Attack Trees with Sequential Conjunction*

Ravi Jhawar[1], Barbara Kordy[2], Sjouke Mauw[1], Saša Radomirović[3], Rolando Trujillo-Rasua[1]

[1] University of Luxembourg, SnT, Luxembourg
[2] INSA Rennes, IRISA, France
[3] Inst. of Information Security, Dept. of Computer Science, ETH Zürich, Switzerland

Abstract. We provide the first formal foundation of SAND attack trees which are a popular extension of the well-known attack trees. The SAND attack tree formalism increases the expressivity of attack trees by introducing the sequential conjunctive operator SAND. This operator enables the modeling of ordered events.
We give a semantics to SAND attack trees by interpreting them as sets of series-parallel graphs and propose a complete axiomatization of this semantics. We define normal forms for SAND attack trees and a term rewriting system which allows identification of semantically equivalent trees. Finally, we formalize how to quantitatively analyze SAND attack trees using attributes.

Keywords: Attack trees, security modeling, sequential operators, SAND

1 Introduction

Attack trees allow for an effective security analysis by systematically organizing the different ways in which a system can be attacked into a tree. The root node of an attack tree represents the *attacker's goal* and the children of a given node represent its refinement into *sub-goals*. A refinement is typically either disjunctive (denoted by OR) or conjunctive (denoted by AND). The leaves of an attack tree represent the attacker's actions and are called *basic actions*.

Since their inception by Schneier [26], attack trees have quickly become a popular modeling tool for security analysts. However, the limitations of this formalism, in particular with respect to expressing the order in which the various attack steps are executed, have been recognized by many authors (see e.g., [14]). In practice, modeling of security scenarios often requires constructs where conditions on the execution order of the attack components can be clearly specified. This is for instance the case when the time or (conditional) probability of an attack is considered, as in [2,29]. Consequently, several studies have extended attack trees informally with sequential conjunctive refinements. Such extensions have resulted in improved modeling and analyses (e.g., [29,21,30]) and software tools, e.g., ATSyRA [22].

* This is an extended version of [8].

Even though the sequential conjunctive refinement, that we denote by SAND, is well understood at a conceptual level and even applied to real world scenarios [22], none of the existing solutions have provided a rigorous mathematical formalization of attack trees with SAND. Indeed, the extensions found in the literature are rather diverse in terms of application domain, interpretation, and formality. Thereby, it is infeasible to answer fundamental questions such as: What is the precise expressibility of SAND attack trees? When do two such trees represent the same security scenario? Or what type of attributes can be synthesized on SAND attack trees in the standard bottom-up way? These questions can only be precisely answered if SAND attack trees are provided with a formal, general, and explicit interpretation, that is to say, if SAND attack trees are given a formal foundation.

Contributions: In this article we formalize the meaning of a SAND attack tree by defining its semantics. Our semantics is based on series-parallel (SP) graphs, which is a well-studied branch of graph theory. We provide a complete axiomatization for the SP semantics and show that the SP semantics for SAND attack trees are a conservative extension of the multiset semantics for standard attack trees [18] (i.e., our extension does not introduce unexpected equivalences w.r.t. the multiset semantics). To do so, we define a term rewriting system that is terminating and confluent and obtain normal forms for SAND attack trees. As a consequence, we achieve the rather surprising result that the domains of SAND attack trees and sets of SP graphs are isomorphic. We also extend the notion of attributes for SAND attack trees which enable the quantitative analysis of attack scenarios using the standard bottom-up evaluation algorithm.

One of the goals of our work is to provide a level of abstraction that encompasses most of the existing approaches from literature. For example, operators, such as the priority-based and the time-based connectors [28], are indeed captured by the SAND operator defined in this article. Moreover, other published semantics, such as those based on cumulative distribution functions [2], conditional probabilities [28], or boolean algebra [10], can be expressed as an attribute in our formalism. Last but not least, even though we make the distinction between AND and SAND refinements explicit, our semantics satisfies backward compatibility with the well-known multiset semantics of attack trees [18]. This stresses the, much needed, unifying character of our approach.

Organization: Section 2 summarizes the related work and puts our work in context. Section 3 provides a formal definition of SAND attack trees and its semantics using series-parallel graphs. Section 4 defines a complete set of axioms for SAND attack trees and presents a term rewriting system which allows identification of semantically equivalent SAND attack trees. Section 5 outlines an approach to quantitatively analyze SAND attack trees using attributes. Finally, Section 6 concludes with an outlook on future work.

2 Related Work and Motivation

Several extensions of attack trees with temporal or causal dependencies between attack steps have been proposed. We observe that there are three di□erent approaches to achieve this goal. The first approach is to use standard attack trees with the added assumption that the children of an AND node are sequentially ordered. This approach is mostly applied to the design of algorithms or tools for the analysis of attack trees under the assumption of ordered events.

The second approach is to introduce a mechanism for ordering events in an attack tree, for instance by adding a new type of edge to express causality or conditionality. In its most general case, any partial order on the events in an attack tree can be specified. The third approach consists of the introduction of a new type of node for sequencing. Most extensions fall in this category. This approach is used by authors who require their formalism to be backward compatible, or who need standard, as well as ordered conjunction. We discuss for each of these approaches the most relevant papers with respect to the present article. That is, we only consider approaches that still have the main characteristics of attack trees, being the presence of AND and OR nodes and the interpretation of the edges as a refinement relation. Thus, we consider approaches such as attack graphs [27,19] and Bayesian networks [23,9,17] as out of scope for this paper.

Approaches with a sequential interpretation of AND. In their work on *Bayesian networks for security*, Qin and Lee define a transformation from attack trees to Bayesian networks [24]. They state that "there always exists an implicit dependent and sequential relationship between AND nodes in an attack tree." Most literature on attack trees seem to contradict this statement, implying that there is a need to explicitly identify such sequential relationships.

Jürgenson and Willemson developed an algorithm to calculate the *expected outcome* of an attack tree [30]. The goal of the algorithm is to determine a permutation of leaves for which the optimal expected outcome for an attacker can be achieved. In essence, their input is an attack tree where an AND node represents all possible sequences of its children. A peculiarity of their interpretation is that multiple occurrences of the same node are considered only once, implying that the execution of twice the same action cannot be expressed.

Approaches introducing a general order. Peine, Jawurek, and Mandel introduce *security goal indicator trees* [20] in which nodes can be related by a notion of *conditional dependency* and Boolean connectors. The authors, however, do not formally specify the syntax and semantics of the model. A more general approach is proposed by Piètre-Cambacédès and Bouissou [21], who apply *Boolean logic driven Markov processes* to security modeling. Their formalism does not introduce new gates, but a (trigger-)relation on the nodes of the attack tree. Although triggers can express a more general sequential relation than the SAND operator, they lack the readability of standard attack tree operators.

Vulnerability cause graphs [1,4] combine properties of attack trees (AND and OR nodes) and attack graphs (edges express order rather than refinement). The interaction between the AND nodes and the order relation is defined through a graph transformation called *conversion of conjunctions*, which ignores the order between nodes. This discrepancy could be solved by considering distinct conjunctive and sequential conjunctive nodes, as we do in this paper.

Approaches introducing sequential AND. As noted by Arnold et al. [2], the analysis of time-dependent attacks requires attack trees to be extended with a sequential operator. This is accomplished by defining sequential nodes as conjunctive nodes with a notion of progress of time. The authors define a formal semantics for this extension based on cumulative distribution functions (CDFs), where a CDF denotes the probability that a successful attack occurs within time t. The main difference with our work is that their approach is based on an explicit notion of time, while we have a more abstract approach based on causality. In their semantics, the meaning of an extended attack tree is a CDF, in which the relation to the individual basic attacks is not explicit anymore. In contrast, in our semantics the individual basic attacks and their causal ordering remain visible. As such, our semantics can be considered more abstract, and indeed, we can formulate their semantics as an *attribute* in our approach.

