Attack-tree series: A case for dynamic attack tree analysis

Olga Gadyatskaya and Sjouke Mauw

CSC/SnT, University of Luxembourg, Luxembourg
olga.gadyatskaya@uni.lu, sjouke.mauw@uni.lu

Abstract. Attack trees are a popular model for security scenario analysis. Yet, they are currently treated in the literature as a static model and are not suitable for dynamic security monitoring. In this paper we introduce *attack-tree series*, a time-indexed set of attack trees, as a model to capture and visualize the evolution of security scenarios. This model supports changes in the attack tree structure as well as changes in the data values. We introduce the notion of a *temperature function* as a special type of attribute that expresses the importance of change in the data values. We also introduce a *consistency* predicate on attack trees to allow inter-relating the evolving scenarios captured as attack trees. Finally, we discuss various application scenarios for attack-tree series and we demonstrate on a case study how the proposed ideas can be implemented to visualize historical trends.

1 Introduction

Today, organizations face unprecedented difficulties to stay secure. Cyber-weaponry becomes more and more commoditized, new vulnerabilities are reported every day, and attack surfaces become ever more complex and interdependent. To win this game against cyber-adversaries, security officers and analysts need to be able to access threat data over time, allowing them to pick up relevant trends and to proactively upgrade security [34].

Useful threat data can come in *quantitative* or *descriptive* formats. Frequency of attacks, cost of exploit kits, and number of lost data records are examples of the former category, while malicious IP addresses, indicators of a new adversary, recent zero-day attacks are examples of the latter one. Access to the latest threat data can be arranged, e.g., via subscriptions to threat feeds and monitoring of the relevant media channels [22,34]. Generally, organizations also collect a lot of threat data internally, from a security information and event management (SIEM) system.

All these threat data types and sources are highly dynamic, but not all threat modelling and analysis techniques applied in organizations are able to accommodate analysis of such data over time.

In particular, in this paper we focus on *attack trees*. Also known as threat trees, they are a popular modelling notation for expressing security threat scenarios. Attack trees are widely applied in organizations to capture anticipated

attack scenarios [31,25,13], to facilitate brainstorming [6], and to estimate different aspects of the considered threats via quantitative analysis [3,17].

Yet, existing attack tree design methodologies and quantitative analysis (QA) techniques focus exclusively on *static scenarios*, in which each tree node and data value is fixed. There is no systematic way to detect trends in dynamic security scenarios on attack trees or to visualize the evolution of the security posture with respect to the considered threat model. This is an unfortunate omission. Indeed, an organization may have access to very relevant threat data over time, but it does not have a way to systematically link this valuable data to the attack tree model.

We argue that there is a case for enhancing attack trees for dynamic, evolving scenarios. In this position paper, we propose an approach to systematically capture the dynamic nature of both facets of the threat data, quantitative and descriptive, in an attack tree. We make the notion of *time* explicit in the definition of attack trees by developing an idea of *attack-tree series* that are not static but accommodate a sequence of attack trees.

We develop the first formalization of attack-tree series (ATSs) in this paper. In our formalization, we allow quantitative threat data values assigned to tree nodes to change over time. We also propose to capture the *importance* of the change as a *temperature* function that enables analysis and monitoring of historical trends. Our methodology supports scenarios in which not only the data, but also the attack tree itself evolves over time. This allows for a dynamic description of the various threats and their relations. In order to develop a consolidated view on an evolving series of attack trees, we introduce the notion of *consistency*. Consistency of a series of attack trees makes it possible to relate the data values attributed to the nodes of the various attack trees to each other.

To better highlight the applicability of the new attack-tree series, we also discuss potential use cases for them and demonstrate their usefulness for highlighting historic trends with an attack tree capturing automated teller machines fraud.

2 Related work

In short, an attack tree represents a collection of attack scenarios. The *main goal* of the adversary common to all these attacks is represented by the *root* of this tree. The root is iteratively *refined* into more detailed attack components. The refinement process uses well-defined decomposition operators, typically `OR` and `AND`, and it continues until the analysts are satisfied that the nodes at the lowest level are atomic attack steps. These unrefined nodes are called *leaves* of the tree.

Attack trees have been introduced by Amoroso [2] and Salter, Saydjari, Schneier and Wallner [30], and formalized by Mauw and Oostdijk [23]. The basic attack tree model has been further extended into attack-defense trees [16] and attack-countermeasure trees [29] that both feature nodes representing security controls.

There exist several approaches to design an attack tree. An expert, or a group of experts, may design a tree manually, by analytically and iteratively considering all relevant attack developments. This is the traditional approach in the threat modelling literature [31,32]. The manual work may be facilitated by relevant knowledge, e.g., from industry-specific catalogues of threats or threat ontologies [6]. Recently, automated and assisted attack tree generation techniques have emerged [14,35,27,11,15]. However, all these approaches work with static scenarios, and they do not take into account potential evolution in the considered threat structure.

