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. Based on expert knowledge and available historical data, we decorate the attack-defense tree with estimations for critical parameters, such as likelihood of an attack. This allows us to evaluate the quality of the model by comparing the outcomes of the quantitative analysis on the attack-defense tree with historical data.
Our paper establishes a practical experience report, where we reflect on the process of modeling and analyzing ATM threats via attack-defense trees. In particular, we share our insights into the benefits and drawbacks of attack-defense tree modeling and analysis, as well as best practices and lessons learned. Our experience highlighted the potential for attack-defense trees to produce good quality results. However, the quality of the outcome and the speed of the process depend significantly on the composition of the team involved and the degree of mutual understanding regarding both the process and the subject matter. Specifically, to support a productive level of engagement, it is important for the subject matter expert to have a clear initial introduction to the process and for the team responsible for the modeling to use appropriate techniques for eliciting information about the environment under discussion.

Key words: attack-defense trees, security modeling, risk assessment, 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. For example, recently attackers managed to withdraw about $2 million in an ATM malware heist in Taiwan[1]. The European ATM Crime Report (EAST)[2] evaluates the loss in 2015 due to ATM 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 in their work, many methodologies have been developed. These include security methods, such as NIST SP800-30 [1], OCTAVE [5], STRIDE [36], CORAS [6]; standards for the risk management process (e.g. ISO/IEC 27005 [12]), and modeling techniques (for example, threat diagrams [6], misuse cases [37], anti-goal refinement [21], Petri nets [29], Markov processes [28], and attack trees [34]). These methodologies and techniques 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 by offering sophisticated metrics, such as expected time of attack or probability of detecting an ongoing attack.

In this paper, we report on the application of *attack-defense trees* to security risk assessment of ATMs. Attack-defense trees (ADTrees, [15]) extend the popular attack trees formalism with defenses (also called countermeasures or responses). Similarly to attack trees, ADTrees enjoy an appealing and intuitive visual representation, a structured way to brainstorm about attack scenarios [17] and formal frameworks to analyze the trees qualitatively or quantitatively [22, 15]. Additionally, ADTrees are capable of reasoning about potential countermeasures, supporting 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 and analyzing ATM security threats, and potential countermeasures via ADTrees. In particular, we report on

- *The process of ATM modeling and analysis with ADTrees.* The paper outlines the case study (Section 3), describes our process for designing a large, comprehensive attack-defense tree (Section 4) and annotating it with data values for quantitative analysis (Section 5). We share the resulting attack-defense tree and exemplify quantitative analysis of the ATM security with the likelihood of attack parameter.
- *Lessons learned and best practices for modeling and analysis with ADTrees.*

While attack trees and attack-defense trees have been applied to evaluate the security of complex systems (Section 2), the existing literature lacks insights into how to validate the resulting trees and how to overcome the issues with

[1] `http://www.reuters.com/article/us-taiwan-banks-theft-idUSKCN0ZT0Y6`
[2] `https://www.european-atm-security.eu/tag/european-atm-crime-report/`

data values for annotating the trees, such as missing values for the leaf nodes with values available for intermediary nodes.

In this paper we share techniques that we found useful when working with ADTrees and report caveats that the practitioners need to become aware of. Furthermore, we report on the positive and negative aspects of the attack-defense tree formalism experienced through its application to practical case study modeling (Section 6). Our process particularly highlighted the value of using taxonomies when working with subject experts in a specialised field such as ATM security. Taxonomies, in embodying collective knowledge in a particular area, support increased mutual understanding, greater completeness and a more rapid, higher quality outcome.

2 Background and preliminaries

Attack trees. Attack trees [34, 22] are a graphical formalism to structure, model and analyze the potential attacks on an asset. Attack trees (ATrees) elucidate how single attack steps combine into a multi-stage attack scenario leading to a security breach. ATrees analysis typically follows a top-down method to obtain these multi-stage attack scenarios, and it can be very useful to document, brainstorm, and analyze system security. For example, the NATO Research and Technology Organization [31] and the 2013 OWASP CISO Application Security Guide [25] recommend attack trees for threat assessment. ATrees (sometimes called threat trees) can be applied with the STRIDE methodology [36] and the OWASP methodology [26] for threat modeling, and the SQUARE methodology for security requirements engineering [23].

An attack tree starts with a security threat, modelled as the root of an attack 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, then one arrives at the *basic attack steps (BASs)*, sitting at the leaves of the ATree. The leaves can then be decorated with quantitative attributes, such as cost, attack times, success probability of the BASs. A wide range of quantitative analysis methods exist that, given the values for the quantitative attributes for the BASs, compute the quantitative value for top level event. In this way, one can obtain, for instance, the the cost, probability, damage of the most likely attack [13, 22, 18, 11, 2].

Attack-defense trees extend attack trees with defensive measures, also called *countermeasures*, yielding a graphical mathematical model of multi-stage attacks along with safeguards [15, 32, 39]. 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 adversary and an enterprise, i.e. an attacker, whose goal is to attack the system, and a defender who tries to protect it [15].

Example 1. Consider the running example in Figure 3 on page 9. The root (or top) of the ADTree represents the high-level attacker's goal under consideration; i.e. *ATM crime*. This goal can be achieved by either *Physical attacks* or *Logical attacks*. This is modelled as an OR-gate (disjunctive refinement). *Logical attacks* can be executed either through *Malware* or via *Black box attack*. The presence of *Sensors to detect ATM modules deactivation* serves as a countermeasure to *Black box attack*. Children and parent nodes of the same type (i.e. attack nodes or defense nodes) are connected by a straight line. Children and parent nodes of the opposite type are connected by a dotted line. An AND-gate is distinguished from an OR-gate by an arc that connects the children of the AND-gate. An AND-gate is featured in Figure 4 at the top. To use a Blackboard device, one must *Approach the ATM critical access system*; *Get the ATM drivers*; *Open the ATM*; and *Connect the black box*.