Enhanced attack trees [5] (EATs) distinguish between OR, AND and OAND (Ordered AND). Similarly to the approach of Arnold et al. [2], ordered AND nodes are used to express temporal dependencies between attack components. The authors evaluate EATs by transforming them into tree automata. Intermediate states in the automaton support the task of reporting partial attacks. However, because every intermediate node of the tree corresponds to a state in the tree automaton, their approach does not scale well. This problem can be addressed by considering the normal form of attack trees, as proposed in this article.

Not every extension of attack trees with SAND refinements concerns time-dependent attack scenarios; some aim at supporting risk analyses with conditional probabilities. For that purpose, Wen-Ping and Wei-Min introduce *improved attack trees* [29]. The concepts, however, are described at an intuitive level only.

Unified parameterizable attack trees [28] unify different extensions of attack trees (structural, computational, and hybrid). The authors consider two types of ordered AND connectors: priority-based connectors and time-based connectors. The children of the former are ordered from highest to lowest priority, whereas the children of the latter are ordered temporally. Our formalism gives a single interpretation to the SAND operator, yet it can capture both connectors.

Due to obvious similarities, we also review approaches that introduce the SAND operator in fault trees. For example, Brooke and Paige include five fault tree gates: AND, OR, priority AND, exclusive OR, and an inhibit gate [3]. The authors do not discuss the semantics of their model for security, though. Another fault tree based approach is discussed by Khand [10], who proposes to extend attack trees with a set of gates from dynamical fault tree modeling that overlaps with

the gates used by Brooke and Paige [3] and in particular contains the priority AND gate. Khand assigns truth values to his attack trees by giving truth tables for all gates. Khand's truth tables, when restricted to AND, OR, and priority AND, constitute an attribute domain which is compatible (in the sense of [13]) with the SP semantics for SAND attack trees as defined in this paper.

We observe that the extensions of attack trees with sequential conjunction are rather diverse in terms of application domain, interpretation, and formality. In order to give a clear and unambiguous interpretation of the SAND operator and capture different application domains, it is necessary to give a formal semantics as a translation to a well-understood domain. Note that, neither the multiset [18] nor the propositional semantics [15] can express ordering of attack components. Therefore, a richer semantical domain needs to be defined. The purpose of this article is to address this problem.

3 Attack Trees with Sequential Conjunction

We extend the attack tree formalism so that a refinement of a (sub-)goal of an attacker can be a sequential conjunct (denoted by SAND) in addition to disjuncts and conjuncts. We first give a definition of attack trees with the new sequential operator and then define series-parallel graphs on which the semantics for the new attack trees is based.

3.1 SAND Attack Trees

Let $\mathbb{B}$ denote the set of all possible basic actions of an attacker. We formalize standard attack trees introduced by Schneier in [26] and call them simply *attack trees* in the rest of this paper. Attack trees are closed terms over the signature $\mathbb{B} \cup \{\mathtt{OR}, \mathtt{AND}\}$, generated by the following grammar, where $b \in \mathbb{B}$ is a terminal symbol.

$$t ::= b \mid \mathtt{OR}(t, \ldots, t) \mid \mathtt{AND}(t, \ldots, t) \tag{1}$$

The universe of attack trees is denoted by $\mathbb{T}$. SAND *attack trees* are closed terms over the signature $\mathbb{B} \cup \{\mathtt{OR}, \mathtt{AND}, \mathtt{SAND}\}$, where SAND is a non-commutative operator called *sequential conjunction*, and are generated by the grammar

$$t ::= b \mid \mathtt{OR}(t, \ldots, t) \mid \mathtt{AND}(t, \ldots, t) \mid \mathtt{SAND}(t, \ldots, t) \tag{2}$$

The universe of SAND attack trees is denoted by $\mathbb{T}_{\mathtt{SAND}}$. The purpose of OR and AND refinements in SAND attack trees is the same as in attack trees. The sequential conjunctive refinement SAND allows us to model that a certain goal is reached if and only if all its subgoals are reached in a precise order.

The following attack scenario motivates the need for extending attack trees with sequential conjunctive refinement.

Example 1. Consider a file server S, offering ftp, ssh, and rsh services. The attack tree in Figure 1 shows how an attacker can gain root privileges on

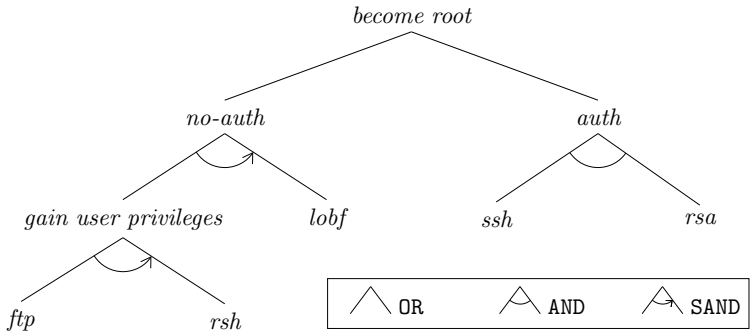

Fig. 1. An attack tree with sequential and parallel conjunctions

S (*become root*), in two ways: either without providing any user credentials (*no-auth*) or by breaching the authentication mechanism (*auth*).

In the first case, the attacker must first gain user privileges on S (*gain user privileges*) and then perform a local buffer overflow attack (*lobf*). Since the attack steps must be executed in this particular order, the use of SAND refinement is substantial. To gain user privileges, the attacker must exploit an ftp vulnerability to anonymously upload a list of trusted hosts to S (*ftp*).[4] Finally, she can use the new trust condition to remotely execute shell commands on S (*rsh*).

The second way is to abuse a buffer overflow in both the ssh daemon (*ssh*) and the RSAREF2 library (*rsa*) used for authentication. These attacks can be executed in any order, which is modeled with the standard AND refinement.

Using the term notation introduced in this section, we can represent the SAND attack tree in Figure 1 as

$$t = \mathtt{OR}\Big(\mathtt{SAND}\big(\mathtt{SAND}(\mathit{ftp}; \mathit{rsh}); \mathit{lobf}\big); \mathtt{AND}(\mathit{ssh}; \mathit{rsa})\Big);$$

where $\mathit{ftp}; \mathit{rsh}; \mathit{lobf}; \mathit{ssh}; \mathit{rsa} \in \mathbb{B}$ are basic actions.

3.2 Series-Parallel Graphs

A *series-parallel graph* (SP graph) is an edge-labeled directed graph that has two unique, distinct vertices, called *source* and *sink*, and that can be constructed with the two operators for sequential and parallel composition of graphs that we formally define below. A source is a vertex which has no incoming edges and a sink is a vertex without outgoing edges.

Our formal definition of SP graphs is based on *multisets*, i.e., sets in which members are allowed to occur more than once. We use $\{\!\{\cdot\}\!\}$ to denote multisets and $\mathcal{P}(\cdot)$ to denote powersets. The *support* $M^\star$ of a multiset M is the set of distinct elements in M. For instance, the support of the multiset $M = \{\!\{b_1; b_2; b_2\}\!\}$ is $M^\star = \{b_1; b_2\}$.

[4] For readability, attack actions are named after the services that are exploited.

In order to define SP graphs, we first introduce the notion of source-sink graphs labeled by the elements of $\mathbb{B}$.