The most popular quantitative analysis technique for attack trees is the *bottom-up computation approach*, in which leaf nodes are assigned values representing *attributes* (e.g., cost or probability of the corresponding atomic attack step). These values are then propagated bottom-up using the attribute rules for the corresponding decomposition operator [23,31]. Various attributes that can be computed on an attack tree with the bottom-up computation approach are proposed by Bagnato et al. [3] and Kordy et al [17]. For example, one can estimate probability of a particular threat, cost of an attack for the adversary or for the defender, satisfiability of attacks, time till successful attack, and many other parameters [3,17]. These quantitative analysis techniques are used, for instance, to perform security risk analysis [26,19] and to make decisions about security investments [9]. There exist other QA techniques for attack trees, usually involving the transformation of an attack tree into another model, e.g., timed automata [19,8].

As we mentioned, existing QA approaches for attack trees work with static data values. Currently, if an organization wants to explore trends in the data values used in an attack tree, it needs to decouple visualization of security data values from the attack tree, for example, in a separate dashboard. Yet, this solution neglects the analyst's intuition behind the attack tree design, and could possibly hinder the detection of higher-level trends.

To the best of our knowledge, there are only few works studying attack tree visualizations. ADTool [18,10] and SecurITree [1] are tools for manual design and quantitative analysis of attack trees, but they do not offer extensive visualization capabilities besides showing static attack trees. Li et al. [21] have developed an approach to visualize complex scenarios with attack trees. Their work focuses on very large attack trees, helping the analyst to quickly identify interesting areas (e.g., the most probable attack scenarios) by highlighting them visually. Yet, this technique has been developed for static scenarios. It is thus the goal of our work to introduce an approach to model dynamic security scenarios with attack trees and to perform quantitative analysis of such scenarios.

Outside the attack tree literature, security visualization is a well-developed research area, and dynamic data visualisation techniques for security risks and attacks have been studied in, e.g., [20,33,24,28,12]. These studies on cybersecurity visualisation, as well as Li et al. [21], agree on the need to support the analyst by directing their attention to the *important* areas in the model. In the next section we outline our proposal for modelling dynamic scenarios with attack

trees enhanced with a novel type of attributes that capture the importance of data dynamics.

3 Attributes on attack-tree series

In this section we formally define an attack-tree series as a sequence of time-indexed attack trees. We also define the valuation of an attribute on an attack-tree series and consider a specific type of attribute, which we call *temperature*. Because the proposed extension is independent of the chosen attack tree semantics, we will provide an intuitive interpretation only.

3.1 Attack-tree series

We consider attack trees constructed from leaf nodes and two types of internal nodes (AND and OR). Following, e.g., [5], we will assume that all nodes of the attack tree are labeled.

Definition 1 (Attack tree). *Let $\mathcal{L}$ be a set of labels. An attack tree is an expression generated by the following grammar (for $\ell \in \mathcal{L}$):*

$$t ::= \ell \mid \mathtt{OR}(t, \ldots, t)_\ell \mid \mathtt{AND}(t, \ldots, t)_\ell.$$

We say that an attack tree has *unique labels*, if all labels occur at most once in that attack tree. In this paper we will only consider attack trees that have unique labels. By $labels(t)$ we denote the set of labels occurring in attack tree t.

An attack-tree series describes the evolution of an attack tree over time. In order to have a consistent view on the development of the individual nodes in the attack tree, we will define a consistency property.

Definition 2 (Consistent trees). *We say that two attack trees t and t' are* consistent *if the following three conditions are fulfilled:*

1. *The root nodes of t and t' have the same label.*
2. *If two non-root nodes of t and t' have the same label l_1, then their parent nodes must have the same label l_2.*
3. *If two non-leaf nodes of t and t' have the same label l_1, then they must have the same type (i.e. OR, resp. AND).*

A sequence (or set) of attack trees is consistent *if all its constituent attack trees are pairwise consistent.*

This notion of consistency guarantees that labels occurring in multiple incarnations of a developing attack tree always relate to the "same" node of the tree. Formulated differently, assume that we have two attack trees t and t' of which the root nodes have the same label, and assume that the intersection of the labels of these trees is L, then the subtree of t with labels from L is identical to the subtree of t' with labels from L.

Definition 3 (Attack-tree series). *Let Δ be a discrete time domain. An* attack-tree series $(t_\delta)_{\delta \in \Delta}$ *is a consistent sequence of attack trees indexed over* Δ.

An example of a consistent attack-tree series is shown in Fig. 1. Ignoring the attributes and temperature function (i.e. the numbers and colours), this figure shows an evolving attack tree at time points $\delta = 1, \ldots, 5$. Each attack tree in the series contains unique labels. It is easy to verify that every pair of attack trees in this series is consistent. For instance, the first two trees in the series are consistent because (1) their root nodes have the same label a, (2) corresponding non-root nodes in the two trees have parents with the same label (the parent of b in both trees is a), and (3) all non-leaf nodes with the same label have the same type (e.g. a is an `OR` node in both trees).