Related work. Several papers report on the applicability of attack trees in practical scenarios. DARPA has applied attack trees in Information Assurance live experiments [14, 20]. Saini et al. evaluated security of the MyProxy system from the Globus toolkit with attack trees [33], Byres et al. used attack trees to evaluate the SCADA communication systems security [4], and Ray and Poolsapassit applied the attack trees methodology to identify insider threats [30]. Security of Vehicular Ad-hoc Networks was evaluated with attack trees in [7]. In [9, 8], Edge et al. modelsan online banking scenario and homeland security through deriving protection trees from attack trees. Bagnato et al. [3] and Schweitzer [35] reported on applying attack-defense trees to model attacks on an RFID-based goods management system for a warehouse. Apart from the last ones, none of these approaches, however, included defenses.

Further, several approaches proposed to generate attack trees from system models, like [38]. Following the approach of [14], a methodology to construct attack trees based on the system architecture, risk assessment study outcomes and a security knowledge database is proposed in [27]. This methodology follows a layered approach to generate skeletons of attack trees. Morais et al. [24] 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.

3 The ATM case study

In this section we establish the context, basic architectural framework, settings, modelling choices and assumptions of the ATM case study.

Establishing the context. Figure 1 schematically depicts the gas station premises. Two main zones can be identified: the **store zone**, enabled with a security glass

door class RC3 and two security glass windows class RC2, 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.

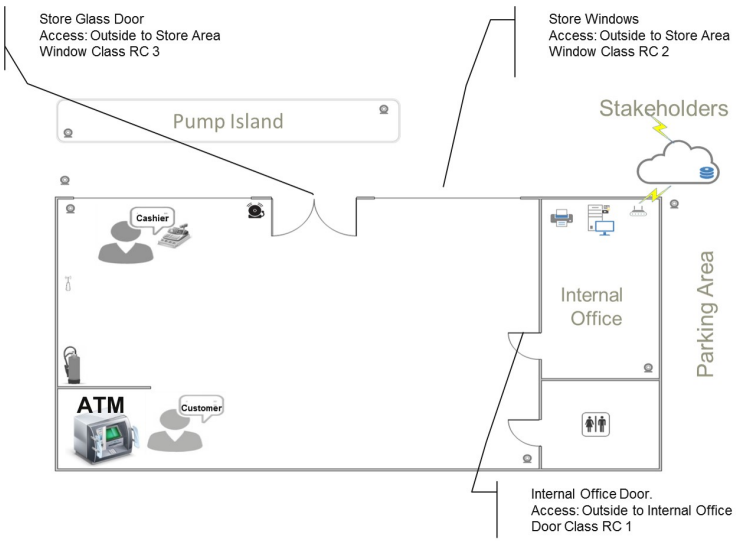

Fig. 1. ATM Case Study - Gas Station Premises.

Major credit card processors can request specific technologies to improve the security in a whole region. This is the case for EMV technology (global standard for credit and debit payments cards based on chip card technology) in Europe. However, each financial institution further defines its own treatment options taking in consideration multiple factors. In order to properly analyse the context, we describe the current countermeasures in place around the ATM and the infrastructure for our specific case study.

- **Risk transfer**: the whole local ATM network is insured to reduce potential losses from different types of attack.
- **Mitigation strategies and security controls**:
  - ATM is compliant with the EMV technology.
  - ATM is bolted to the floor.
  - ATM has a lock to prevent physical access to its internal structure.
  - The Internet connection used to connect the ATM to the ATM network is completely independent of any other existing communication in the gas station and a VPN is in place.

- The network bank has implemented a IDS/IPS solution to manage potential threats.
- The Bank IT department has a Change Control process in place.
- The Bank fraud department has heuristic controls in place to identify irregular ATM activities.
- ATM maintenance staff have been trained to perform physical inspections of the ATM to identify implementation of potential threats.
- ATM maintenance staff activities are monitored to identify irregular activities.
- Local ATM network administrators have received information security awareness sessions and training about potential ATM threats and their information security responsibilities.
- Local ATM administrator activities are monitored to identified irregular activities.
- Cardholders have received awareness campaigns related to the potential threats associated with credit/debit cards and electronic channels, such as ATM and POS.

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 four external cameras (one of them hidden) and three internal cameras, and burglar alarm enabled with several kinds of sensors (window/door vibration and movement detectors) and anti-jamming features. Also, the Security Provider has 24x7 service and direct connection with police and fire stations in case of any verified incident.
- The **ATM maintenance** function is performed by an external company that provides services to several financial organizations including the interbank provider. The services "cash reload" and "parts" (operational maintenance) are handled by the same provider.
- **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.
- **Interbank provider** is an organisation formed either by several banks or independently to offer interconnections between different ATM/POS (Point of Sales terminals) networks or bank networks. In some cases the interbank provider can have its own ATMs.
- **Bank** is an organisation that manages a range of financial services, including ATM transactions from its own ATMs or from other ATMs as issuer/acquirer. The Bank monitors its own ATM network and can use an outsourced provider to perform the maintenance function (operational maintenance and cash reload). Besides, its local ATM network is connected with other ATM networks through the Interbank provider. For this case study the ATM located at the gas station belongs to the Bank.

– **Customers** are people 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.
– **Internal employee/intruder** is an internal employee who could potentially provide information or physical access (voluntarily or not) to attackers.

4 The attack-defense tree model

Developing attack tree models for large-scale and complex systems has been traditionally a cumbersome task requiring a team of security experts. The first step towards attack tree modeling is to understand the system by identifying stakeholders, the components that make up the system, and the relevant attacker profiles. Another important aspect, often neglected in practice, is to fundamentally grasp the semantics of the attack-defense tree 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, and he played the domain expert role. The second industry expert has expertise in security assessment, financial services, and has prior experience with attack trees. She played the role of validator and was responsible for quality evaluation. The other members of the team have extensive knowledge of different semantics and analysis techniques for attack-defense trees. They were responsible for structuring the tree and handing the quantitative analysis. In the rest of this section we explain the process followed by this team to design a large and comprehensive attack-defense tree model for the ATM scenario described in the previous section.