Definition 1. *A* source-sink graph *over $\mathbb{B}$ is a tuple $G = (V; E; s; z)$, where V is the set of vertices, E is a multiset of labeled edges with support $E^\star \subseteq V \times \mathbb{B} \times V$, $s \in V$ is the unique source, $z \in V$ is the unique sink, and $s \neq z$.*

The sequential composition of a source-sink graph $G = (V; E; s; z)$ with a source-sink graph $G' = (V'; E'; s'; z')$, denoted by $G \cdot G'$, is the graph resulting from taking the disjoint union of G and G' and identifying the sink of G with the source of G'. More precisely, let $\dot\cup$ denote the disjoint union operator and $E^{[s/z]}$ denote the multiset of edges in E, where all occurrences of vertex z are replaced by vertex s. Then we define

$$G \cdot G' = (V \setminus \{z\} \dot\cup V'; E^{[s'/z]} \dot\cup E'; s; z').$$

The parallel composition, denoted by $G \parallel G'$, is defined similarly, except that the two sources are identified and the two sinks are identified. Formally, we have

$$G \parallel G' = (V \setminus \{s; z\} \dot\cup V'; E^{[s'/s, z'/z]} \dot\cup E'; s'; z').$$

It follows directly from the definitions that the sequential composition is associative and that the parallel composition is associative and commutative.

We write $\xrightarrow{b}$ for the graph with a single edge labeled with b and define SP graphs as follows.

Definition 2. *The set $\mathbb{G}_{SP}$ of* series-parallel graphs *(SP graphs) over $\mathbb{B}$ is defined inductively by the following two rules*

- *For $b \in \mathbb{B}$, $\xrightarrow{b}$ is an SP graph.*
- *If G and G' are SP graphs, then so are $G \cdot G'$ and $G \parallel G'$.*

It follows directly from Definition 2 that SP graphs are connected and acyclic. Moreover, every vertex of an SP graph lies on a path from the source to the sink. We consider two SP graphs to be *equal* if there is a bijection between their sets of vertices that preserves the edges and edge labels.

Example 2. Figure 2 shows an example of an SP graph with the source s and the sink z. This graph corresponds to the construction

$$\left(\xrightarrow{a} \parallel \xrightarrow{b} \parallel \xrightarrow{b} \right) \cdot \xrightarrow{c} \cdot \left(\left(\xrightarrow{d} \cdot (\xrightarrow{e} \parallel \xrightarrow{f}) \right) \parallel \xrightarrow{g} \right).$$

3.3 SP Semantics for SAND Attack Trees

Numerous semantics have been proposed to interpret attack trees, including propositional logic [16], multisets [18], De Morgan lattices [15], tree automata [5], and Markov processes [21,2]. The choice of a semantics allows us to accurately

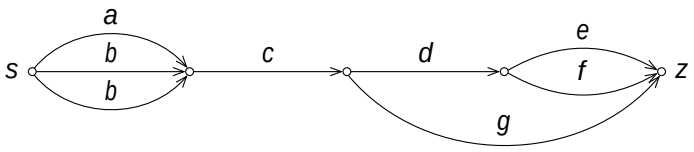

Fig. 2. A series-parallel graph

represent the assumptions made in a security scenario, e.g., whether actions can be repeated or resources reused, and to decide which trees represent the same security scenario. The advantages of formalizing attack trees and the need for various semantics have been discussed in [13]. Since attack trees are AND/OR trees, the most natural interpretation is based on propositional logic. However, because the logical operators are idempotent, this interpretation assumes that the multiplicity of an action is irrelevant. As a consequence, the propositional semantics is not well suited to reason about scenarios with multiple occurrences of the same action. Due to this lack of expressivity a semantics was proposed [18] in which the multiplicity of actions is taken into account. This was achieved by interpreting an attack tree as a set of multisets that represent different ways of reaching the root goal. This multiset semantics is compatible with computations that depend on the number of occurrences of an action in the tree, such as the minimal time to carry out the attack represented by the root goal.

We now extend the multiset semantics to SAND attack trees. Since SP graphs naturally extend multisets with a partial order, they supply a formalism in which we can interpret trees using both — commutative and sequential — conjunctive refinements. SP graphs therefore provide a canonical semantics for SAND trees in which multiplicity and ordering of goals and actions are significant. The idea is to interpret an attack tree t as a set of SP graphs. The semantics $[\![t]\!]_{\mathcal{SP}} = \{G_1, \ldots, G_k\}$ of a tree t corresponds to the set of possible attacks G_i, where each attack is described by an SP graph labeled by the basic actions of t.

Definition 3. *The* SP semantics *for* SAND *attack trees is given by the function* $[\![\cdot]\!]_{\mathcal{SP}} : \mathbb{T}_{\mathtt{SAND}} \to \mathcal{P}(\mathbb{G}_{\mathcal{SP}})$*, which is defined recursively as follows: for* $b \in \mathbb{B}$*,* $t_i \in \mathbb{T}_{\mathtt{SAND}}$*,* $1 \leq i \leq k$*,*

$$
\begin{aligned}
&[\![b]\!]_{\mathcal{SP}} = \{\xrightarrow{b}\} \\
&[\![\mathtt{OR}(t_1, \ldots, t_k)]\!]_{\mathcal{SP}} = \textstyle\bigcup_{i=1}^{k} [\![t_i]\!]_{\mathcal{SP}} \\
&[\![\mathtt{AND}(t_1, \ldots, t_k)]\!]_{\mathcal{SP}} = \{G_1 \parallel \cdots \parallel G_k \mid (G_1, \ldots, G_k) \in [\![t_1]\!]_{\mathcal{SP}} \times \cdots \times [\![t_k]\!]_{\mathcal{SP}}\} \\
&[\![\mathtt{SAND}(t_1, \ldots, t_k)]\!]_{\mathcal{SP}} = \{G_1 \cdot \ldots \cdot G_k \mid (G_1, \ldots, G_k) \in [\![t_1]\!]_{\mathcal{SP}} \times \cdots \times [\![t_k]\!]_{\mathcal{SP}}\}.
\end{aligned}
$$

The SP semantics maps SAND attack trees to sets of SP graphs as follows. A leaf corresponding to a basic action b is translated into a singleton set containing the SP graph which consists of a single edge labeled with b. The semantics of a disjunctive node is the set of all the alternative attacks described by the

node's children. The semantics of a conjunctive node is the parallel composition of every attack alternative from each of its children. Finally, the semantics of a sequential conjunctive node is a sequential composition of attack alternatives for the children.

Example 3. The SP semantics of the attack tree t depicted in Figure 1 is

$$[\![t]\!]_{\mathcal{SP}} = \{\xrightarrow{ftp}\xrightarrow{rsh}\xrightarrow{lobf} \; ; \; \xrightarrow{ssh}\|\xrightarrow{rsa}\}.$$

As shown in Example 3, the SP semantics provides an alternative graph representation for attack trees and therefore contributes a different perspective on an attack scenario. The SAND attack tree emphasizes the refinement of goals, whereas SP graphs highlight the sequential aspect of attacks.

The SP semantics provides a natural partition of $\mathbb{T}_{\mathtt{SAND}}$ into equivalence classes.

Definition 4. *Two* SAND *attack trees* t_1 *and* t_2 *are* equivalent with respect to the SP semantics *if and only if they are interpreted by the same set of SP graphs, i.e.,* $[\![t_1]\!]_{\mathcal{SP}} = [\![t_2]\!]_{\mathcal{SP}}$.

By Definition 4, if the SP semantics provides accurate assumptions for an attack scenario, then two SAND attack trees represent the same attack scenario if and only if they are equivalent with respect to the SP semantics.

We finish this section by noticing that in the case of SAND attack trees without any SAND refinement, the SP semantics coincides with the multiset semantics introduced in [18]. Indeed, it suffices to identify the multiset $\{\!|b_1,\ldots,b_k|\!\}$ with the SP graph $\xrightarrow{b_1}\| \ldots \|\xrightarrow{b_k}$. We discuss this issue more in details in Section 4.3.

4 Axiomatization of the SP Semantics

In order to provide efficient analysis methods for attack tree-like models, we need to be able to decide whether two trees are equivalent with respect to a given semantics. Ideally, we would like to find the most efficient (e.g., the smallest) representation of a given security scenario. However, in the case of the SAND semantics, there exists an infinite number of trees t' equivalent to a given tree t.