This figure also illustrates the rationale behind restricting Condition 3 in Definition 2 to non-leaf nodes. The reason for this restriction is that during the evolution of an attack tree an analyst may recognize substructure in an attack step that was first considered atomic. In the attack tree, this would mean that the node representing the atomic attack step evolves from a leaf node into an internal node of type `OR` or `AND`. As we want to relate the original leaf node to the new internal node, we only require corresponding types for non-leaf nodes, as expressed in Condition 3. This is illustrated by the first two attack trees in Fig. 1, where leaf node d changes into an internal node.

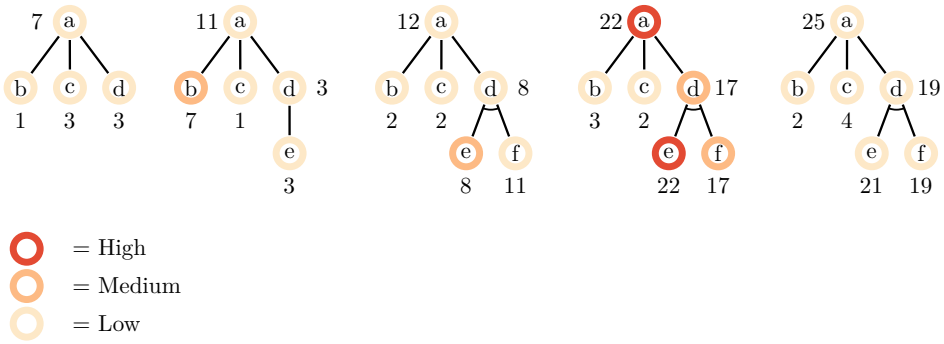

Fig. 1. An attack-tree series. The numbers indicate attribute γ and colours indicate temperature function τ.

3.2 Attributes

An attribute is an abstract notion expressing a property of interest concerning the attack tree or its nodes, such as *cost of attack* [31]. Given an attack-tree series $(t_\delta)_{\delta \in \Delta}$, an attribute is represented by a series of valuation functions

$\alpha_\delta \colon \mathit{labels}(t_\delta) \to D_\alpha$ for each time point $\delta \in \Delta$. The set D_α determines the range of values that can be taken by the attribute. We don't require that a valuation is a total function, thus allowing for valuations defined on a subset of a tree's nodes, such as the leaves.

There are various ways to determine the values of an attribute. Attributes can, for instance, be determined through observing events or by expert's opinions. Alternatively, an attribute can be completely determined by other attributes at the same time point. We call such an attribute a *derived* attribute. For instance, if we have an attribute *direct damage* and an attribute *reputation damage*, then the sum of these two defines the derived attribute *total damage*.

If, for any time point δ_0, an attribute depends on the values of other attributes for time points $\delta \leq \delta_0$ then we say that this attribute is *history-dependent*. By restricting this to a fixed prefix of δ_0 with size n, we define the subclass of n-*history-dependent* attributes. The class of derived attributes then corresponds to 0-history-dependent attributes.

For the temporal analysis of an attack-tree series we introduce the notion of *temperature*. Let Γ be a finite and totally ordered set, called the *temperature domain*. A *temperature* is a history-dependent attribute with range Γ. The intuition behind this notion is that nodes with a high temperature indicate that they are of interest to an analyst observing the development of the attack tree over time. Consider, for instance, an attribute γ counting the *observed number of occurrences of an attack*, then the difference between the current and previous value of γ could be used to highlight an increasing prevalence of a certain attack type. A relevant temperature in this case would be the 1-history dependent attribute τ defined by $\tau_\delta(\ell) = \gamma_\delta(\ell) - \gamma_{\delta-1}(\ell)$

Following these definitions, we can now define the challenge of a time-dependent analysis of a developing attack tree. Given an attack-tree series $(t_\delta)_{\delta \in \Delta}$, the challenge is to design one or more temperature attributes that relate to relevant security aspects of the system modeled by the attack-tree sequence, allowing the analyst to quickly observe security-related developments.

For an example of an attribute and a temperature function, we consider again Fig. 1. The attribute γ is defined through the values attributed to the nodes, e.g. $\gamma_3(a) = 12$. This particular attribute satisfies the property that conjunctive refinement is interpreted by the *min* function, while disjunctive refinement is interpreted by the *sum* function, e.g. $\gamma_3(a) = \gamma_3(b) + \gamma_3(c) + \gamma_3(d) = 2 + 2 + 8 = 12$.

