Using Attack-Defense Trees to Analyze Threats and Countermeasures in an ATM: A Case Study

Marlon Fraile, Margaret Ford, Olga Gadyatskaya, Rajesh Kumar, Mariëlle Stoelinga, Rolando Trujillo-Rasua

Using Attack-Defense Trees to Analyze Threats and Countermeasures in an ATM: A Case Study*

Marlon Fraile[1], Margaret Ford[2], Olga Gadyatskaya[3], Rajesh Kumar[4], Mariëlle Stoelinga[4], and Rolando Trujillo-Rasua[3]

[1] GMV, Spain
[2] Consult Hyperion, UK
[3] University of Luxembourg, SnT, Luxembourg
[4] University of Twente, Netherlands

Abstract. Securing automated teller machines (ATMs), as critical and complex infrastructure, requires a precise understanding of the associated threats. This paper reports on the application of attack-defense trees to model and analyze the security of ATMs. We capture the most dangerous multi-stage attack scenarios applicable to ATM structures, and establish a practical experience report, where we reflect on the process of modeling ATM threats via attack-defense trees. In particular, we share our insights into the benefits and drawbacks of attack-defense tree modeling, as well as best practices and lessons learned.

Key words: attack-defense trees, security modeling, ATM security

1 Introduction

Worldwide, the compromise of automated teller machines (ATMs) is a very lucrative criminal business. One of the prime reasons is the monetary incentive, allowing successful attackers to take money instantly. Moreover, their geographical spread, dependence on human interactions, and integration of local and external networks make ATMs a very accessible target for exploitation, vulnerable to a large variety of attack scenarios. Thus, criminals constantly invent new ways to circumvent protections and compromise the machines. The European ATM Crime Report (EAST)[2] evaluates the loss in 2015 due to ATM Related Fraud Attacks in Europe was around 300 millions Euro.

The security of individual ATMs concerns both banks and the organizations hosting the machines. In this context, security risk management, being a critical activity for any enterprise, becomes essential. To support risk analysts, many methodologies have been developed. These include security methods, such as NIST SP800-30, standards for the risk management process (e.g. ISO/IEC

* The research leading to the results presented in this work received funding from the European Commission's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 318003 (TREsPASS).

[2] `https://www.european-atm-security.eu/tag/european-atm-crime-report/`

27005), and modeling techniques and formalisms (for example, misuse cases [13], anti-goal refinement [10], and attack trees [18]). These methodologies aim at providing structure to the risk assessment process, facilitating interactions among stakeholders, and cataloguing the identified threats. Furthermore, some of these techniques enable advanced quantitative risk analysis with security metrics, e.g. expected time of attack or worst case impact.

In this paper, we report on the application of *attack-defense trees* to security risk assessment of ATMs. Attack-defense trees (ADTrees, [7]) extend the popular attack trees formalism with defenses (also called countermeasures). Similarly to attack trees, ADTrees enjoy an appealing and intuitive visual representation, a structured way to brainstorm about attack scenarios [8] and formal frameworks to analyze the trees qualitatively or quantitatively [11, 7]. Additionally, ADTrees support reasoning about potential defenses, enabling highly effective decision-making processes for countermeasure selection. Since defenses are crucial in the case of ATMs, the ADTrees formalism provided valuable support for our case.

Our paper presents a practical experience report, where we reflect on the process of modeling ATM security threats, and potential countermeasures via ADTrees. The paper outlines the case study, and describes our process for designing a large, comprehensive ADTree. We also share techniques that we found useful when working with ADTrees and report caveats that the practitioners need to become aware of.

2 Background and preliminaries

Attack trees. Attack trees (ATrees, [18, 11]) are a graphical formalism to structure, model and analyze the potential attacks on an asset. An attack tree starts with a security threat, modelled as the root of the tree, representing the attacker's top level goal. This root is recursively refined into the attacker's subgoals through *logical gates*, modelling how successful attack steps propagate through the system. AND-gates model that, to succeed in this step, the attacker must succeed in all of its child nodes; OR-gates model that, to succeed in this step, the attacker must succeed in at least one of its child nodes. When further refinement is not possible or not required, one arrives at the *basic attack steps (BASs)*, sitting at the leaves of the ATree. Leaves can further be decorated with quantitative attributes, such as cost or success probability of the BASs [11, 9].