4.1 Visualising the structure of the attack-defense tree

The attack-defense tree is not only a formal mathematical language to capture potential attacks and countermeasures for a given system, but also a powerful communication means for security experts and stakeholders. It is thus convenient to structure the tree similarly to standard reports and documentation already familiar to stakeholders, lawyers, and analysts in general.

We observe that incident reports already present events and incidents within one or several categories. Typically, each event is of a particular type, and it is provided together with information about the attacker's goal and the attack itself. We, for example, used for this case study the European ATM Crime Report (EAST) 2015[3] and the ATMIA Global Fraud Survey 2015[4], which cover ATM fraud incidents in Europe. The implicit type relation between events in those reports allows us to create a taxonomy of attacks, which indeed resembles an attack tree (see Figure 2).

[3] `https://www.european-atm-security.eu/tag/european-atm-crime-report/`
[4] `https://www.atmia.com/whitepapers/global-fraud-survey-2015/1104/`

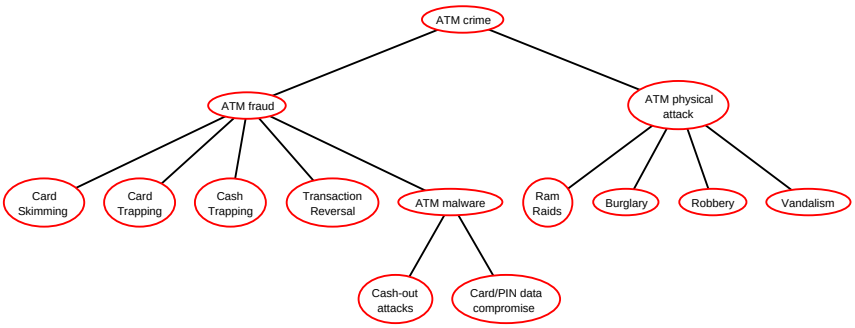

Fig. 2. Taxonomy of ATM crime.

The taxonomy depicted in Figure 2 shows popular attacks against ATMs, which are first categorized into physical attacks and fraud-related attacks. Physical attacks require brute-force in order to tamper with the physical integrity of the ATM, while ATM fraud attacks are more sophisticated and involve compromising credentials of legitimate accounts via a variety of techniques, such as card skimming or malware (malicious software) infection.

We stress that the advantages of choosing a taxonomy as a starting point for attack-defense tree design are threefold. First, it focuses the attention of the expert team on the most relevant attacks that ought to be considered. Second, it suggests a tree structure similar to official reports, implying that the communication potential of the generated attack-defense tree increases. Third, it does not require expertise on attack tree semantics. The evolution of this taxonomy into a comprehensive attack-defense tree is detailed in the rest of this section.

4.2 Overcoming the lack of attack intelligence

The task of mapping a security scenario into an attack tree greatly depends on the security expertise of the person or team developing the attack tree. However, security expertise needs to be complemented with data about previous attacks. Such data could come in the form of an attack pattern library [5], 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 other attackers to exploit similar vulnerabilities.

If budget and resources permit it, security data can be gathered by penetration testing which helps to create a map of the system. This is indeed the approach pursued by the Defense Advanced Research Projects Agency (DARPA) [14], back in 2001. The task was conducted by a so-called *red team*, which identified vulnerabilities and constructed an attack tree considering a well-resourced adversary, such as a foreign or national intelligence agency. After successfully executing their intended actions without being detected, the red

[5] `www.sei.cmu.edu/reports/01tn001.pdf`

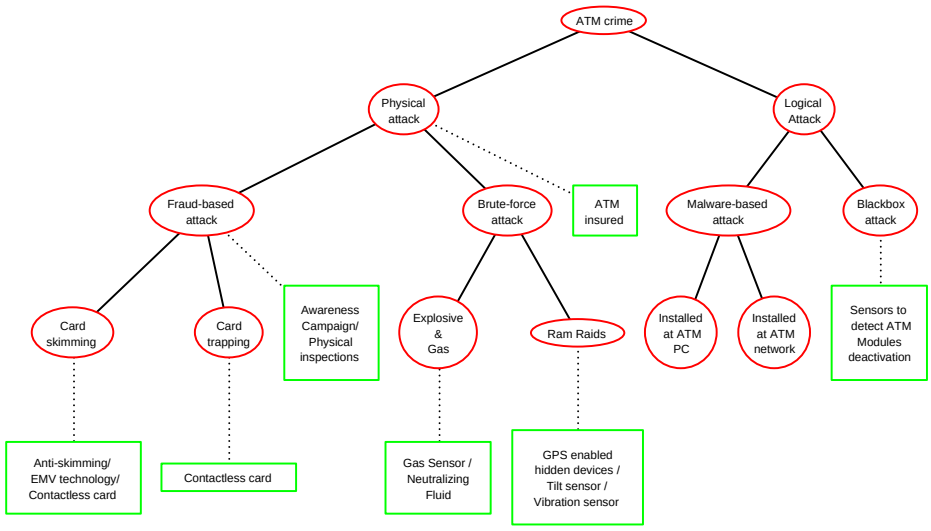

Fig. 3. An excerpt of an attack–defence tree on ATM crime.

team provided the created attack tree, which helped the agency to modify defensive countermeasures in the system. However, this methodology is expensive, time consuming, and therefore unaffordable for many companies.