We use colours of varying intensity to display the temperature values. The temperature attribute $\tau_\delta \colon \{a, b, c, d, e, f\} \to \{\mathit{High}, \mathit{Medium}, \mathit{Low}\}$ is defined by

$$\tau_\delta(\ell) = \begin{cases} \mathit{High} & \text{if } \gamma_{\delta-1} \text{ is defined and } \gamma_\delta(\ell) - \gamma_{\delta-1}(\ell) \in [10, \infty), \\ \mathit{Medium} & \text{if } \gamma_{\delta-1} \text{ is defined and } \gamma_\delta(\ell) - \gamma_{\delta-1}(\ell) \in [5, 10), \\ \mathit{Low} & \text{otherwise.} \end{cases}$$

The same attack-tree series with a different temperature attribute is shown in Fig. 2. The temperature attribute used in this case is 0-history dependent. It simply splits up the range of γ into four intervals. The advantage of the

previously defined temperature function τ over τ' is that it only triggers if there is a sudden increase of γ.

$$\tau'_\delta(\ell) = \begin{cases} \textit{Very high} & \text{if } \gamma_\delta(\ell) \in [15, \infty), \\ \textit{High} & \text{if } \gamma_\delta(\ell) \in [10, 15), \\ \textit{Low} & \text{if } \gamma_\delta(\ell) \in [5, 10), \\ \textit{Very low} & \text{otherwise.} \end{cases}$$

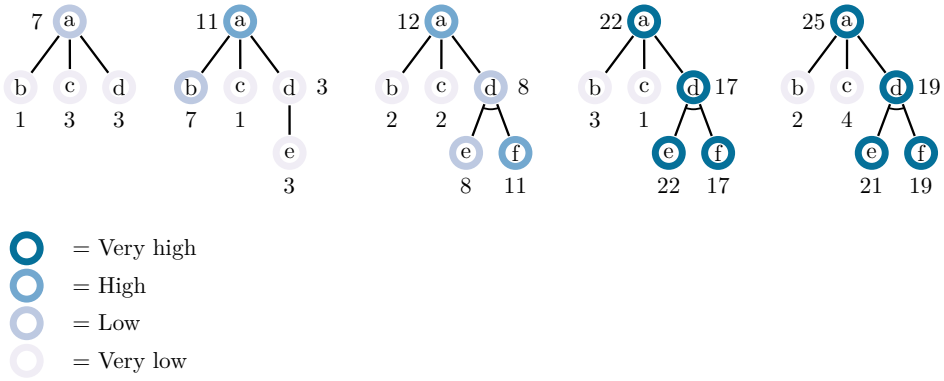

Fig. 2. An attack-tree series with the same attribute γ and a different temperature function τ'.

4 Usage scenarios

In this section we discuss application scenarios for attack-tree series.

4.1 Highlighting historical trends

The most straightforward usage scenario for the proposed model is to spot and highlight trends using historical data. Our approach allows analysts to consistently visualize changes in the data values on an evolving set of attack scenarios expressed in an attack-tree series. Particularly, the analyst may choose a *temperature* function that expresses how important a change in some data values is, by highlighting more drastic fluctuations. We demonstrate how this may be displayed in an intuitive manner using a small case study.

Case study description. We use part of the realistic attack tree from [7]. This attack tree captures different scenarios in automatic teller machines (ATMs). For our case study, we focus on the ATM fraud scenario. This part of the attack

tree contains 20 nodes. For the sake of simplicity, we limit this case study to dynamic data values, and we do not consider evolution of the tree structure.

We assume a company performing risk assessment for the ATM fraud scenario is interested in historical trends for three attributes: *probability*, *impact* (monetary loss) and *risk*, where risk is a derived attribute computed as the product *probability* $\times$ *impact*. The company uses historical data to evaluate these attributes. Probability of an attack can be evaluated from the frequency of such attacks over a given period of time, e.g.:

$$\mathsf{Pr}(\mathit{attack}) \approx \mathit{frequency}(\mathit{attack})/\mathit{total\ number\ of\ ATMs}$$

While frequency is an imperfect estimator, it expresses well historical trends, i.e., with an increased frequency of an attack, its probability also goes up, and vice-versa.

Impact of an attack is estimated as the maximal direct monetary loss stemming from this attack. To compute average impact over a year, the company sums the total amount of losses and then divides it by the number of attacks. Historical losses can vary over the years due to many factors, for example, the extent of an attack, purchased insurance, or a change in legislation.

The data for the ATM scenario can be acquired from industry-related catalogues. For example, the European Association for Secure Transactions (EAST) regularly shares with its members extended statistics on ATM attacks and incurred losses[1]. We have used historical data from the ATM Crime Report 2015, where we have found statistics on ATM fraud attacks in the period 2010–2015 in a selected European area. Since the attack tree we use is not fully mapped to the ATM Crime Report data, we only have statistics about frequencies and losses for 5 attack tree nodes out of 20: `ATM fraud`, `cash trapping`, `transaction reversal`, `card trapping`, and `card skimming`. We visualize historical trends on real data only for these nodes.