Attack-defense trees. Attack-defense trees extend attack trees with defensive measures, also called *countermeasures*, yielding a graphical mathematical model of multi-stage attacks along with safeguards [7]. Defense nodes can appear at any level of the tree, and can be further refined with AND- and OR-gates. Moreover, countermeasures can themselves be attacked. Thus, each node belongs either to the attacker (represented as red ellipses in our figures) or defender (green squares our figures). Countermeasures prevent an adversary from reaching the goal, thus an ADTree represents an interplay between an attacker, whose goal is to attack the system, and a defender who tries to protect it [7].

Related work. Several papers report on the applicability of attack trees in practical scenarios. Opdahl and Sindre have empirically compared ATrees with misuse cases and reported that participants were able to identify more threats with ATrees [13]. Saini et al. evaluated security of the MyProxy system with ATrees [17], Byres et al. used attack trees to evaluate the SCADA communication systems security [2], and Ray and Poolsapassit applied the ATree methodology to identify insider threats [15]. Security of vehicular ad-hoc networks was evaluated with ATrees in [3]. In [4], Edge et al. modeled an online banking scenario via deriving protection trees from ATrees. Following the approach of [6], a methodology to construct ATrees based on the system architecture, risk assessment study outcomes and a security knowledge base is proposed in [14]. This methodology follows a layered approach to generate skeletons of attack trees. Morais et al. [12] follow a similar methodology but in a top-down manner, when first high-level attacks are collated, and then these are refined into concrete attack steps. None of these approaches, however, included defenses.

To the best of our knowledge, the ADTree formal language has been empirically validated only once; through a case study on a warehouse goods management system developed by Bagnato et al. [1]. That work focused mainly on the quantitative aspects of the ADTree methodology. In this article, instead, we focus on modeling aspects of the ADTree methodology.

3 The ATM case study

Establishing the context. This case study considers an ATM located inside a gas station, which is split in two main zones: the **store zone**, enabled with a security glass door class RC3 and two security glass windows class RC2[3], and the **internal office**, where the technological components related to the gas station management (workstation, printer, router and local Internet connection used to share information with the headquarters) are located. Customers can transit the store zone to buy or request services, including ATM services, during the business hours of the store. The gas station is open from 6:00 AM to midnight and provides several services including: fuel, car-wash, food, cash, etc.

Identification of interested parties. The gas station involves different interested parties that may perform several roles. Physical security is outsourced to a *Security Provider* who has physical countermeasures implemented in the gas station. These include a fire alarm, video surveillance enabled with external cameras, and burglar alarm enabled with several kinds of sensors (window/door vibration and movement detectors) and anti-jamming features. *Insurance provider* lets ATM owners insure their assets in case of any incident based on several scenarios and configurations. The ATM per se is an asset and the investment in each unit can vary widely depending on the brand, model, configuration, etc. *Bank* is an

[3] `http://www.din.de/en/getting-involved/standards-committees/nabau/projects/wdc-proj:din21:208343022`

organisation that manages a range of financial services, including ATM transactions from its own ATMs or from other ATMs as issuer/acquirer. *Customers* are users who use the gas station's services, including the ATM. *Attacker* is an interested party responsible for exploiting a vulnerability with the objective of achieving an illegal goal. Finally, *Insider* is an employee who could potentially provide information or physical access (voluntarily or not) to attackers.

4 The attack-defense tree model

Developing ATree models for complex systems has been traditionally a cumbersome task requiring a team of experts. The first step towards modeling is to understand the system and the context by identifying stakeholders, system components, and attackers. Another important aspect is to grasp the semantics of the ADTree modelling language. We covered both aspects by building a team of four security experts, two from industry and two from academia. One industry expert from our team has experience in security and financial services; he played the domain expert role. The second industry expert has expertise in security assessment, financial services, and has prior experience with ATrees. She played the role of validator and was responsible for quality evaluation. The other team members have extensive knowledge of semantics and analysis techniques for ADTrees. They were responsible for structuring the tree. In the rest of this section we explain the process followed by our team to design a comprehensive ADTree model for the ATM scenario. We used the open source ADTool software for designing the tree [5].