For this case study, the financial services security specialist overcame this problem by using different sources of information on ATM security, such as *PCI-DSS ATM Security Guidelines*[6] to understand how secure channels for payment systems based on smartcards are implemented, *ATM Industry Association*[7] to collect best practices in ATM security, *European Union's law enforcement agency*[8] to depict recent trends in cybercrime, *National Crime Agency*[9] and *ATM Marketplace*[10] to gather recent reports on financial fraud and potential countermeasures. This study resulted in a better understanding of the above taxonomy, therefore leading to a more fine-grained attack tree with countermeasures, i.e. to an actual attack-defense tree as depicted in Figure 3. We remark that, due to space constraints and anonymization requirements, Figure 3 only shows extracts of the full attack-defense tree.

Our first step towards developing a more fine-grained taxonomy of attacks and countermeasures was to refine the notion of ATM fraud. We observed that two main types of ATM fraud exist: those requiring highly sophisticated software (e.g. malware or blacbox devices) and those which use conventional electronic devices (e.g. card skimmers) and/or require the participation of the victim (e.g.

[6] `https://www.pcisecuritystandards.org/pci_security/`
[7] `https://www.atmia.com/`
[8] `https://www.europol.europa.eu/`
[9] `http://www.nationalcrimeagency.gov.uk/`
[10] `http://www.atmmarketplace.com/`

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. A piece of malicious software could be malware 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.

According to the new taxonomy depicted in Figure 3, card skimming is still considered a fraud as in Figure 2, with the peculiarity that it is also regarded as a physical attack in the new taxonomy. Notice that this apparent contradiction comes from a design problem that appears frequently in tree-based modeling languages, which is, how to efficiently represent similar, yet different, concepts and notions. For example, in Object Oriented Programming the class *Bird* would normally contain a functional method *fly*. However, penguins do not fly, implying that in order to represent penguins in the model, one should either come up with an odd design where birds may fly or not, or accept the inconsistency. For this case study, we were looking for simplicity and accepted the inconsistency that some logical attacks may require physical attack steps, and vice versa.

A visible improvement of the model in Figure 3 is the presence of 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 prevented, but detected. Detection mechanisms are GPS-enabled devices to localize the ATM, tilt, vibration, and gas sensors, etc. A fairly recent commercially available prevention technique against gas attacks is *neutralizing fluid*, which is expected to delay the gas explosion for around 20 minutes.

4.3 Capturing attack vectors in a semantically meaningful way

The attack-defense tree we produced, an excerpt of which is shown in Figure 3, is a useful classification of attacks to ATMs, and applicable countermeasures and mitigation strategies. However, it does not benefit from the main feature of attack–defense trees 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 attack-defense trees, attack vectors are expressed by using the conjunctive operator AND, which expresses that all sub-goals of a given goal ought to be achieved.

The main challenge when creating a large attack-defense tree 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 taxonomies depicted in Figures 2 and 3 are 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 attack-defense trees [15] matches this set of attack vectors. As an example, let us consider the refinement of ATM crime into physical and logical attack. Initially, the node ATM crime was refined conjunctively, in order to express that most attacks involve both physical and logical steps. The team soon realized, after developing the canonical form of the multiset semantics [15], that such a refinement leads to unrealistic attack vectors, e.g. the use of a explosive and malware at the same time. 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 attack-defense tree (see Figure 4).

4.3.1 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 4). 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 attack–defense tree, 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. As we mentioned, an interesting feature of attack-defense trees is that the game between attacker and defender can be easily modeled. In this particular case, the defender can use anti-jamming techniques or a security guard to counteract the adversary's goal of disabling the burglar 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.

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 attack-defense tree depicted in Figure 4, we remark that potential countermeasures against blackbox devices are: encrypt-