We show a simple 1-history dependent temperature function τ'' (where $om(x)$ denotes order of magnitude of x):

$$\tau''_\delta(\ell) = \begin{cases} \mathit{High} & \text{if } \gamma_{\delta-1} \text{ is defined and } \gamma_\delta(\ell) - \gamma_{\delta-1}(\ell) \in [3om(\gamma_{\delta-1}(\ell)), \infty), \\ \mathit{Medium} & \text{if } \gamma_{\delta-1} \text{ is defined and } \gamma_\delta(\ell) - \gamma_{\delta-1}(\ell) \in [om(\gamma_{\delta-1}(\ell)), 3om(\gamma_{\delta-1}(\ell))), \\ \mathit{Low} & \text{if } \gamma_{\delta-1} \text{ is undefined or } \gamma_\delta(\ell) - \gamma_{\delta-1}(\ell) \in [-om(\gamma_{\delta-1}(\ell)), om(\gamma_{\delta-1}(\ell))), \\ \mathit{Very\ low} & \text{if } \gamma_{\delta-1} \text{ is defined and } \gamma_\delta(\ell) - \gamma_{\delta-1}(\ell) \in [-\infty, -om(\gamma_{\delta-1}(\ell))). \end{cases}$$

This temperature function works for both probability and risk attributes. An animated demo of our visualisation approach is available online in our github repository[2].

Fig. 3 shows a snapshot of our ATM fraud visualization with temperature function τ'' for the risk attribute in 2012–2013. It visualizes that the risk of `card skimming`, while being the most important contributor to the overall `ATM fraud`

[1] `https://www.association-secure-transactions.eu/tag/atm-crime-report/`

[2] Visualizations and code are published at `https://github.com/vilena/atreeseries_viz`

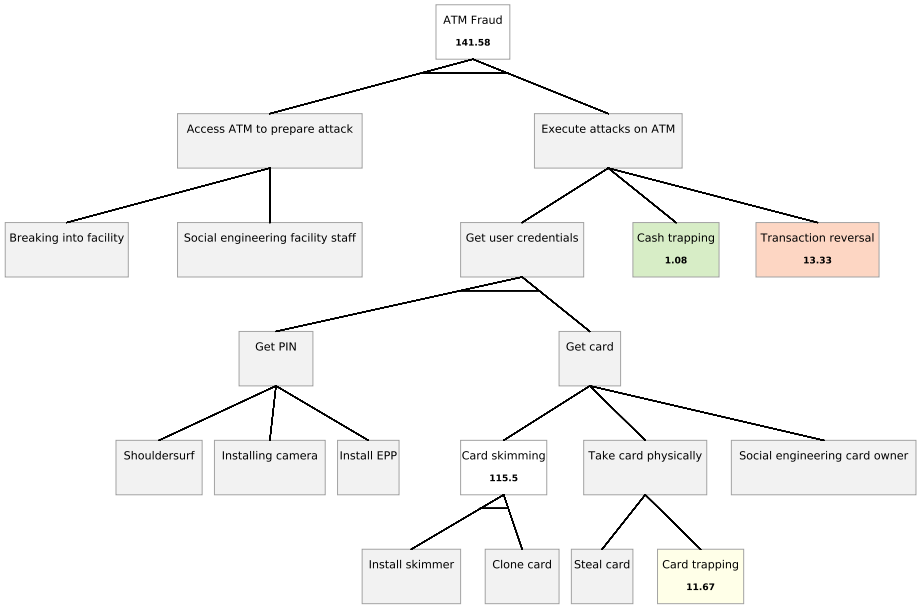

Fig. 3. A snapshot of the ATM fraud attack tree visualization on real data in 2013, with the temperature function τ'' for the *risk* attribute. The *high* value of the temperature function is shown in red, the *medium* value is shown in yellow, and the *very low* value is shown in green. The *low* values corresponding to very small changes in the attribute are shown as white nodes. We mark nodes for which we do not have data in grey.

risk, did not change much from the previous year. Risk of `cash trapping` has noticeably decreased in 2013, while risk of `card trapping` has increased. Finally, risk of `transaction reversal` has increased drastically from the previous year.

Due to the lack of complete historical data, we make another visualization using synthetic random data assigned to leaf nodes. By applying the bottom-up computation approach we complete the decoration process and obtain data for all intermediary nodes. For each year, for the probability values we generate random values in [0,1], and for the impact values we generate random integers in appropriate intervals (from [0, 100] for losses from shoulder-surfing, to [0, 10000] for losses from attacks involving damaging the ATM, such as installing skimmers and trappers). Then we apply the appropriate bottom-up computation rules for the probability and maximal cost to the defender attributes [17] to fully decorate the attack tree. Finally, we multiply each probability and impact data point to compute risk, and we visualize the trends on all attack tree nodes using the temperature function above. Animated visualizations with synthetic data are available online at our github repository.