In this section we study the mathematical implications of using sets of SP graphs as an interpretation domain for SAND attack trees. We introduce an axiomatization of SAND attack trees which is complete with respect to the SP semantics. This allows us to reason directly on SAND attack trees, without having to move to the semantical domain. Further, we derive a term rewriting system from the axiomatization as a means to effectively decide whether two SAND attack trees are equivalent with respect to the SP semantics. As a consequence, we obtain a canonical representation of SAND attack trees which we prove to be isomorphic to sets of SP graphs.

4.1 A complete set of axioms for the SP semantics

Let $\mathbb{V}$ be a set of variables denoted by capital letters. Following the approach developed in [13], we axiomatize SAND attack trees with equations $l = r$, where l and r are terms over variables in $\mathbb{V}$, constants in $\mathbb{B}$, and the operators AND, OR, and SAND. The equations formalize the intended properties of refinements and provide semantics-preserving transformations of SAND attack trees.

Example 4. Let Sym_ℓ denote the set of all bijections from $\{1;:::;`\}$ to itself. The axiom

$$\mathtt{AND}(Y_1;:::;Y_\ell) = \mathtt{AND}(Y_{\sigma(1)};:::;Y_{\sigma(\ell)})$$

expresses that the order between children refining a parallel conjunctive node is not relevant. In other words, the operator AND is commutative. This implies that any two trees of the form $\mathtt{AND}(t_1;:::;t_l)$ and $\mathtt{AND}(t_{\sigma(1)};:::;t_{\sigma(l)})$ represent the same scenario.

Our goal is to define a complete set of axioms, denoted by $E_{\mathcal{SP}}$, for the SP semantics for SAND attack trees. Intuitively, $E_{\mathcal{SP}}$ is a set of equations that can be applied to transform a SAND attack tree into any equivalent SAND attack tree with respect to the SP semantics. Before defining the set $E_{\mathcal{SP}}$, we formalize the notion of a complete set of axioms for a given semantics for (SAND) attack trees, following [13].

Let $T(\mathbb{V};\ \ )$ be the free term algebra over the set of variables $\mathbb{V}$ and a signature $\ \ $, and let E be a set of equations over $T(\mathbb{V};\ \ )$. The equation $t = t'$, where $t; t' \in T(\mathbb{V};\ \ )$, is a *syntactic consequence* of E (denoted by $E \vdash t = t'$) if it can be derived from E by application of the following rules. For all $t; t'; t'' \in T(\mathbb{V};\ \ )$, $\ : \mathbb{V} \to T(\mathbb{V};\ \ )$, and $X \in \mathbb{V}$:

- $E \vdash t = t$,
- if $t = t' \in E$, then $E \vdash t = t'$,
- if $E \vdash t = t'$, then $E \vdash t' = t$,
- if $E \vdash t = t'$ and $E \vdash t' = t''$, then $E \vdash t = t''$.
- if $E \vdash t = t'$, then $E \vdash \ (t) = \ (t')$,
- if $E \vdash t = t'$, then $E \vdash t''[t{=}X] = t''[t'{=}X]$, where $t''[t{=}X]$ is the term obtained from t'' by replacing all occurrences of variable X with t.

Let $\mathbb{T}^{\mathbb{V}}_{\mathtt{SAND}}$ denote the set of terms constructed from the set of variables $\mathbb{V}$, the set of basic actions $\mathbb{B}$ (treated as constants), and operators OR, AND and SAND. Let $\mathbb{T}^{\mathbb{V}}$ be the set of terms constructed from the same parts, except for the operator SAND. Using the notion of syntactic consequence, we define a complete set of axioms for a semantics for attack trees.

Definition 5. *Let $[\![\cdot]\!]$ be a semantics for attack trees (resp.* SAND *attack trees) and let E be a set of equations over $\mathbb{T}^{\mathbb{V}}$ (resp. $\mathbb{T}^{\mathbb{V}}_{\mathtt{SAND}}$). The set E is a* complete set of axioms *for $[\![\cdot]\!]$ if and only if, for all $t; t' \in \mathbb{T}$ (resp. $\mathbb{T}_{\mathtt{SAND}}$)*

$$[\![t]\!] = [\![t']\!] \iff E \vdash t = t':$$

We are now ready to give a complete set of axioms for the SP semantics for SAND attack trees. These axioms allow us to determine whether two visually distinct trees represent the same security scenario according to the SP semantics.

Theorem 1. *Given $k; m \geq 0$, and $\grave{} \geq 1$, let $\overline{X} = X_1; \ldots; X_k$, $\overline{Y} = Y_1; \ldots; Y_\ell$, and $\overline{Z} = Z_1; \ldots; Z_m$ be sequences of variables. Let Sym_ℓ be the set of all bijections from $\{1; \ldots; \grave{}\}$ to itself. The following set of equations over $\mathbb{T}^{\mathbb{V}}_{\mathtt{SAND}}$, denoted by $E_{\mathcal{SP}}$, is a complete set of axioms[5] for the SP semantics for* SAND *attack trees.*

$$\mathtt{OR}(Y_1; \ldots; Y_\ell) = \mathtt{OR}(Y_{\sigma(1)}; \ldots; Y_{\sigma(\ell)}); \quad \forall \quad \in \mathrm{Sym}_\ell \tag{E_1}$$

$$\mathtt{AND}(Y_1; \ldots; Y_\ell) = \mathtt{AND}(Y_{\sigma(1)}; \ldots; Y_{\sigma(\ell)}); \quad \forall \quad \in \mathrm{Sym}_\ell \tag{E_2}$$

$$\mathtt{OR}\big(\overline{X}; \mathtt{OR}(\overline{Y})\big) = \mathtt{OR}(\overline{X}; \overline{Y}) \tag{E_3}$$

$$\mathtt{AND}\big(\overline{X}; \mathtt{AND}(\overline{Y})\big) = \mathtt{AND}(\overline{X}; \overline{Y}) \tag{E_4}$$

$$\mathtt{SAND}\big(\overline{X}; \mathtt{SAND}(\overline{Y}); \overline{Z}\big) = \mathtt{SAND}(\overline{X}; \overline{Y}; \overline{Z}) \tag{$E_{4'}$}$$

$$\mathtt{OR}(A) = A \tag{E_5}$$

$$\mathtt{AND}(A) = A \tag{E_6}$$

$$\mathtt{SAND}(A) = A \tag{$E_{6'}$}$$

$$\mathtt{AND}\big(\overline{X}; \mathtt{OR}(\overline{Y})\big) = \mathtt{OR}\big(\mathtt{AND}(\overline{X}; Y_1); \ldots; \mathtt{AND}(\overline{X}; Y_\ell)\big) \tag{E_{10}}$$

$$\mathtt{SAND}\big(\overline{X}; \mathtt{OR}(\overline{Y}); \overline{Z}\big) = \mathtt{OR}\big(\mathtt{SAND}(\overline{X}; Y_1; \overline{Z}); \ldots; \mathtt{SAND}(\overline{X}; Y_\ell; \overline{Z})\big) \tag{$E_{10'}$}$$

$$\mathtt{OR}(A; A; \overline{X}) = \mathtt{OR}(A; \overline{X}): \tag{E_{11}}$$

The numbering of the axioms in $E_{\mathcal{SP}}$ corresponds to the numbering of the axioms for the multiset semantics for standard attack trees, as presented in [13], while new axioms (involving SAND) are marked with primes.

Proof. The proof of this theorem follows the same line of reasoning as the proofs of Theorems 4.2 and 4.3 of Gischer [7], where series–parallel pomsets are axiomatized. To prove the theorem, we remark that SP graphs form a visual representation of series-parallel partially ordered multisets (SP pomsets). A complete, finite axiomatization of pomsets under concatenation, parallel composition and union has been provided in [7], where sets of series-parallel pomsets have been used to represent processes. In our case, sets of series-parallel pomsets (i.e., sets of SP graphs) represent attack trees constructed using AND (having the same properties as the parallel composition of processes), SAND (having the same properties as concatenation), and OR (having the same properties as choice). The set $E_{\mathcal{SP}}$ corresponds to the axioms from [7]. The axioms involving the identity elements (i.e., 1 – the empty pomset, and 0 – the empty process) have been omitted because they can only be used for transforming processes involving 0 or 1 and such identity elements do not exist in the case of attack trees. Furthermore, our axioms are written using unranked operators contrary to the binary operators of concatenation, parallel composition, and choice.