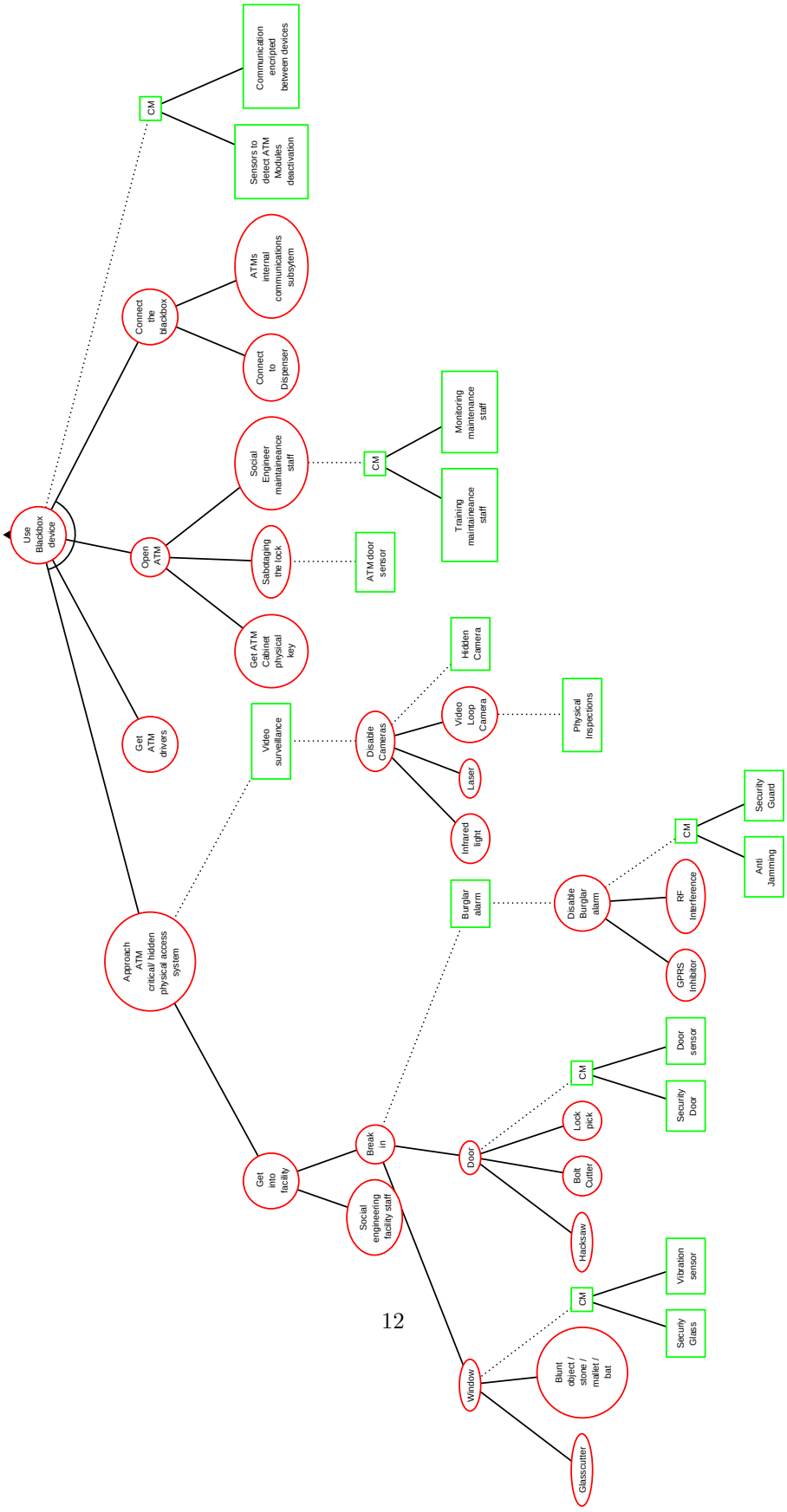

Fig. 4. A sub-branch of the attack-defense tree modelling the use of blackbox devices.

ing 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 attack-defense tree in Figure 4 covers 900 attack vectors, called bundles in the multiset semantics in [15]. This emphasizes the modelling power of the attack-defense tree model.

5 Quantitative analysis

A quantitative cyber risk assessment formalisms, such as attack-defense trees, provides a structured framework to answer several security-related questions, such as – "How frequent successful attacks happen?" or "What's the minimum cost/time to attack/defend the system?". In particular, we are interested on how likely ... We have analyzed this question by using a similar workflow to that proposed in [16], which is detailed next.

1. Describe the system (Section 3) and model it via an attack-defense tree (Section 4).
2. Informally elicit security questions and map the security questions onto attribute domains (Subsection 5.1.1).
3. Decorate the basic attack/defense steps with data values (Subsection 5.1.2).
4. Use the bottom-up evaluation algorithm to obtain the attribute values for refined nodes (Subsection 5.1.3).
5. Analyze and validate results (Subsection 5.2).

5.1 Attack-defense tree decoration

5.1.1 Security question and attribute domain. We can use attack-defense trees to answer questions pertaining to either a single player (the attacker or the enterprise), or both players. To show the applicability of our approach, we first frame a question of interest to security managers, that is, "*Given the countermeasures in place, what is the likelihood of an ATM crime attack?*", where *likelihood* is defined as the probability of occurrence of an attack. In the attack-defense tree formalisms [15], the probability attribute domain is defined as follows.

The *probability attribute domain* on an attack-defense tree is a tuple $P = (V, \beta)$, where V is a set of values and β a function that assigns an attribute value from V to every leaf node. We use $(\nabla^a, \triangle^a)$ and $(\nabla^d, \triangle^d)$ to denote OR and AND refinements of an attacker and defender goals, respectively. We also use $c^a(t_1, t_2)$ to denote the attack-defense tree with attacker goal t_1, which is counteracted by the defensive mechanism modeled by the tree t_2. Similarly, $c^d(t_1, t_2)$ represents tree with defensive goal t_1, which is counteracted by the attack modeled by the tree t_2.

Definition 1. *The probability attribute domain $P = (V, \beta)$ on an attack-defense tree t can be computed bottom-up as follows.*

$$P(t) = \begin{cases} \beta(t) & \textit{if } t \textit{ is a leaf node} \\ 1 - \prod_{i=k}^{i=1}(1 - (P(t_i)) & \textit{if } t = \nabla^s(t_1, \ldots, t_k) \textit{ with } s \in \{a, d\} \\ \prod_{i=k}^{i=1} P(t_i) & \textit{if } t = \triangle^s (t_1, \ldots, t_k) \textit{ with } s \in \{a, d\} \\ P(t_1)(1 - P(t_2)) & \textit{if } t = c^s(t_1, t_2) \textit{ with } s \in \{a, d\} \end{cases}$$

We observe that in the above definition basic attack steps are assumed to be independent.

5.1.2 Deriving input data. Our attack-defense tree model requires each basic attack/defense step (leaf nodes) to be annotated with values data for each considered attribute domain. In our case study, we use probability of occurrence to derive the overall likelihood of an ATM crime attack. The domain expert in the team provided an initial estimation of likelihood values for all leaf nodes in the tree, which were later validated by other team members.

Note that, some defensive nodes were assigned with probability 0 as they are still not deployed. Nevertheless, further analysis can be performed by introducing *what-if* questions, i.e. how does the likelihood change if a given countermeasure is implemented? Such type of analysis, know as sensitive analysis, is out of scope, though. Risk transfer treatment options were also assigned with value 0, as these countermeasures do not reduce probability of occurrence of an attack.

5.1.3 Bottom-up computation. Once we decorated all the basic attack steps/defense steps, we applied the single parameter bottom-up technique to propagate the quantitave towards the root of the tree. This task was accomplish supported by the ADtool 2.0 [10].

5.2 Quality evaluation

One interesting outcome of this case study is that we have managed to use the quantitative analysis process to measure the quality of the model. This is done iteratively by first decorating the leaves with input data, computing the attribute values for the intermediate nodes, and comparing the obtained values with estimates from historical data and surveys. If there is a large mismatch, we can again go back to the first step until we establish a decorated model with the desired accuracy with respect to historical data.