Overcoming the lack of attack intelligence. The task of mapping a security scenario into an ADtree greatly depends on the security expertise of the team developing the tree. However, security expertise needs to be complemented with data about previous attacks. Such data can come in the form of an attack pattern library[4], i.e. a structure containing precondition and postcondition of attacks, attack profiles, and a glossary of defined terms and phrases. Yet, businesses and governments are usually reluctant to disclose attack data, as it may harm their reputation and could help attackers to exploit similar vulnerabilities.

For this case study, the financial services specialist overcame the lack of attack data by using different sources of information on ATM security, such as *PCI-DSS ATM Security Guidelines*[5] to understand how secure channels for payment systems based on smartcards are implemented, *ATM Industry Association*[6] to collect best practices in ATM security, *EU law enforcement agency*[7] to get recent

[4] `www.sei.cmu.edu/reports/01tn001.pdf`
[5] `https://www.pcisecuritystandards.org/pci_security/`
[6] `https://www.atmia.com/`
[7] `https://www.europol.europa.eu/`

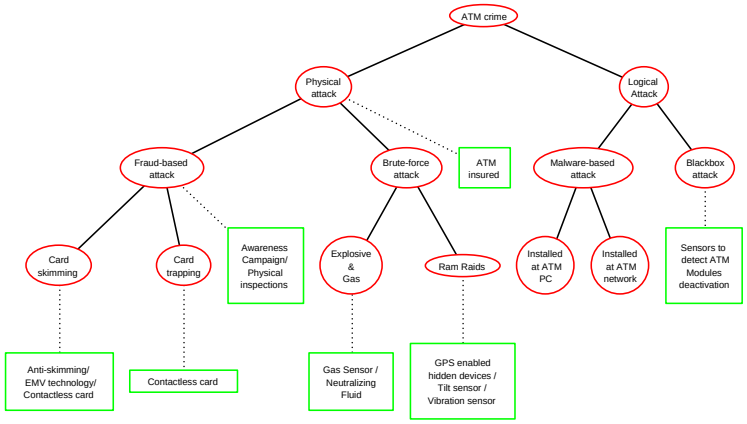

Fig. 1. An excerpt of an ADTree on ATM crime: a taxonomy.

trends in cybercrime, *National Crime Agency*[8] and *ATM Marketplace*[9] to gather recent reports on financial fraud and potential countermeasures.

Because it is convenient to structure the tree similarly to standard reports and documentation already familiar to stakeholders, lawyers, and analysts in general, we split the list of attacks and countermeasures resulting from the previously mentioned study in categories that can be found in well-known incident reports (e.g. the EAST report 2015[10] and the ATMIA Global Fraud Survey 2015[11]). This led to the taxonomy of attacks and countermeasures shown in Figure 1.

Figure 1 shows two main types of ATM fraud: those requiring highly sophisticated software (e.g. malware or blackbox devices) and those which use conventional electronic devices (e.g. card skimmers) and/or require the participation of the victim (e.g. card trapping). The first category led to the notion of *logical* attacks, which require installing malicious software on the ATM or acquiring system credentials through cyber attacks. Malware could be deployed in the ATM PC or a blackbox device connected with the ATM computer system in order to gain access to cash or sensitive data. The second category includes *physical* attacks, where a legitimate user's account is involved, which we call *fraud*, and also physical attacks jeopardizing the physical integrity of the ATM.

Figure 1 also depicts potential countermeasures. For example, card skimming and cash trapping can be prevented by anti-skimming solutions such as stress sensors, or by making compulsory the use of EMV technology (Chip&PIN) and contactless cards. A general countermeasure against this type of fraud is to make customers and employees aware of the fraud in order to perform quick physical inspections themselves. Brute force attacks in contrast, cannot be actually pre-

[8] `http://www.nationalcrimeagency.gov.uk/`
[9] `http://www.atmmarketplace.com/`
[10] `https://www.european-atm-security.eu/tag/european-atm-crime-report/`
[11] `https://www.atmia.com/whitepapers/global-fraud-survey-2015/1104/`

vented, but detected. Detection mechanisms are GPS-enabled devices to localize the ATM, tilt, vibration, and gas sensors, etc.