The analyst looking at our visualization of historical trends for the ATM fraud scenario is able to identify the areas where the risk fluctuations are con-

centrated and review whether new countermeasures would be appropriate. However, this scenario also underlines the challenge to create informative temperature functions. Temperature function τ'' focuses on changes in a single node value. Thus, it may bring analyst attention to nodes where individual values change significantly, but they do not correspond to a major contribution to the value at the root node. In our example with the ATM Crime Report data, risk values at the `transaction reversal` node change considerably, due to the fact that it is a rare attack and it does not appear in the logs every year. Still, this attack is also not expensive for the ATM owner, as the risk value shows. The analyst may not want to investigate this type of attack, as it does not contribute significantly to the overall risk. Therefore, they may want to design a different temperature function, e.g., commesurable to the overall attribute value at the root node.

4.2 Other application scenarios

Spotting emerging trends. The visualization approach presented in the previous subsection can be integrated in a security dashboard to enable quick identification of emerging trends. For example, if the probability of a certain attack scenario goes up, the corresponding area in the attack tree will be highlighted, commanding attention of the security manager. This notification will highlight the emerging threat in its context (related attacks that may be enabled by the affected scenario).

Forecasting and trend extrapolation. Time-series data analysis is widely used to forecast future trends and extrapolate the ongoing dynamics [4]. Thus, our attack tree model naturally supports these usage scenarios for suitable data attributes. For instance, an organization may want to forecast, with regression techniques, how frequencies of social engineering attacks will change given the projected personnel growth rate. It may also want to extrapolate frequencies of probing attacks or cost of certain attack steps (e.g., cost of exploits or number of vulnerabilities in a software suite) given the dynamics observed locally or acquired from threat intelligence feeds.

What-if analysis. The proposed attack tree model lends itself very well to what-if analysis. For instance, the analyst may evaluate potential consequences of a risk management decision to avoid or accept some attack scenarios (by removing or, respectively, adding/keeping some branches of the tree). Our consistency notion allows to investigate these planned scenarios and perform quantitative analysis in a coherent manner.

The analyst may also consider advanced scenarios when many data values are modified simultaneously or in sequence. For example, they can simulate several possible attack data dynamics (cost of certain attack steps goes down or stays the same), evaluating how resilient the company is to adverse event developments.

Security investment analysis. Last but not least, one of the main goals of security risk assessment and threat modelling is to identify missing security controls and prioritize security investments. The *temperature* function applied to attack-tree series visualization focuses the analyst's attention on critical attack steps or areas of the attack tree that are affected by the data dynamics. The analyst may elaborate from this visualization the right abstraction or system level where defences need to be positioned. They may also modify the data values in the what-if analysis fashion to investigate whether reducing probability or affecting cost of an attack via positioning a security control strategically would reduce the risk to the desired level. Additionally, if a company wants to purchase cyber-insurance, defender's costs may be projected considering the envisaged premiums that will depend on the company size and assets involved.

5 Conclusions

In this paper we have developed the notion of attack-tree series that enable modeling, visualization and analysis of dynamic scenarios with attack trees. We have also defined the notion of temperature functions as a special type of quantitative attributes that capture the importance of changes in the data values. We have visualized these ideas on a case study with the ATM fraud attack tree.

This preliminary work extends the theory of attack trees towards the demands of security data analysis and visualization. We plan to extend it in several directions. Firstly, we will further develop the theory of attack-tree series and temperature functions for attack-defense trees, thereby explicitly integrating security controls and facilitating security investment analysis.

In our current development, a temperature function is a history-dependent attribute, meaning that it is based on the history of values of an attribute. This works well if the snapshots are taken at regular intervals. However, if the time elapsed between the various snapshots of an attack-tree series is not constant, the temperature function may give a distorted result. Therefore, a straightforward generalisation would be to allow the temperature function to not only depend on the previous attack trees, but also on their time points.

As exemplified by our case study on real data, data values may not be available for some leaf nodes. In such cases, the bottom-up computation technique is infeasible. Buldas et al. [5] have recently shown that intermediary data values can be used to complete the attack tree decoration process. Our current theory of attack-tree series with attributes, including temperature functions, is agnostic to the decoration process, as it only requires that all attack trees in the time-series are decorated. We plan to further develop a more general theory for quantitative problems on attack-tree series that will take into account data decoration algorithms and the types of data available in threat intelligence feeds (e.g., the number of infections).

Finally, we would like to develop a system for automatic attack-tree series design and visualization from threat intelligence feeds. Recent work [35,27,14,11,15] demonstrated the viability of generating attack trees automatically. Particularly,

Jhawar et al. [15] have shown that a threat library can be used to compose attack trees. A well-defined attack library like MITRE ATT&CK[3] could be used to automatically annotate information from threat feeds and add new relevant attack scenarios to an existing attack-tree series.

