ADTool: Security Analysis with Attack–Defense Trees*

Barbara Kordy, Piotr Kordy, Sjouke Mauw, Patrick Schweitzer
{barbara.kordy, piotr.kordy, sjouke.mauw,
patrick.schweitzer}@uni.lu

University of Luxembourg, SnT

Abstract. ADTool is free, open source software assisting graphical modeling and quantitative analysis of security, using attack–defense trees. The main features of ADTool are easy creation, efficient editing, and automated bottom-up evaluation of security-relevant measures. The tool also supports the usage of attack trees, protection trees and defense trees, which are all particular instances of attack–defense trees.

1 Background and Motivation

Attack–defense trees (ADTrees) extend and improve the well-known formalism of attack trees, by including not only the actions of an attacker, but also possible counteractions of a defender. Since interactions between an attacker and a defender are modeled explicitly in ADTrees, the extended formalism allows for a more thorough and accurate security analysis compared to regular attack trees. This paper presents ADTool software [7] which supports quantitative and qualitative security assessment using attack–defense trees.

Theoretical foundations of the ADTree methodology, including a graphical and a term-based syntax as well as numerous formal semantics, have been introduced in [6]. A mathematical framework for quantitative evaluation of ADTrees is based on the notion of attributes, which allow us to formalize and specify relevant security metrics. Standard quantitative analysis of ADTrees relies on a step-wise computation procedure. Numerical values are assigned to all atomic actions, represented by the non-refined nodes. The values for the remaining nodes, including the root of the tree, are deduced automatically in a bottom-up way. This bottom-up algorithm makes use of attribute domains which specify operators to be used while calculating values for different node configurations.

The practical use of the ADTree methodology requires dedicated tool support. Lack of such support may result in numerous modeling difficulties and computational errors. On the one hand, there exist a number of commercial

* The research leading to the results presented in this work received funding from the Fonds National de la Recherche Luxembourg under the grants C08/IS/26 and PHD-09-167 and the European Commission's Seventh Framework Programme (FP7/2007-2013) under grant agreement number 318003 (TREsPASS).

software applications for attack tree-like modeling, including SecurITree[1] and AttackTree+[2]. However, these are closed source tools and their use is not free of charge. On the other hand, existing academic software, such as SeaMonster[3], does not support quantitative analysis and uniformly integrated defenses.

The above observations motivated the development of ADTool, which

- is a free and open source application supporting qualitative and quantitative analysis of tree-based models integrating attack and defense components;
- is based on well-founded formal framework;
- guides the user in constructing well-formed and visually appealing models;
- facilitates sharing, management and updating of the models;
- automates computation of security related parameters.

This paper provides a brief overview of the main features and practical capabilities of ADTool. For a more detailed description we refer the reader to an extended and illustrated version of this article [5] and to the ADTool manual available at `http://satoss.uni.lu/software/adtool/manual.pdf`.

2 Main features of ADTool

ADTool is guiding the user in constructing models that comply with the graphical ADTree language. All options that allow to modify or refine the models can be accessed via a user-friendly GUI of the application.

ADTool uses an improved version of Walker's algorithm [2] to produce trees having an appealing layout. Furthermore, when an ADTree is built, the corresponding attack–defense term (ADTerm), is immediately displayed. ADTerms form a compact, algebraic representation of ADTrees. The shortest tree edit distance algorithm [3] implemented in ADTool ensures that when an ADTerm is modified, the corresponding ADTree is adapted accordingly.

ADTool provides advanced features for model manipulation and management. Folding, expanding and zooming options make the analysis of large models possible. Temporarily hiding parts of a tree permits users to focus on the displayed components. This is highly appreciated during industrial meetings and presentations. ADTrees created with ADTool can be saved as special .adt files, which enables their reuse and modification. Models can also be exported to vector graphics files (pdf), raster graphics files (png, jpeg) and LaTeX files (tex). Resulting figures can be used as illustrations in presentations, research papers and posters. A dedicated option makes it possible to print trees on a specified number of pages, which enhances readability of large-scale models.

The bottom-up algorithm for evaluation of attributes on ADTrees has been implemented in ADTool. Supported measures include: attributes based on real values (e.g., time, cost, probability), attributes based on levels (e.g., required

[1] `http://www.amenaza.com/`
[2] `http://www.isograph-software.com/2011/software/attacktree/`
[3] `http://sourceforge.net/projects/seamonster/`

skill level, reachability of the goal in less than k units of time), and Boolean properties (e.g., satisfiability of a scenario). The implemented measures can be computed from the point of view of an attacker (e.g., the cost of an attack), of a defender (e.g., the cost of defending a system), or relate to both of them (e.g., overall maximum power consumption). Using different attribute domains allows us to distinguish between actions executed sequentially or in parallel.