Capturing attack vectors in a semantically meaningful way. The ADTree we produced, an excerpt of which is shown in Figure 1, is a useful classification of attacks to ATMs, and applicable countermeasures and mitigation strategies. However, it does not benefit from the main feature of ADTrees as a mathematical language, that is, the ability to encode several attack vectors in a compact tree structure. An *attack vector* is a path or a set of attack steps an adversary can follow in order to successfully attack a system. In ADTrees, attack vectors are expressed by using the conjunctive operator AND, which expresses that all subgoals of a given goal ought to be achieved.

The main challenge when modeling many attack vectors in an ADTree is to guarantee that it is semantically meaningful, while keeping its communication potential. The team addressed this challenge by frequently executing two different verification processes. The first one consisted in checking that the taxonomy depicted in Figure 1 is preserved as much as possible. The second one consisted in keeping track of those attack vectors we expected to model, and verifying that the multiset semantics of ADTrees [7] matches this set of attack vectors. The whole process took 6 days of work, involving all four team members. Each modification to the tree was cross-checked by at least two team members, taking into account the two verification processes explained before. Next we detail one sub-branch of the full ADTree (see Figure 2); the latter can be downloaded at the ADTool official website: `http://satoss.uni.lu/members/piotr/adtool/`.

Blackbox attack. An interesting logical attack to ATMs consists in embedding a blackbox device into the ATM and connecting it to the ATM's computer system (see Figure 2). This can only be done by accessing the ATM's internal infrastructure without being detected, implying that the adversary needs to get into the facility where the ATM is located. A classical way to enter into a facility is by breaking in, e.g. through a window or a door, but the adversary could also try to social engineer an employee. As contemplated in the ADTree, a burglar alarm can deter or prevent a break in. Consequently, the adversary ought to disable the burglar alarm by, for example, using a radio network inhibitor against the used communication signal or protocol, e.g. Radio Frequency (RF) and General Packet Radio Service (GPRS), respectively. In this particular case, the defender can use anti-jamming techniques or a security guard to counteract the adversary's goal of disabling the alarm.

The use of video surveillance and burglar alarms for gas stations, required by law in many countries, can dissuade the attacker from approaching the ATM in order to install the blackbox device. However, there exist several techniques to disable a burglar alarm, e.g. RF/GPRS inhibitors, and video surveillance system, e.g. infrared light, laser, or video looping. From the defender point of view, making the camera less visible and accessible, or performing regular physical inspections, improve robustness of the video surveillance system.