References

1. Amenaza. Securitree software, 2017.
2. Edward G Amoroso. *Fundamentals of computer security technology*. Prentice-Hall, Inc., 1994.
3. Alessandra Bagnato, Barbara Kordy, Per Håkon Meland, and Patrick Schweitzer. Attribute decoration of attack-defense trees. *Int. J. Secur. Softw. Eng.*, 3(2):1–35, April 2012.
4. George EP Box, Gwilym M Jenkins, Gregory C Reinsel, and Greta M Ljung. *Time series analysis: forecasting and control*. John Wiley & Sons, 2015.
5. Ahto Buldas, Olga Gadyatskaya, Aleksandr Lenin, Sjouke Mauw, and Rolando Trujillo-Rasua. Attribute evaluation on attack trees with incomplete information. *Computers & Security*, 2019. To appear.
6. Marlon Fraile, Margaret Ford, Olga Gadyatskaya, Rajesh Kumar, Mariëlle Stoelinga, and Rolando Trujillo-Rasua. Using attack-defense trees to analyze threats and countermeasures in an atm: a case study. In *IFIP Working Conference on The Practice of Enterprise Modeling*, pages 326–334. Springer, 2016.
7. Marlon Fraile, Margaret Ford, Olga Gadyatskaya, Rajesh Kumar, Mariëlle Stoelinga, and Rolando Trujillo-Rasua. Using attack-defense trees to analyze threats and countermeasures in an ATM: A case study. In *Proc. 9th IFIP Working Conference on the Practice of Enterprise Modeling (PoEM'16)*, Lecture Notes in Business Information Processing, pages 326–334. Springer, 2016.
8. Olga Gadyatskaya, René Rydhof Hansen, Kim Guldstrand Larsen, Axel Legay, Mads Chr. Olesen, and Danny Bøgsted Poulsen. Modelling attack-defense trees using timed automata. In *Proc. Int. Conf. on Formal Modeling and Analysis of Timed Systems (FORMATS'16)*, volume 9884 of *LNCS*, pages 35–50. Springer, 2016.
9. Olga Gadyatskaya, Carlo Harpes, Sjouke Mauw, Cédric Muller, and Steve Muller. Bridging two worlds: reconciling practical risk assessment methodologies with theory of attack trees. In *Proc. 3rd Int. Workshop on Graphical Models for Security (GraMSec'16)*, volume 9987 of *LNCS*, pages 80–93. Springer, 2016.
10. Olga Gadyatskaya, Ravi Jhawar, Piotr Kordy, Karim Lounis, Sjouke Mauw, and Rolando Trujillo-Rasua. Attack trees for practical security assessment: ranking of attack scenarios with ADTool 2.0. In *Proc. 13th Int. Conf. on Quantitative Evaluation of Systems (QEST'16)*, volume 9826 of *LNCS*, pages 159–162. Springer, 2016.
11. Olga Gadyatskaya, Ravi Jhawar, Sjouke Mauw, Rolando Trujillo-Rasua, and Tim Willemse. Refinement-aware generation of attack trees. In *Proc. 13th Workshop on Security and Trust Management (STM'17)*, volume 10547 of *LNCS*, pages 164–179, Oslo, Norway, 2017. Springer.
12. Jeffery Garae and Ryan K.L. Ko. Visualization and data provenance trends in decision support for cybersecurity. In *Data Analytics and Decision Support for Cybersecurity*, pages 243–270. Springer, 2017.