□

[5] Note that the axioms are in fact *axiom schemes*. The operators OR, AND and SAND are *unranked*, representing infinitely many k-ary function symbols ($k \geq 1$).

4.2 SAND Attack Trees in Canonical Form

Let $[\![\cdot]\!]$ be a semantics for (SAND) attack trees. A complete axiomatization of $[\![\cdot]\!]$ can be used to derive a canonical form of trees interpreted with $[\![\cdot]\!]$. Such canonical forms provide the most concise representation for equivalent trees and are the natural representatives of equivalence classes defined by $[\![\cdot]\!]$.

When SAND attack trees are interpreted using the SP semantics, their canonical forms consist of either a single basic action, or of a root node labeled with OR and subtrees with nested, alternating occurrences of AND and SAND nodes. Canonical forms correspond exactly to the sets of SP graphs labeled by $\mathbb{B}$ and they depict all attack alternatives in a straightforward way.

Canonical representations of SAND attack trees under the SP semantics can be defined using the complete set of axioms $E_{\mathcal{SP}}$. By orienting the equations (E_3), (E_4), ($E_{4'}$), (E_5), (E_6), ($E_{6'}$), (E_{10}), ($E_{10'}$), and (E_{11}) from left to right, we obtain a term rewriting system, denoted by $R_{\mathcal{SP}}$. The canonical representations of SAND attack trees correspond to normal forms with respect to $R_{\mathcal{SP}}$. In the rest of this section we show that the normal forms with respect to $R_{\mathcal{SP}}$ are exactly the terms generated by the following grammar, where $k \geq 2$ and $b \in \mathbb{B}$

$$\begin{aligned} N ::= &\quad C \mid \texttt{OR}(C_1, \ldots, C_k) \quad \text{for } C_i \neq C_j \text{ if } i \neq j \\ C ::= &\quad A \mid S \\ A ::= &\quad b \mid \texttt{AND}(S_1, \ldots, S_k) \\ S ::= &\quad b \mid \texttt{SAND}(A_1, \ldots, A_k). \end{aligned} \tag{3}$$

The non-terminal A produces all trees that consist of a single basic action or being a nested alternation of AND and SAND operators, where the outer operator is AND. Similarly, S produces all such trees where the outer operator is SAND. The non-terminal C generates the two previously described types of trees. Finally, N combines the trees generated by C using the OR refinement. We denote the sets of terms generated by N, C, A, and S, by $\mathbb{T}_N$, $\mathbb{T}_C$, $\mathbb{T}_A$, and $\mathbb{T}_S$, respectively.

We first observe that the terms generated by the non-terminal N correspond exactly to all sets of SP graphs labeled by the elements of $\mathbb{B}$.

Lemma 1. *The restriction of function $[\![\cdot]\!]_{\mathcal{SP}}$ to $\mathbb{T}_N$ is a bijection from $\mathbb{T}_N$ to $\mathcal{P}(\mathbb{G}_{\mathcal{SP}})$.*

Proof. The proof consists of two steps. First we prove that the terms from $\mathbb{T}_C$ exactly correspond to SP graphs, after which we extend this result to the correspondence between $\mathbb{T}_N$ and the sets of SP graphs.

1. $[\![\cdot]\!]_{\mathcal{SP}}$ is a bijection from $\mathbb{T}_C$ to $\mathbb{G}_{\mathcal{SP}}$.
   For injectivity we prove by induction that $[\![C_1]\!]_{\mathcal{SP}} = [\![C_2]\!]_{\mathcal{SP}}$ implies $C_1 = C_2$. Given that for every $t \in \mathbb{T}_C$ the set $[\![t]\!]_{\mathcal{SP}}$ contains a single SP graph, we abuse notation and refer to $[\![t]\!]_{\mathcal{SP}}$ as the SP graph contained in $[\![t]\!]_{\mathcal{SP}}$. Let us assume that C_1 is a basic action b, then $[\![C_1]\!]_{\mathcal{SP}}$ is a single edge graph $\xrightarrow{b}$,

it follows that $C_2 = b$ considering the edge labels preservation property of two isomorphic SP graphs. Let us consider now that $C_1 = \texttt{AND}(S_1^1; \cdots ; S_k^1)$ for some $k \geq 2$, which means that $[\![C_1]\!]_{\mathcal{SP}} = [\![S_1^1]\!]_{\mathcal{SP}} \parallel \cdots \parallel [\![S_k^1]\!]_{\mathcal{SP}}$. Given that $[\![C_1]\!]_{\mathcal{SP}} = [\![C_2]\!]_{\mathcal{SP}}$, for every $i \in \{1; \cdots ; k\}$, $[\![C_2]\!]_{\mathcal{SP}}$ contains a subgraph G_i which is isomorphic to $[\![S_i^1]\!]_{\mathcal{SP}}$. Thus, considering that $[\![S_i^1]\!]_{\mathcal{SP}}$ is either a basic action or a sequential composition, we have that $[\![C_2]\!]_{\mathcal{SP}} = G_1 \parallel \cdots \parallel G_k$ and, according to grammar (3), $C_2 = \texttt{AND}(S_1^2; \cdots ; S_k^2)$, for some terms $S_1^2; \cdots ; S_k^2 \in \mathbb{T}_S$. By the induction hypothesis, it follows that $[\![S_i^1]\!]_{\mathcal{SP}} = G_i = [\![S_i^2]\!]_{\mathcal{SP}}$ implies $S_i^1 = S_i^2$, which leads to $C_1 = C_2$. A similar proof can be obtained for the case where $C_1 = \texttt{SAND}(A_1^1; \cdots ; A_k^1)$.
Surjectivity follows from the fact that every SP graph has a unique (modulo associativity) decomposition in terms of the operators for sequential and parallel composition. Such a decomposition naturally corresponds to terms from $\mathbb{T}_C$.

2. $[\![\cdot]\!]_{\mathcal{SP}}$ is a bijection from $\mathbb{T}_N$ to $\mathcal{P}(\mathbb{G}_{\mathcal{SP}})$.
For injectivity, we assume that $[\![N_1]\!]_{\mathcal{SP}} = [\![N_2]\!]_{\mathcal{SP}}$. This implies that the sets $[\![N_1]\!]_{\mathcal{SP}}$ and $[\![N_2]\!]_{\mathcal{SP}}$ have the same size and the same elements. If this size is 1, then they both contain the same element, which uniquely corresponds to a term C, so $N_1 = C = N_2$. If the size of the sets is larger than 1, then N_i are of the form $\texttt{OR}(C_1^i; \cdots ; C_k^i)$, for $i \in \{1; 2\}$. Since $[\![N_1]\!]_{\mathcal{SP}} = [\![N_2]\!]_{\mathcal{SP}}$, the elements of these two sets are pairwise identical. Moreover, by definition, all arguments in N_i are different, which implies $k_1 = k_2$. From the previous item, it follows that the elements of $[\![N_i]\!]_{\mathcal{SP}}$ correspond uniquely to $C_1^i; \cdots ; C_k^i$. From the pairwise equality between the arguments of the two terms, it follows that N_1 and N_2 are identical.
For surjectivity, let $\{G_1; \cdots ; G_k\}$ be a set of SP graphs. It follows from the previous item that there exist trees $C_1; \cdots ; C_k$, such that $[\![C_i]\!]_{\mathcal{SP}} = G_i$, for $i \in \{1; \cdots ; k\}$. This implies that $\{G_1; \cdots ; G_k\} = [\![\texttt{OR}(C_1; \cdots ; C_k)]\!]_{\mathcal{SP}}$, which finishes the proof of surjectivity.