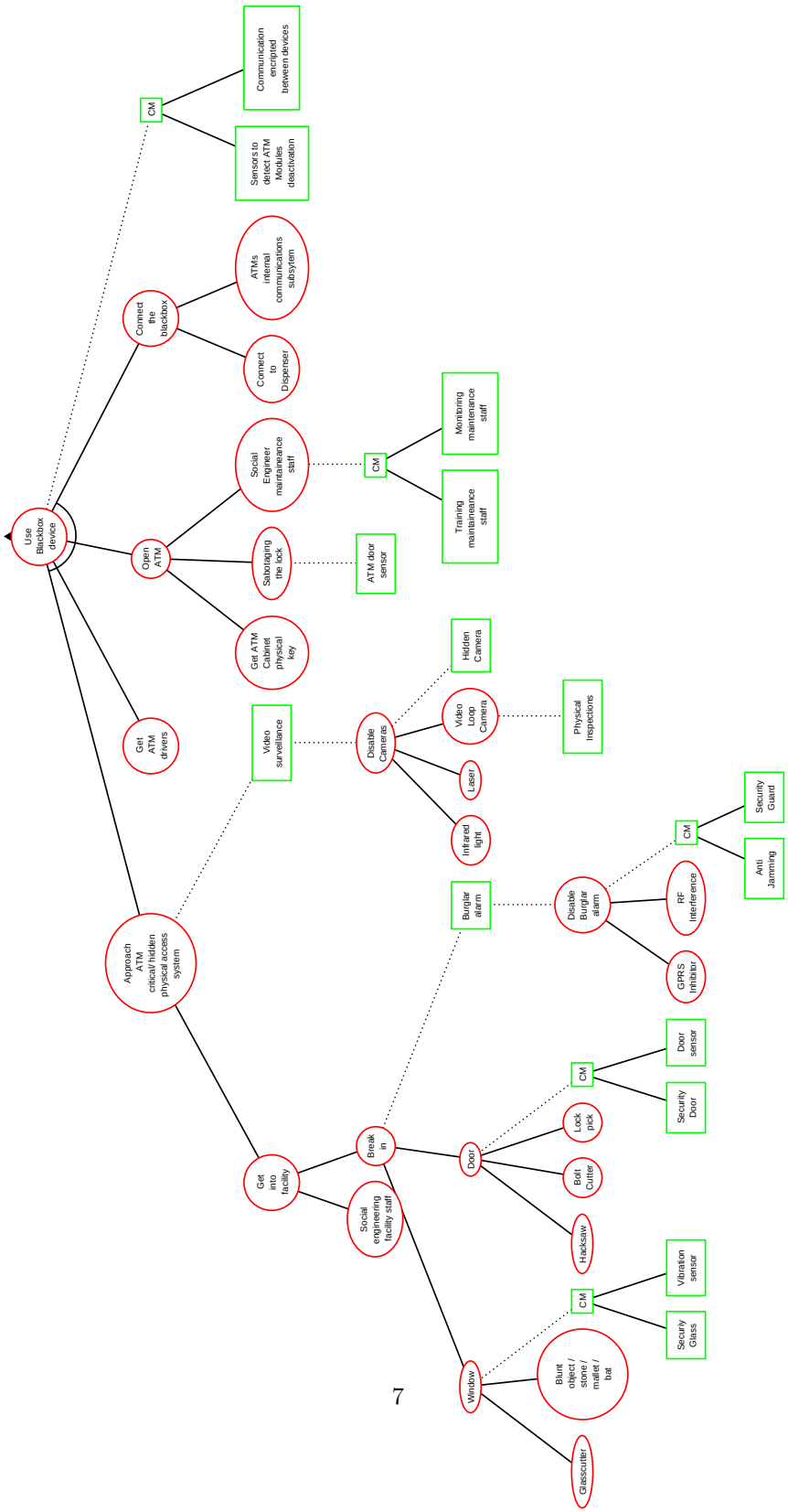

Fig. 2. A sub-branch of the ADTree modelling the use of blackbox devices.

Remark that owning a functional blackbox device means that the ATM drivers have been disclosed or stolen. Moreover, even if the adversary manages to approach the ATM, he/she still needs to insert the blackbox device into the ATM. That is to say, the adversary must open the ATM and connect the blackbox to either the dispenser or the ATM's internal communication system. Opening the ATM in our case requires getting the cabinet physical key, or social engineering the maintenance staff, or simply sabotaging the lock. As usual, the success of social engineering attacks diminishes with regular training and monitoring, while the lock sabotage can be prevented with an ATM door sensor. To finalize the description of the ADTree depicted in Figure 2, we remark that potential countermeasures against blackbox devices are: encrypting the messages exchanged between different components and devices, and a dedicated sensor that detects when a data cable has been disconnected.

It is worth mentioning that the ADTree in Figure 2 covers 900 attack vectors, called bundles in the multiset semantics in [7]. This emphasizes the modelling power of ADTrees.

5 Discussion and conclusions

The ATM case study has enabled us to explore the application of ADTrees in a challenging environment, with multiple stakeholders and a diverse range of threats. Through this process, we have evaluated positive and negative aspects of working with ADTrees, identified some best practices that improved the process, and learned a number of lessons regarding the application of the formalism. We share our findings in this section.

The intuitive graphical nature of ADTrees enables them to bridge the gap between stakeholders from diverse backgrounds, providing an environment to brainstorm, amend, document and analyze a wide range of threats. In particular, ADTrees provide a succinct and meaningful structure for a huge number of potential attack vectors. This strength was already prominent when Schneier introduced the attack tree back in 1999 [18], and our own experience reinforces this observation. However, attack trees are constructed from the attacker's perspective. Organizations are more focused on the overall risk (in terms of worst case impact) and the inventory of effective treatment options to be implemented to mitigate those risks. This is where ADTrees are more useful than attack trees.