To illustrate the approach, we consider a small subtree as in Figure 5. On the one hand, given the input data in the leaf nodes, we obtain that the likelihood of *Card skimming* is 0.272. On the other hand, according to historical data and global survey reports in the banking industry, we estimated that the likelihood of Card skimming is around 0.3. This discrepancy is indeed small, which indicates that both the model and the provided data are consistent with respect to background knowledge.

Moreover, we can find which countermeasures or which basic attack step is the most critical with respect to the chosen metric. This can be done through sensitivity analysis, where we run the analysis multiple times, and in each run, we vary slightly one basic attack step data value, while keeping the others fixed.

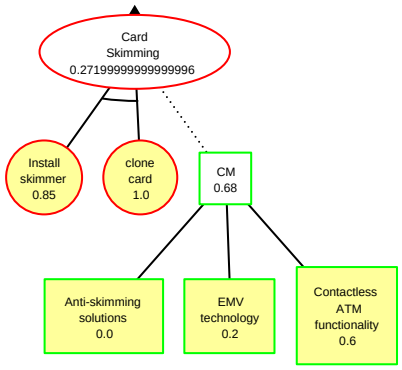

Fig. 5. An example of the bottom-up quantification method on a subbranch (Card Skimming) of the attack-defense tree.

We can observe the percentage change in, for example, likelihood of the attack goal caused by small changes in likelihood of the chosen basic attack step by using the Birnbaum importance measure [19].

We used this estimate of card skimming likelihood to validate the opinion of our expert on the likelihood data for basic attack/defense steps under card skimming, i.e. *Install skimmer, clone card, countermeasure*, as shown in Figure 5.

6 Our findings on practical application of ADTrees

The ATM case study has enabled us to explore the application of attack-defense trees in a challenging environment, with multiple stakeholders and a diverse range of threats. The ability to combine digital and physical risks in a single analysis is a significant strength, but also a considerable challenge. The process required extensive input from the case study owner during the construction of the model, however the results of the analysis provided real insight into the nature, extent and implications of the different threats challenging those responsible for managing estates of ATMs.

Through this process, we have evaluated positive and negative aspects of working with attack-defense trees, 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.

Benefits.

ADTrees support communication in a multi-stakeholder environment. The intuitive graphical nature of attack-defense trees enable them to bridge the gap between stakeholders from a diverse range of backgrounds and disciplines. It provides a clear, visible environment in which to brainstorm, discuss, amend, document and analyze a wide range of threats. In particular, ADTrees provide

a meangingful structure for the huge number of potential attacks (attack vectors). This particular strength was already prominent when this approach was proposed by Schneier back in 1999 [34], and our own experience reinforces this observation.

ADTrees are well suited to analysis of both current and potential scenarios. We have found that where the context is well-defined from the outset, attack-defense trees can provide a valuable means for charting the full range of issues relating to that context. Once those issues are charted, it is quite easy for practitioners to focus on particular aspects of the tree, apply the single-parameter quantitative analysis, and identify inter-relationships, the impact of particular countermeasures, and also perhaps unintended consequences of particular design choices. Furthermore, once the tree is designed, it is relatively easy for the analysts, even novices in the methodology, to perform "*what-if*" analysis by adding or removing nodes, varying attribute values, and so on.

Challenges.

Scalability and maintenance of attack-defense trees can be challenging. In general, attack-defense trees can be hard to manage, due to the size of the tree required to model a complex enterprise environment. Once the designed tree reaches a certain size, analysts can face cognitive issues when analyzing (qualitatively) or extending this model. This can particularly be the case when revisiting the tree after an extended period of time. This can, to some extent, be alleviated by appropriate annotations.

ADTrees miss dynamic features. ADTrees are a relatively static model, and they do not fully capture the dynamic nature of attacker and system behaviour. For example, in the ATM case study, whenever an attack is executed, it is logged. This is done in order to comply with banking regulations. This data could be a relevant source of information for the analyst, as it can be used to validate/extend the designed trees or perform quantitative analysis. It would be interesting to see integration of the analysis with this post-attack data log in the context of ADTres.

ADTree analysis requires an investment in data quality. A frequently documented challenge in security risk assessment via graphical modelling techniques such as attack-defense trees is acquisition of input data [3, 35]. Furthermore, the standard approach for attack trees, when only leaf nodes are annotated with values seems overly restrictive, as often data for intermediate nodes can be more readily available than data for leaves. This observation is further reinforced if we consider the costs of data collection in an organization (in terms of personnel, time, specialized resources, etc.). Indeed, we found that often more generic data than we expected was available, e.g. from historical databases, multiple surveys and empirical results. The real challenge for us was to correlate the data acquired from different sources and normalize it by using both geographical and technological inputs, while applying relevant risk assessment tools. We see that an extension to the attack-defense tree quantitative analysis techniques is required that will take into account the data handling issues (when data for some

leaves is missing, when only intermediate node data is available, if the analyst is uncertain about some data values).

Best practices.

Attack taxonomies are valuable guidelines for designing trees. We started to create the ADTree from a taxonomy of attacks on an ATM. This taxonomy proved to be very helpful. Using an established attack taxonomy allowed us to structure the reasoning, to compare the attacks we identified with the globally known attacks, thus serving as a reference to check the tree for completeness. Furthermore, empirical data is generally available for the attacks mentioned in the taxonomy. This data can be further used for analysis and validation purposes.

Different data sources can be aggregated for qualitative analysis. In our case study, data aggregation efforts have been very important. Several appropriate data sources were first identified to obtain reliable estimations of intermediate nodes in an attack-defense tree. This data could then be used to validate expert opinion about them. For instance, where the domain expert concluded based on his experience that particular features were contributing significantly to the level of risk at the ATM, this conclusion was further confirmed by reference to statistical data regarding actual attacks.