After a user selects an attribute, the tool decorates the ADTree with default values representing the worst case scenario, e.g., infinite cost or maximal required skill level. The user then customizes the inputs for the relevant non-refined nodes and the linear bottom-up algorithm computes the values of the remaining nodes. Input values can be modified directly on the tree or using an overview table which is particularly helpful in case of large models. The tool ensures that the provided values are consistent and belong to a specified value domain. This is especially important when several specialists supply values for different parts of the tree.

The tool has been extensively tested and has proven to be able to easily handle realistic models containing a few thousand nodes. The computations using ADTool are performed instantaneously. The limiting factor is the graphical display of ADTrees. For trees of more than ten thousand nodes, a delay of about five seconds occurs when a new node is added. This is due to the recalculation of the positions of some nodes.

3 Implementation Characteristics

The application has been written in a modular way with a clear distinction between the GUI and the Implementation Model. An overview of the ADTool architecture is depicted in Figure 1. The Implementation Model consists of the

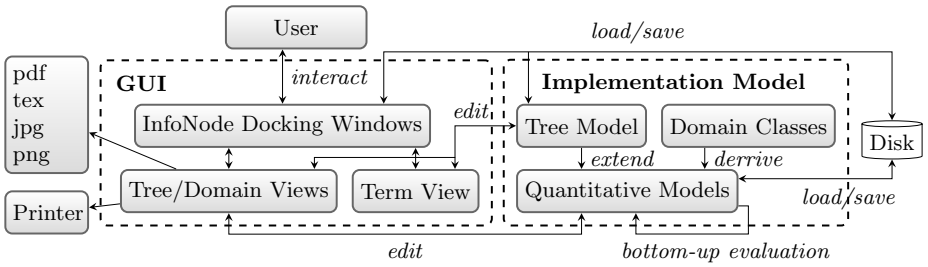

Fig. 1. An overview of the ADTool architecture

Tree Model (which stores the basic tree structure), Domain Classes (defining the implemented attribute domains), and Quantitative Models (which are derived from Domain Classes and contain inserted and computed values). The functionality of the tool can easily be extended by defining new attributes. For this purpose, a new Domain Class needs to be created and compiled. Domain

Classes have been designed to be simple, in order to make it possible for a user with minimal knowledge of Java to add a new domain. Due to the use of Java reflection, no recompilation or other modifications of the program are required after adding a new Domain Class.

ADTool runs on all common operating systems (Windows, Linux, Mac OS). The program is written in Java and it depends on the following free libraries: abego TreeLayout[4], implementing an efficient and customizable tree layout algorithm in Java, and InfoNode Docking Windows[5], a pure Java Swing based docking windows framework, allowing to set up windows in a flexible way and to save and restore their layout. ADTool is available for download and as an online application at `http://satoss.uni.lu/software/adtool/`.

4 Conclusion and future work

ADTool provides security consultants as well as academic researchers with a rigorous but user-friendly application that supports security analysis using ADTrees. It integrates two crucial modeling aspects: the creation of security models and their quantitative analysis. From a formal perspective, attack trees [8], protection trees [4], and defense trees [1] are instances of ADTrees. Thus, ADTool can also be employed to automate and facilitate the usage of all these formalisms.

We are currently working on combining the ADTree methodology with Bayesian Networks, to make probabilistic reasoning about scenarios involving dependent actions possible. Related theoretical findings and newly identified features will be implemented in the next versions of ADTool.

References

1. Bistarelli, S., Fioravanti, F., Peretti, P.: Defense Trees for Economic Evaluation of Security Investments. In: ARES'06. pp. 416–423. IEEE Computer Society (2006)
2. Buchheim, C., Jünger, M., Leipert, S.: Drawing rooted trees in linear time. Software: Practice and Experience 36(6), 651–665 (2006)
3. Demaine, E.D., Mozes, S., Rossman, B., Weimann, O.: An Optimal Decomposition Algorithm for Tree Edit Distance. ACM Trans. Algorithms 6(1), 2:1–2:19 (2009)
4. Edge, K.S., Dalton II, G.C., Raines, R.A., Mills, R.F.: Using Attack and Protection Trees to Analyze Threats and Defenses to Homeland Security. In: MILCOM. pp. 1–7. IEEE (2006)
5. Kordy, B., Kordy, P., Mauw, S., Schweitzer, P.: ADTool: Security Analysis with Attack–Defense Trees (Extended Version). CoRR abs/1305.6829 (2013)
6. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Attack–Defense Trees. Journal of Logic and Computation pp. 1–33 (2012), available online `http://logcom.oxfordjournals.org/content/early/2012/06/21/logcom.exs029.short?rss=1`
7. Kordy, P., Schweitzer, P.: ADTool (2012), `http://satoss.uni.lu/software/adtool`
8. Mauw, S., Oostdijk, M.: Foundations of Attack Trees. In: Won, D., Kim, S. (eds.) ICISC'05. LNCS, vol. 3935, pp. 186–198. Springer (2006)

[4] `http://code.google.com/p/treelayout/`

[5] `http://www.infonode.net/index.html?idw`