□

Lemma 1 shows that the grammar (3) generates all SAND attack trees in canonical form. It remains to be proven that the set of trees generated by the grammar (3) is equal to the set of normal forms of the term rewriting system $R_{\mathcal{SP}}$ and that these normal forms are unique.

Theorem 2. *The term rewriting system $R_{\mathcal{SP}}$ is strongly terminating and confluent.*

Proof. We show that the term rewriting system $R_{\mathcal{SP}}$ is terminating and confluent with help of the grammar (3), in four steps.

1. First, we show with standard methodology that the term rewriting system is terminating. We define the following norm which assigns natural numbers to terms:

$$\begin{aligned} |b| &= 1 \\ |\texttt{OR}(X_1; \cdots ; X_k)| &= |X_1| + \cdots + |X_k| + 2 \\ |\texttt{AND}(X_1; \cdots ; X_k)| &= 2 \cdot |X_1| \cdot \cdots \cdot |X_k| \\ |\texttt{SAND}(X_1; \cdots ; X_k)| &= 2 \cdot |X_1| \cdot \cdots \cdot |X_k| \end{aligned}$$

It can be easily verified that for every rewrite rule $l \to r \in R_{\mathcal{SP}}$, we have $|l| > |r|$. Consequently, there are no infinite reduction sequences, or, in other words, the term rewriting system is strongly terminating. Notice that because we consider term rewriting modulo commutativity of OR and AND, we have to verify that the left-hand side and the right-hand side of equations (E_1) and (E_2) have equal norms [11]. This is clearly the case.

2. Now we prove that the terms produced by the grammar (3) are exactly the normal forms with respect to $R_{\mathcal{SP}}$. For the terms in $\mathbb{T}_C$, none of the rewrite rules can be applied, because these terms do not contain OR, have no occurrences of AND containing an argument of type AND, have no occurrences of SAND containing an argument of type SAND, and do not contain operators with a single argument. We extend this to terms $\mathbb{T}_N$ by observing that all OR operators occurring in such terms have at least two arguments and that all these arguments are di□erent.
   Conversely, consider a term t in normal form that contains an OR operator. Then $t = \mathtt{OR}(t_1; \ldots; t_n)$, where the t_i do not contain an OR operator, else (E_3), (E_{10}), or ($E_{10'}$) can be applied. It remains to show that normal form terms without occurrence of an OR operator are in $\mathbb{T}_C$. Such terms are basic terms or have SAND or AND as their top-level operator. The last two cases are symmetric and we therefore only consider the case $\mathtt{AND}(t_1; \ldots; t_n)$. We must show that each t_i is a basic term or in the form $t_i = \mathtt{SAND}(t'_1; \ldots; t'_m)$. Suppose not, then there exists a t_i that has AND as its top-level operator. It follows that the term is not in normal form because (E_4) can be applied.
3. The normal forms are unique. To show that the normal forms are unique, assume that N_1 and N_2 are both normal forms for a SAND attack tree t. Since the rewrite system $R_{\mathcal{SP}}$ was constructed by orienting the axioms from $E_{\mathcal{SP}}$, we have that $E_{\mathcal{SP}} \vdash N_1 = N_2$. This means that $[\![N_1]\!]_{\mathcal{SP}} = [\![N_2]\!]_{\mathcal{SP}}$. From bijectivity proven in Lemma 1, we obtain $N_1 = N_2$.
4. Now that we have proven termination and uniqueness of normal forms, it immediately follows that the term rewriting system is confluent [6].

□

Example 5 illustrates the notion of canonical form for SAND attack trees.

Example 5. The canonical form of the SAND attack tree t in Figure 1 is the tree

$$t' = \mathtt{OR}\Big(\mathtt{SAND}\big(ftp; rsh; lobf\big); \mathtt{AND}\big(ssh; rsa\big)\Big)$$

shown in Figure 3. It is easily seen to be in normal form with respect to $R_{\mathcal{SP}}$.

4.3 SP Semantics as a Generalization of the Multiset Semantics

Having a complete set of axioms for the SP semantics allows us to formalize the relation between SAND attack trees under the SP semantics and attack trees under the multiset semantics, denoted by $[\![\cdot]\!]_{\mathcal{M}}$. This is achieved by extracting a

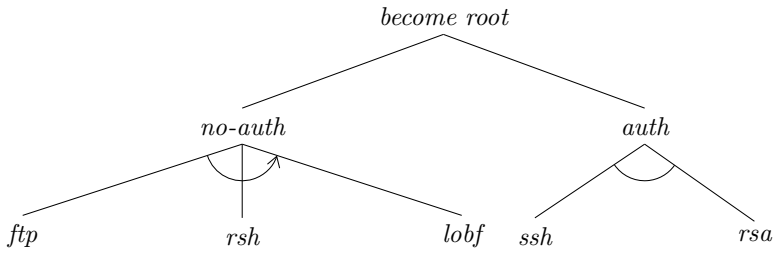

Fig. 3. SAND attack tree t' equivalent to SAND attack tree t from Figure 1

complete set of axioms for the multiset semantics for attack trees from the set $E_{\mathcal{SP}}$. Let $E_{\mathcal{M}}$ be the subset of axioms from $E_{\mathcal{SP}}$ that do not contain the SAND operator, i.e., $E_{\mathcal{M}} = \{(E_1), (E_2), (E_3), (E_4), (E_5), (E_6), (E_{10}), (E_{11})\}$.

Theorem 3. *The axiom system $E_{\mathcal{M}}$ is a complete set of axioms for the multiset semantics for attack trees.*

Proof. In [13, Theorem 4.9], a complete axiomatization of the multiset semantics for an extention of attack trees called attack–defense trees (ADTrees) is given. In the following, we call that axiomatization E_{ADT}. ADTrees are a superset of attack trees. They may contain defender's nodes modeled by the so called opponent's functions and countermeasures. We claim that $E_{\mathcal{M}}$ is a complete axiomatization of the multiset semantics for attack trees. Obviously if two attack trees are equal with respect to $E_{\mathcal{M}}$, then they are also equal with respect to E_{ADT}. This is clear, because $E_{\mathcal{M}} \subset E_{ADT}$.

Conversely, we prove that if two attack trees are equal with respect to E_{ADT}, they are equal with respect to $E_{\mathcal{M}}$. This follows from the following syntactical reasoning. E_{ADT} contains function symbols which we call *countermeasures*. Observe by inspecting the axioms of E_{ADT} that if a countermeasure occurs at the left-hand side of an equation, then it also occurs at the right-hand side, and vice versa. Therefore, axioms (E_{13}); (E_{16}); (E_{17}); (E_{18}); (E_{19}); (E_{20}) from E_{ADT} can never be used in a derivation of equality of two standard attack trees. Further, observe that the remaining axioms (E_9) and (E_{12}) from E_{ADT} make use of *opponent's functions*. In these axioms, an opponent function occurs on the left-hand side if and only if it occurs on the right hand side. Thus these axioms are never used to equate two attack trees which do not contain opponent's nodes. The remaining axioms are precisely (E_1), (E_2), (E_3), (E_4), (E_5), (E_6), (E_{10}), (E_{11}). So, we can only use these axioms to derive equalities of attack trees with respect to E_{ADT}, which implies that such a derivation is also possible using axioms from $E_{\mathcal{M}}$. □

By comparing the complete sets of axioms $E_{\mathcal{SP}}$ and $E_{\mathcal{M}}$ we obtain that two attack trees are equivalent under the multiset semantics if and only if they are equivalent under the SP semantics. This is formalized in the following theorem.