Tree validation techniques can ensure good results. To ensure that the designed tree is meaningful and reflects the threat scenarios we want to capture, we have applied two validation techniques. The first technique consisted in analyzing the current tree design and interpreting it in terms of bundles (in the multiset semantics of attack-defense trees [15]). This step ensured that the attack scenarios captured by the tree were meaningful and no expected attack vectors were missing in the tree. The second technique was quality evaluation of the annotated tree. Any discrepancy between the expected attribute value (estimated by the domain expert or retrieved from relevant documentation) and the value computed via the bottom-up computation approach is a sign of potential mismatch between the reality and the designed tree. These discrepancies were carefully reviewed and addressed by either redesigning the tree or reassessing the data values involved.

Lessons learned.

Defenses in ADTrees bridge the gap between the defensive goal of risk analysts and attack tree models. Attack trees are usually constructed from the attacker perspective, i.e. given a system design, what are the ways in which an attacker can reach a particular asset. This approach is helpful as it can be used to predict the behaviour of the attacker against the system design. However, organizations are frequently more focused on the overall risk (in terms of worst case impact) and the most effective inventory of treatment options which can be implemented to mitigate those risks. This is where attack-defense trees are more useful than attack trees. In many cases, the main focus of security risk analysis is prioritization of risks rather than prevention of individual risks. In this respect, quantitative analysis techniques on attack-defense trees need to be further extended to support the risk prioritization task.

Attack-defense trees present greater challenges than attack trees. The initial construction of the attack tree was a comparatively intuitive, if somewhat lengthy, process. However, identifying a clear and precise procedure to add defenses and their corresponding attribute values to the initial attack tree proved significantly more challenging. There was a need to handle different types of treatment options (risk reduction, acceptance, transfer and avoidance) and different types of controls (preventive, detective and reactive). It thus became rather complex to add countermeasures and their corresponding attribute values/domains. For example, a user may block his credit card once he detects it has been stolen. So it seems natural to add *Block card* as a countermeasure of the attack *Steal card*. However, blocking the card does not in fact affect the action of stealing a card. We considered adding this countermeasure elsewhere, but have not found a satisfactory position for it. 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 group discussions. We suspect that these challenges in handling countermeasures and treatment options arise from the fact that they are not clearly addressed in the attack-defense tree methodology itself (e.g. any of the established semantics). One possibility is to extend the attack-defense tree methodology by explicitly typing defense nodes, following the example of attack-countermeasure trees that support detective and reactive countermeasures (but not preventive) [32]. However, this change will also increase the cognitive load on the analysts.

Domain knowledge is pivotal in ADTree modeling and analysis. Our relatively small case study required 6 days of work for the team of four people. The case study owner has also spent 10 days preparing the documentation, collecting the data, and outlining the scenario. Overall, we found that the amount of effort required to design a single attack-defense tree model is quite significant. This was due to the steep learning curve of the method, as a lot of initial time was invested for learning how to develop an ADTree. Once the methodology was mastered by all participants, the development became more fluid. We have also found that the process curve of the attack-defense tree methodology is quite steep. The planning phase and the tree design for the "*as-is*" scenario required a lot of time. Subsequent "*what-if*" analysis and experimentation with different treatment options and data values were much faster.

Organizations require a light-weight security risk assessment framework, which can be understood easily and serve as a decision support system to evaluate their existing and potential technical and administrative controls. The initial investment required to create the attack-defense tree may constitute a considerable overhead in such environments.

7 Conclusions and future research

In this paper we shared our experiences with modeling and analyzing attacks and defenses in the ATM domain with ADTrees. We can conclude that attack-

defense trees are a very powerful formalism, which can encompass large numbers of attack vectors in a comparatively manageable space. In most enterprises, it would be most desirable for a subject matter expert to work directly with a modelling specialist experienced in working with attack-defense trees. In the initial stages, the formalism can seem wide-ranging and at times demanding, however it rapidly becomes more familiar and consequently intuitive.

The success of this approach is very dependent on the relationship between the subject matter expert and the modelling specialist. In addition to handling the process of eliciting the relevant data, the modelling specialist needs to take a gradual approach to creating the tree, in order to ensure that the subject matter expert remains engaged. The subject matter expert, for their part, must be open to the new experience of constructing the tree and frequently reconstructing it under specialist guidance.

Although these early stages can seem onerous for both parties, the tree can provide a firm foundation for decision-making on an ongoing basis. It also provides a valuable means for the subject matter to explain relevant features and decisions to non-specialist colleagues. In this respect, its true value becomes clear when it is revisited and updated with input from various parties over an extended period of time. Retaining earlier copies of the tree, as it develops over time, can also help to support traceability in the decision-making process.

It is clear that taxonomies have the potential to make a very valuable contribution in the development of attack-defense trees. This relationships could usefully be explored further, in addition to the various approaches to handling countermeasures under different circumstances.

References

1. NIST Special Publication 800-30 Revision 1. Guide for conducting risk assessments `http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecialpublication800-30r1.pdf`, 2012.
2. F. Arnold, H. Hermanns, R. Pulungan, and M.I.A. Stoelinga. Time-dependent analysis of attacks. In *POST*, volume 8414 of *LNCS*, pages 285–305. Springer, 2014.
3. Alessandra Bagnato, Barbara Kordy, Per Håkon Meland, and Patrick Schweitzer. Attribute Decoration of Attack–Defense Trees. *International Journal of Secure Software Engineering, Special Issue on Security Modeling*, 3(2):1–35, 2012.
4. Eric J Byres, Matthew Franz, and Darrin Miller. The use of attack trees in assessing vulnerabilities in scada systems. In *Proceedings of the international infrastructure survivability workshop*, 2004.
5. Richard A. Caralli, James F. Stevens, Lisa R. Young, and William R. Wilson. Introducing OCTAVE Allegro: Improving the information security risk assessment process. Technical report, CERT, 2007.
6. CORAS. `http://coras.sourceforge.net/`, 2016.
7. Suguo Du and Haojin Zhu. Security assessment via attack tree model. In *Security Assessment in Vehicular Networks*, pages 9–16. Springer, 2013.

