Digital Forensics III*, 242:331–343, 2007.

[15] 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*, pages 1–4, 2010.

[16] B. Schneier. Attack trees. *Dr. Dobb's Journal*, 24(12):21–29, 1999.

[17] J. Steffan and M. Schumacher. Collaborative attack modeling. In *Proceedings of the 2002 ACM Symposium on Applied Computing*, pages 253–259, 2002.

[18] I. Tøndel, J. Jensen, and L. Røstad. Combining misuse cases with attack trees and security activity models. In *Proceedings of International Conference on Availability, Reliability, and Security*, pages 438–445, Februray 2010.

[19] H. Wang, S. Liu, and X. Zhang. A prediction model of insider threat based on multi-agent. In *Proceedings of 1st International Symposium on Pervasive Computing and Applications*, pages 273–278, August 2006.

[20] J. Wang, R. Phan, J. Whitley, and D. Parish. Quality of Detectability (QoD) and QoD-aware AAT-based Attack Detection. In *Proceedings of 2010 International Conference for Internet Technology and Secured Transactions*, pages 152–157, November 2010.

[21] J. Wang, R.-W. Phan, J. Whitley, and D. Parish. Augmented attack tree modeling of distributed denial of services and tree based attack detection method. In *Proceedings of IEEE 10th International Conference on Computer and Information Technology*, pages 1009–1014, June 2010.

[22] J. Wang, R.-W. Phan, J. Whitley, and D. Parish. Augmented attack tree modeling of SQL injection attacks. In *Prceedings of The 2nd IEEE International Conference on Information Management and Engineering*, pages 182–186, April 2010.

[23] L. Whitney. Report: Spam now 90% of All E-mail. In *CNET News*, 26 May 2009.

[24] R. Yager. OWA trees and their role in security modeling Using attack trees. *Information Sciences*, 176(20):2933–2959, 2006.

[25] S. Zonouz, H. Khurana, W. Sanders, and T. Yardley. RRE: a game-theoretic intrusion response and recovery engine. In *Proceedings of IEEE/IFIP International Conference on Dependable Systems & Networks*, pages 439–448, 2009.