Theorem 4. SAND *attack trees under the SP semantics are a* conservative extension *of attack trees under the multiset semantics.*

Proof. Let t and t' be standard attack trees. Let $[\![t]\!]_{\mathcal{M}}$ and $[\![t']\!]_{\mathcal{M}}$ be their interpretation in the multiset semantics and $[\![t]\!]_{\mathcal{SP}}$ and $[\![t']\!]_{\mathcal{SP}}$ be their interpretation in the SP semantics. We prove that $[\![t]\!]_{\mathcal{M}} = [\![t']\!]_{\mathcal{M}}$ if and only if $[\![t]\!]_{\mathcal{SP}} = [\![t']\!]_{\mathcal{SP}}$.

By Theorem 3, a complete axiomatization of the multiset semantics for attack trees consists of axioms (E_1), (E_2), (E_3), (E_4), (E_5), (E_6), (E_{10}), (E_{11}). The complete axiomatization of the SP semantic for SAND attack trees additionally contains axioms ($E_{4'}$), ($E_{6'}$), and ($E_{10'}$). Thus, every equivalence of attack trees under the multiset semantics is clearly an equivalence of SAND attack trees under the SP semantics.

To see the converse, we show that the additional axioms do not introduce new equalities on standard attack trees. First inspect the three additional axioms and note that all of them contain the SAND operator.

Next, observe that for all axioms, the set of variables occurring on the left-hand side is equal to the set of variables occurring on the right-hand side. Thus, there is no axiom eliminating all occurrences of a variable. In particular, we claim that all axioms transform terms containing a p-ary SAND expression, where $p \geq 2$, into terms containing a q-ary SAND expression, for some $q \geq 2$. This is evident for equations without the SAND operator (since no variables are eliminated) and remains to be shown for equations ($E_{4'}$), ($E_{6'}$), and ($E_{10'}$). Axiom ($E_{6'}$) introduces and removes unary SAND, but does not modify the single variable A and therefore satisfies the claim. The arities of the two left-hand side SAND operators in equation ($E_{4'}$) are l and $k + l + m$ and the arity of the right-hand side operator is $k + l + m$, where $k; m \geq 0$ and $l \geq 1$. Since $1 \leq l \leq k + l + m$ and both sides contain a SAND operator of arity $k + l + m$, if either of the two sides contains a SAND operator with two or more arguments, then so does the other side. Finally, since $l \geq 1$, the arity of the SAND operator on the left-hand side of equation ($E_{10'}$) is equal to the arities of the SAND operators on its right-hand side and at least one SAND operator occurs on the right-hand side.

We can now show that none of the three axioms ($E_{4'}$), ($E_{6'}$), ($E_{10'}$) introduces new equalities on standard attack trees. In particular, axiom ($E_{6'}$) introduces and removes unary SAND, but this does not introduce new equalities on standard attack trees. Equations ($E_{4'}$) and ($E_{10'}$) match unary SAND, but require a further SAND with 2 or more arguments to add a new equality. Since, by the above claim, no p-ary SAND for $p \geq 2$ can be introduced with any of the equations, the additional equations do not introduce new equalities on standard attack trees.

□

5 Attributes

Attack trees do not only serve to represent security scenarios in a graphical way. They can also be used to quantify such scenarios with respect to a given parameter, called an *attribute*. Typical examples of attributes include the likelihood that

the attacker's goal is satisfied and the minimal time or cost of an attack. Schneier described [26] an intuitive bottom-up algorithm for calculating attribute values on attack trees: attribute values are assigned to the leaf nodes and two functions[6] (one for the OR and one for the AND refinement) are used to propagate the attribute value up to the root node. Mauw and Oostdijk showed [18] that if the binary operations induced by the two functions define a semiring, then the evaluation of the attribute on two attack trees equivalent with respect to the multiset semantics yields the same value. This result has been generalized to any semantics and attribute that satisfy a notion of *compatibility* [13]. We briefly discuss it for SAND attack trees at the end of this section. We start with a demonstration on how the bottom-up evaluation algorithm can naturally be extended to SAND attack trees.

An *attribute domain for an attribute* A_α *on* SAND *attack trees* is a tuple $D_\alpha = (V_\alpha; \triangledown_\alpha; \triangle_\alpha; \Diamond_\alpha)$ where V_α is a set of values and $\triangledown_\alpha; \triangle_\alpha; \Diamond_\alpha$ are families of k-ary functions of the form $V_\alpha \times \cdots \times V_\alpha \to V_\alpha$, associated to OR, AND, and SAND refinements, respectively. An *attribute for* SAND *attack trees* is a pair $A_\alpha = (D_\alpha; \ _\alpha)$ formed by an attribute domain D_α and a function $_\alpha : \mathbb{B} \to V_\alpha$, called *basic assignment* for A_α, which associates a value from V_α with each basic action $b \in \mathbb{B}$.

Definition 6. *Let* $A_\alpha = \big((V_\alpha; \triangledown_\alpha; \triangle_\alpha; \Diamond_\alpha); \ _\alpha\big)$ *be an attribute. The attribute evaluation function* $: \mathbb{T}_{\texttt{SAND}} \to V_\alpha$ *which calculates the value of attribute* A_α *for every* SAND *attack tree* $t \in \mathbb{T}_{\texttt{SAND}}$ *is defined recursively as follows*

$$(t) = \begin{cases} \ _\alpha(t) & \text{if } t = b;\ b \in \mathbb{B} \\ \triangledown_\alpha\big(\ (t_1); \ldots; \ (t_k)\big) & \text{if } t = \texttt{OR}(t_1; \ldots; t_k) \\ \triangle_\alpha\big(\ (t_1); \ldots; \ (t_k)\big) & \text{if } t = \texttt{AND}(t_1; \ldots; t_k) \\ \Diamond_\alpha\big(\ (t_1); \ldots; \ (t_k)\big) & \text{if } t = \texttt{SAND}(t_1; \ldots; t_k) \end{cases}$$

The following example illustrates the bottom-up evaluation of the attribute *minimal attack time* on the SAND attack trees given in Example 1.

Example 6. Let denote the minimal time that the attacker needs to achieve her goal. We make the following assignments to the basic actions: $ftp \mapsto 3$, $rsh \mapsto 5$, $lobf \mapsto 7$, $ssh \mapsto 8$, $rsa \mapsto 9$. Since we are interested in the minimal attack time, the function for an OR node is defined by $\triangledown_\alpha(x_1; \ldots; x_k) = \min\{x_1; \ldots; x_k\}$. The function for an AND node is $\triangle_\alpha \ (x_1; \ldots; x_k) = \max\{x_1; \ldots; x_k\}$, which models that the children of a conjunctively refined node are executed in parallel. Finally, in order to model that the children of a SAND node need to be executed sequentially, we let $\Diamond_\alpha(x_1; \ldots; x_k) = \sum_{i=1}^{k} x_i$. According to Definition 6, the minimal attack time for our running scenario t is

$$\triangledown_\alpha\Big(\Diamond_\alpha\big(\Diamond_\alpha(3;5);7\big); \triangle_\alpha \ (8;9)\Big) = \min\Big(\Sigma\big(\Sigma(3;5);7\big); \max(8;9)\Big) = 9.$$

[6] These are actually families of functions representing infinitely many k-ary function symbols, for all $k \geq 2$.

In the case of standard attack trees, the bottom-up procedure uses only two functions to propagate the attribute values to the root – one for conjunctive and one for disjunctive nodes. This means that the same function is employed to calculate the value of every conjunctively refined node, independently of whether its children need to be executed sequentially or can be executed simultaneously. Evidently, with SAND attack trees, we can apply di□erent propagation functions for AND and SAND nodes, as in Example 6. Therefore, SAND attack trees can be evaluated over a larger set of attributes, and hence may provide more accurate evaluations of attack scenarios than standard attack trees.