We started to create the ADTree from a taxonomy of attacks on an ATM, which proved to be very helpful. This established attack taxonomy allowed us to structure the reasoning and compare the attacks we identified with the globally known attacks, thus serving as a reference to check the tree for completeness.

In general, we found several countermeasures that did not really prevent the attack, but triggered actions that could mitigate the impact of the attack. Handling these countermeasures required lengthy team discussions. We suspect that these challenges in handling treatment options arise from the fact that they are not clearly addressed in the ADTree methodology itself (e.g. any of the established semantics). One possibility is to extend the ADTree methodology by

explicitly typing defense nodes, following the example of attack-countermeasure trees that support detective and reactive countermeasures [16]. However, this change will also increase the cognitive load on the analysts.

References

1. A. Bagnato, B. Kordy, P. H. Meland, and P.Schweitzer. Attribute Decoration of Attack–Defense Trees. *Int. J. of Sec. Soft. Engineering*, 3(2):1–35, 2012.
2. E. J. Byres, M. Franz, and D. Miller. The use of attack trees in assessing vulnerabilities in SCADA systems. In *Proc. of the Int. Infrastructure Survivability Workshop*, 2004.
3. S. Du and H. Zhu. Security assessment via attack tree model. In *Security Assessment in Vehicular Networks*, pages 9–16. Springer, 2013.
4. 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 *Proc. of HICSS*. IEEE Computer Society, 2007.
5. O. Gadyatskaya, R. Jhawar, P. Kordy, K. Lounis, S. Mauw, and R. Trujillo-Rasua. Attack trees for practical security assessment: Ranking of attack scenarios with ADTool 2.0. In *Proc. of QEST*, volume 9826 of *LNCS*. Springer, 2016.
6. D. L. Kewley and J. F. Bouchard. DARPA information assurance program dynamic defense experiment summary. *IEEE Trans. Systems, Man, and Cybernetics, Part A*, 31(4):331–336, 2001.
7. B. Kordy, S. Mauw, S. Radomirović, and P. Schweitzer. Attack–Defense Trees. *J. of Logic and Computation*, 24(1):55–87, 2014.
8. B. Kordy, L. Pièvre-Cambacédès, and P. Schweitzer. DAG-Based Attack and Defense Modeling: Don't Miss the Forest for the Attack Trees. *Comp. Science Review*, 2014.
9. R. Kumar, E. Ruijters, and M.I.A Stoelinga. Quantitative attack tree analysis via priced timed automata. In *FORMATS*, volume 9268 of *LNCS*, pages 156–171. Springer, 2015.
10. T. Li, J. Horkoff, E. Paja, K. Beckers, and J. Mylopoulos. Analyzing attack strategies through anti-goal refinement. In *Proc. of PoEM*, pages 75–90. Springer, 2015.
11. S. Mauw and M. Oostdijk. Foundations of Attack Trees. In *Proc. of ICISC'05*, volume 3935 of *LNCS*, pages 186–198. Springer, 2006.
12. A. Morais, I. Hwang, A. Cavalli, and E. Martins. Generating attack scenarios for the system security validation. *Networking science*, 2(3-4):69–80, 2013.
13. A. Opdahl and G. Sindre. Experimental comparison of attack trees and misuse cases for security threat identification. *Information and Software Technology*, 51(5):916–932, 2009.
14. S. Paul. Towards automating the construction & maintenance of attack trees: a feasibility study. In *Proc. of GraMSec*, volume 148 of *EPTCS*, pages 31–46, 2014.
15. I. Ray and N. Poolsapassit. Using attack trees to identify malicious attacks from authorized insiders. In *Proc. of ESORICS*, pages 231–246. Springer, 2005.
16. A. Roy, D. S. Kim, and K. S. Trivedi. Attack countermeasure trees (ACT): Towards unifying the constructs of attack and defense trees. *Sec. and Commun. Netw.*, 5(8):929–943, August 2012.
17. V. Saini, Q. Duan, and V. Paruchuri. Threat modeling using attack trees. *J. of Computing Sciences in Colleges*, 23(4):124–131, 2008.
18. B. Schneier. Attack Trees. *Dr. Dobb's Journal of Software Tools*, 24(12):21–29, 1999.