8. 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*, pages 1–7, Oct 2006.
9. Kenneth Edge, Richard Raines, Michael Grimaila, Rusty Baldwin, Robert Bennington, and Christopher Reuter. The use of attack and protection trees to analyze security for an online banking system. In *Proceedings of the 40th Annual Hawaii International Conference on System Sciences*, HICSS '07. IEEE Computer Society, 2007.
10. 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.
11. K. Ingols, M. Chu, R. Lippmann, S. Webster, and S. Boyer. Modeling modern network attacks and countermeasures using attack graphs. In *Annual Conference on Computer Security Applications ACSAC '09*, pages 117–126, Dec 2009.
12. ISO, Geneva, Switzerland. *ISO/IEC 27005 - Information technology – Security techniques – Information security risk management*, ISO/IEC 27005:2011 edition, 20011.
13. A. Jürgenson and J. Willemson. Computing exact outcomes of multi-parameter attack trees. In *Proceedings of the OTM 2008 Confederated International Conferences, CoopIS, DOA, GADA, IS, and ODBASE 2008. Part II*, OTM '08, pages 1036–1051. Springer, 2008.
14. 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.
15. Barbara Kordy, Sjouke Mauw, Saša Radomirović, and Patrick Schweitzer. Attack–Defense Trees. *Journal of Logic and Computation*, 24(1):55–87, 2014.
16. Barbara Kordy, Sjouke Mauw, and Patrick Schweitzer. Quantitative Questions on Attack–Defense Trees. In *Proc. of ICISC*, volume 7839 of *LNCS*, pages 49–64. Springer, 2012.
17. Barbara Kordy, Ludovic Piètre-Cambacédès, and Patrick Schweitzer. DAG-Based Attack and Defense Modeling: Don't Miss the Forest for the Attack Trees. *Computer Science Review*, 2014. DOI: 10.1016/j.cosrev.2014.07.001.
18. Rajesh Kumar, Enno Ruijters, and Mariëlle Stoelinga. Quantitative attack tree analysis via priced timed automata. In *13th International Conference on Formal Modeling and Analysis of Timed Systems, FORMATS*, pages 156–171, 2015.
19. Lawrence Mark Leemis. *Reliability: Probabilistic Models and Statistical Methods*. Prentice Hall, 1995.
20. David Levin. Lessons learned in using live red teams in IA experiments. In *3rd DARPA Information Survivability Conference and Exposition (DISCEX-III 2003)*, pages 110–119, 2003.
21. Tong Li, Jennifer Horkoff, Elda Paja, Kristian Beckers, and John Mylopoulos. Analyzing attack strategies through anti-goal refinement. In *Proc. of PoEM*, pages 75–90. Springer, 2015.
22. Sjouke Mauw and Martijn Oostdijk. Foundations of Attack Trees. In *Proc. of ICISC'05*, volume 3935 of *LNCS*, pages 186–198. Springer, 2006.
23. Nancy Mead. SQUARE Process `https://buildsecurityin.us-cert.gov/articles/best-practices/requirements-engineering/square-process`, 2013.
24. Anderson Morais, Iksoon Hwang, Ana Cavalli, and Eliane Martins. Generating attack scenarios for the system security validation. *Networking science*, 2(3-4):69–80, 2013.

25. OWASP. CISO AppSec Guide: Criteria for managing application security risks, 2013.
26. OWASP. Application threat modeling `https://www.owasp.org/index.php/Application_Threat_Modeling`, 2016.
27. Stéphane Paul. Towards automating the construction & maintenance of attack trees: a feasibility study. In *Proc. of GraMSec*, volume 148 of *EPTCS*, pages 31–46, 2014.
28. Ludovic Piètre-Cambacédès and Marc Bouissou. Beyond attack trees: dynamic security modeling with Boolean logic Driven Markov Processes (BDMP). In *Dependable Computing Conference (EDCC), 2010 European*, pages 199–208. IEEE, 2010.
29. Srdjan Pudar, Govindarasu Manimaran, and Chen-Ching Liu. Penet: a practical method and tool for integrated modeling of security attacks and countermeasures. *Computers & Security*, 28(8):754–771, 2009.
30. Indrajit Ray and Nayot Poolsapassit. Using attack trees to identify malicious attacks from authorized insiders. In *Proc. of ESORICS*, pages 231–246. Springer, 2005.
31. NATO Research and Technology Organisation (RTO). Improving Common Security Risk Analysis. Technical Report AC/323(ISP-049)TP/193, North Atlantic Treaty Organisation, University of California, Berkeley, 2008.
32. 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.
33. Vineet Saini, Qiang Duan, and Vamsi Paruchuri. Threat modeling using attack trees. *Journal of Computing Sciences in Colleges*, 23(4):124–131, 2008.
34. Bruce Schneier. Attack Trees. *Dr. Dobb's Journal of Software Tools*, 24(12):21–29, 1999.
35. Patrick Schweitzer. Attack-defense trees, 2013. PhD Dissertation.
36. Adam Shostack. *Threat modeling: Designing for security.* John Wiley & Sons, 2014.
37. Guttorm Sindre and Andreas L Opdahl. Eliciting security requirements with misuse cases. *Requirements engineering*, 10(1):34–44, 2005.
38. J. Xu, K. K. Venkatasubramanian, and V. Sfyrla. A methodology for systematic attack trees generation for interoperable medical devices. In *2016 Annual IEEE Systems Conference (SysCon)*, pages 1–7, April 2016.
39. S.A. Zonouz, H. Khurana, W.H. Sanders, and T.M. Yardley. Rre: A game-theoretic intrusion response and recovery engine. *IEEE Transactions on Parallel and Distributed Systems*, 25(2):395–406, 2014.