To guarantee that the evaluation of an attribute on equivalent attack trees yields the same value, the attribute domain must be *compatible* with a considered semantics [13]. Our complete set of axioms is a useful tool to check for compatibility. Consider an attribute domain $D_\alpha = (V_\alpha; \nabla_\alpha; \triangle_\alpha; \Diamond_\alpha)$, and let be a mapping $= \{\texttt{OR} \mapsto \nabla_\alpha; \texttt{AND} \mapsto \triangle_\alpha; \texttt{SAND} \mapsto \Diamond_\alpha\}$. Guaranteeing that D_α is compatible with a semantics axiomatized by E amounts to verifying that the equality $(l) = (r)$ holds in V_α, for every axiom $l = r \in E$. It is an easy exercise to show that the attribute domain for minimal attack time, considered in Example 6, is compatible with the SP semantics for SAND attack trees.

6 Conclusions

We have formalized the extension of attack trees with sequential conjunctive refinement, called SAND, and given a semantics to SAND attack trees in terms of sets of series-parallel graphs. This SP semantics naturally extends the multiset semantics for attack trees from [18]. We have shown that the notion of a complete set of axioms for a semantics and the bottom-up evaluation procedure can be generalized from attack trees to SAND attack trees, and have proposed a complete axiomatization of the SP semantics.

A number of recently proposed solutions focus on extending attack trees with defensive measures [25,13]. These extensions support reasoning about security scenarios involving two players – an attacker and a defender – and the interaction between them. In future work, we intend to add the SAND refinement to such trees. Afterwards, we plan to investigate sequential disjunctive refinement, as used for instance in [2]. Our goal is to propose a complete formalization of trees with attack and defense nodes, that have parallel and sequential, conjunctive and disjunctive refinements. The findings will be implemented in the software application ADTool [12].

Ackn The research leading to these results has received funding from the European Union Seventh Framework Programme under grant agreement number 318003 (TREsPASS) and from the Fonds National de la Recherche Luxembourg under grant C13/IS/5809105.

References

1. Ardi, S., Byers, D., Shahmehri, N.: Towards a structured unified process for software security. In: SESS'06. pp. 3–10. ACM (2006)
2. Arnold, F., Hermanns, H., Pulungan, R., Stoelinga, M.: Time-Dependent Analysis of Attacks. In: Abadi, M., Kremer, S. (eds.) POST'14. LNCS, vol. 8414, pp. 285–305. Springer (2014)
3. Brooke, P., Paige, R.: Fault Trees for Security System Design and Analysis. Computers & Security 22(3), 256–264 (2003)
4. Byers, D., Ardi, S., Shahmehri, N., Duma, C.: Modeling software vulnerabilities with vulnerability cause graphs. In: ICSM'06. pp. 411–422 (2006)
5. Camtepe, S., Yener, B.: Modeling and Detection of complex Attacks. In: SecureComm'07. pp. 234–243. IEEE (2007)
6. Dershowitz, N.: Review: Term Rewriting Systems by Terese. Theory Pract. Log. Program. 5(3), 395–399 (2005)
7. Gischer, J.L.: The Equational Theory of Pomsets. Theor. C. Sc. 61, 199–224 (1988)
8. Jhawar, R., Kordy, B., Mauw, S., Radomirović, S., Trujillo-Rasua, R.: Attack Trees with Sequential Conjunction. In: Proceedings of the International Conference on ICT Systems Security and Privacy Protection (IFIP SEC 2015). Springer (2015), (to appear).
9. Jürgenson, A., Willemson, J.: Processing Multi-Parameter Attacktrees with Estimated Parameter Values. In: Miyaji, A., Kikuchi, H., Rannenberg, K. (eds.) IWSEC. LNCS, vol. 4752, pp. 308–319. Springer (2007)
10. Khand, P.A.: System level security modeling using attack trees. In: IC4'09. pp. 1–6 (2009)
11. Klop, J.W., Bezem, M., Vrijer, R.C.D. (eds.): Term Rewriting Systems. Cambridge University Press, New York, NY, USA (2001)
12. Kordy, B., Kordy, P., Mauw, S., Schweitzer, P.: ADTool: Security Analysis with Attack–Defense Trees. In: Joshi, K.R., Siegle, M., Stoelinga, M., D'Argenio, P.R. (eds.) QEST'13. LNCS, vol. 8054, pp. 173–176. Springer (2013)
13. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Attack–Defense Trees. Journal of Logic and Computation 24(1), 55–87 (2014)
14. Kordy, B., Piètre-Cambacédès, L., Schweitzer, P.: DAG-Based Attack and Defense Modeling: Don't Miss the Forest for the Attack Trees. Computer Science Review 13–14(0), 1–38 (2014)
15. Kordy, B., Pouly, M., Schweitzer, P.: Computational Aspects of Attack–Defense Trees. In: Bouvry, P., Klopotek, M.A., Leprévost, F., Marciniak, M., Mykowiecka, A., Rybinski, H. (eds.) S&IIS'11. LNCS, vol. 7053, pp. 103–116. Springer (2011)
16. Kordy, B., Pouly, M., Schweitzer, P.: A Probabilistic Framework for Security Scenarios with Dependent Actions. In: Albert, E., Sekerinski, E. (eds.) iFM'14. LNCS, vol. 8739. Springer (2014)
17. Liu, Y., Man, H.: Network vulnerability assessment using Bayesian networks. In: Proceedings of SPIE Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2005. vol. 5812, pp. 61–71 (2005)
18. Mauw, S., Oostdijk, M.: Foundations of Attack Trees. In: Won, D., Kim, S. (eds.) ICISC'05. LNCS, vol. 3935, pp. 186–198. Springer (2006)
19. Ou, X., Boyer, W.F., McQueen, M.A.: A scalable approach to attack graph generation. In: CCS'06. pp. 336–345 (2006)
20. Peine, H., Jawurek, M., Mandel, S.: Security Goal Indicator Trees: A Model of Software Features that Supports E□cient Security Inspection. In: HASE'08. pp. 9–18. IEEE Computer Society (2008)

21. Piètre-Cambacédès, L., Bouissou, M.: Beyond Attack Trees: Dynamic Security Modeling with Boolean Logic Driven Markov Processes (BDMP). In: EDCC'10. pp. 199–208. IEEE Computer Society, Los Alamitos, CA, USA (2010)
22. Pinchinat, S., Acher, M., Vojtisek, D.: Towards Synthesis of Attack Trees for Supporting Computer-Aided Risk Analysis. In: Software Engineering and Formal Methods. LNCS, vol. 8938, pp. 363–375. Springer (2014)
23. Qin, X., Lee, W.: Attack Plan Recognition and Prediction using Causal Networks. In: ACSAC'04. pp. 370–379 (2004)
24. Qin, X., Lee, W.: Attack plan recognition and prediction using causal networks. In: 20th Annual Computer Security Applications Conference. pp. 370–379 (2004)
25. Roy, A., Kim, D.S., Trivedi, K.S.: Attack Countermeasure Trees (ACT): towards unifying the constructs of attack and defense trees. Security and Communication Networks 5(8), 929–943 (2012)
26. Schneier, B.: Attack Trees: Modeling Security Threats. Dr. Dobb's Journal of Software Tools 24(12), 21–29 (1999)
27. Sheyner, O., Haines, J., Jha, S., Lippmann, R., Wing, J.M.: Automated generation and analysis of attack graphs. In: S&P'02. pp. 273–284. IEEE (2002)
28. Wang, J., Whitley, J.N., Phan, R.C.W., Parish, D.J.: Unified Parametrizable Attack Tree. Int. Journal for Information Security Research 1(1), 20–26 (2011)
29. Wen-ping, L., Wei-min, L.: Space Based Information System Security Risk Evaluation Based on Improved Attack Trees. In: (MINES'11). pp. 480–483 (2011)
30. Willemson, J., Jürgenson, A.: Serial Model for Attack Tree Computations. In: Lee, D., Hong, S. (eds.) ICISC'09. LNCS, vol. 5984, pp. 118–128. Springer (2010)