[3] `https://attack.mitre.org/`

13. I. Green. Extreme cyber scenario planning & attack tree analysis, 2013. Talk at RSA Conference `https://www.rsaconference.com/writable/presentations/file_upload/grc-t17.pdf`.
14. Marieta Georgieva Ivanova, Christian W. Probst, René Rydhof Hansen, and Florian Kammüller. Attack tree generation by policy invalidation. In *Proc. 9th IFIP Int. Conf. on Information Security Theory and Practice (WISTP'15)*, volume 9311 of *LNCS*, pages 249–259, Heraklion, Crete, Greece, 2015. Springer.
15. Ravi Jhawar, Karim Lounis, Sjouke Mauw, and Yunior Ramírez-Cruz. Semi-automatically augmenting Attack Trees using an annotated Attack Tree library. In *Proc. 14th Workshop on Security and Trust Management (STM'18)*, volume 11091 of *Lecture Notes in Computer Science*, pages 85–101, Barcelona, Spain, 2018. Springer-Verlag.
16. B. Kordy, S. Mauw, S. Radomirovic, and P. Schweitzer. Attack-defense trees. *J. Log. Comput.*, 24(1):55–87, 2014.
17. B. Kordy, S. Mauw, and P. Schweitzer. Quantitative questions on attack-defense trees. In *Proc. 15th Annual International Conference on Information Security and Cryptology (ICISC'12)*, volume 7839 of *LNCS*, pages 49–64, Seoul, South Korea, 2013. Springer.
18. Barbara Kordy, Piotr Kordy, Sjouke Mauw, and Patrick Schweitzer. ADTool: Security analysis with attack–defense trees. In *Proc. 10th Int. Conf. on Quantitative Evaluation of SysTems (QEST'13)*, volume 8054 of *LNCS*, pages 173–176, Buenos Aires, Argentina, 2013. Springer.
19. Rajesh Kumar and Mariëlle Stoelinga. Quantitative security and safety analysis with attack-fault trees. In *Proc. 18th Int. Symposium on High Assurance Systems Engineering (HASE'17)*, pages 25–32. IEEE, 2017.
20. Kiran Lakkaraju, William Yurcik, and Adam J. Lee. NVisionIP: netflow visualizations of system state for security situational awareness. In *Proc. 2004 ACM workshop on Visualization and data mining for computer security (VizSEC/DMSEC'04*, pages 65–72. ACM, 2004.
21. Eric Li, Jeroen Barendse, Frederic Brodbeck, and Axel Tanner. From A to Z: developing a visual vocabulary for information security threat visualisation. In *Proc. 3rd Int. Workshop on Graphical Models for Security (GraMSec'16)*, LNCS, pages 102–118, Lisbon, Portugal, 2016. Springer.
22. Xiaojing Liao, Kan Yuan, XiaoFeng Wang, Zhou Li, Luyi Xing, and Raheem Beyah. Acing the ioc game: Toward automatic discovery and analysis of open-source cyber threat intelligence. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 755–766. ACM, 2016.
23. Sjouke Mauw and Martijn Oostdijk. Foundations of attack trees. In *Proc. 8th Int. Conf. on Information Security and Cryptology (ICISC'05)*, volume 3935 of *LNSC*, pages 186–198, Seoul, South Korea, 2006. Springer.
24. Steven Noel, Eric Harley, Kam Him Tam, Michael Limiero, and Matthew Share. CyGraph: graph-based analytics and visualization for cybersecurity. In *Handbook of Statistics*, volume 35, pages 117–167. Elsevier, 2016.
25. S. Paul. Towards automating the construction & maintenance of attack trees: a feasibility study. In *Proc. 1st Int. Workshop on Graphical Models for Security (GraMSec'14)*, volume 148 of *EPTCS*, pages 31–46, Grenoble, France, 2014.
26. Stéphane Paul and Raphael Vignon-Davillier. Unifying traditional risk assessment approaches with attack trees. *Journal of Information Security and Applications*, 19(3):165 – 181, 2014.

27. Sophie Pinchinat, Mathieu Acher, and Didier Vojtisek. ATSyRa: an integrated environment for synthesizing attack trees. In *Proc. 2nd Int. Workshop on Graphical Models for Security (GraMSec'15)*, LNCS, pages 97–101, Verona, Italy, 2015. Springer.
28. Jamie Rasmussen, Kate Ehrlich, Steven Ross, Susanna Kirk, Daniel Gruen, and John Patterson. Nimble cybersecurity incident management through visualization and defensible recommendations. In *Proc. 7th int. symposium on visualization for cyber security(VizSec'10)*, pages 102–113. ACM, 2010.
29. Arpan Roy, Dong Seong Kim, and Kishor S. Trivedi. Attack countermeasure trees (ACT): towards unifying the constructs of attack and defense trees. *Security and Communication Networks*, 5(8):929–943, 2012.
30. Chris Salter, O. Sami Saydjari, Bruce Schneier, and Jim Wallner. Toward a secure system engineering methodology. In *Proc. 1998 Workshop on New Security Paradigms (NSPW'98)*, pages 2–10. ACM, 1998.
31. Bruce Schneier. Attack Trees: Modeling Security Threats. *Dr. Dobb's Journal of Software Tools*, 24(12):21–29, 1999.
32. Adam Shostack. *Threat modeling: Designing for security.* John Wiley & Sons, 2014.
33. Takeshi Takahashi, Keita Emura, Akira Kanaoka, Shin'ichiro Matsuo, and Tadashi Minowa. Risk visualization and alerting system: Architecture and proof-of-concept implementation. In *Proc. 1st int. workshop on Security in embedded systems and smartphones (SESP'13)*, pages 3–10. ACM, 2013.
34. Wiem Tounsi and Helmi Rais. A survey on technical threat intelligence in the age of sophisticated cyber attacks. *Computers & security*, 72:212–233, 2018.
35. Roberto Vigo, Flemming Nielson, and Hanne Riis Nielson. Automated generation of attack trees. In *Proc. 27th IEEE Computer Security Foundations Symposium (CSF'14)*, pages 337–350. IEEE, 2014.