Attack Trees for Practical Security Assessment: Ranking of Attack Scenarios with ADTool 2.0

Olga Gadyatskaya, Ravi Jhawar, Piotr Kordy, Karim Lounis, Sjouke Mauw, and Rolando Trujillo-Rasua(✉)

SnT, University of Luxembourg, Luxembourg City, Luxembourg
{olga.gadyatskaya,ravi.jhawar,Piotr.Kordy,karim.lounis,
sjouke.mauw,rolando.trujillo}@uni.lu

Abstract. In this tool demonstration paper we present the ADTool2.0: an open-source software tool for design, manipulation and analysis of attack trees. The tool supports ranking of attack scenarios based on quantitative attributes entered by the user; it is scriptable; and it incorporates attack trees with sequential conjunctive refinement.

1 Introduction

Attack trees are a well-known and established methodology for security assessment that facilitates brainstorming, structures available information, and assists human experts in analysis. An attack tree is a graphical model, and as such it is better comprehensible than pure text-based approaches. However, graphical models require usable and efficient tools with suitable Graphical User Interfaces (GUIs) in order to be practical. Moreover, recent advances in automated risk assessment techniques now call for tool support to handle automatically generated attack trees with many thousands of nodes [2,3]. Therefore, the need for more comprehensive analysis tools emerged in the community. In this paper we present the ADTool2.0 that provides advanced capabilities for design, visualization, and analysis of attack trees [9], attack-defense trees [6], and attack trees with sequential conjunctive refinement (SAND attack trees for short) [4].

The ADTool2.0 is not a simple extension of the previous tool [5], but a fully revamped, more advanced system. It has been reimplemented using the advanced cross-platform Docking Frames library[1]. The new version of the tool brings in many new features, including ranking of critical attack scenarios, attack trees with the sequential AND (SAND) operator, and scriptability.

In contrast to many commercial tools, such as SecurITree[2] and AttackTree+[3], the ADTool2.0 is an open source software, freely available to the community[4].

The research leading to the results presented in this work received funding from the European Commission's Seventh Framework Programme (FP7/2007–2013) under grant agreement number 318003 (TREsPASS) and Fonds National de la Recherche Luxembourg under the grant C13/IS/5809105 (ADT2P).

[1] http://www.docking-frames.org/.

[2] http://www.amenaza.com.

[3] http://www.isograph.com/software/.

[4] https://github.com/tahti/ADTool2.

G. Agha and B. Van Houdt (Eds.): QEST 2016, LNCS 9826, pp. 159–162, 2016.
DOI: 10.1007/978-3-319-43425-4_10

Moreover, it continues to be the only software tool providing support for the attack-defense tree modeling language [6]. In that sense, the ADTool2.0 provides unique features in comparison to integration frameworks (e.g., the Möbius framework [1]) and tools based on attack graphs (e.g., ADVISE [8]).

2 Main Features of the ADTool2.0

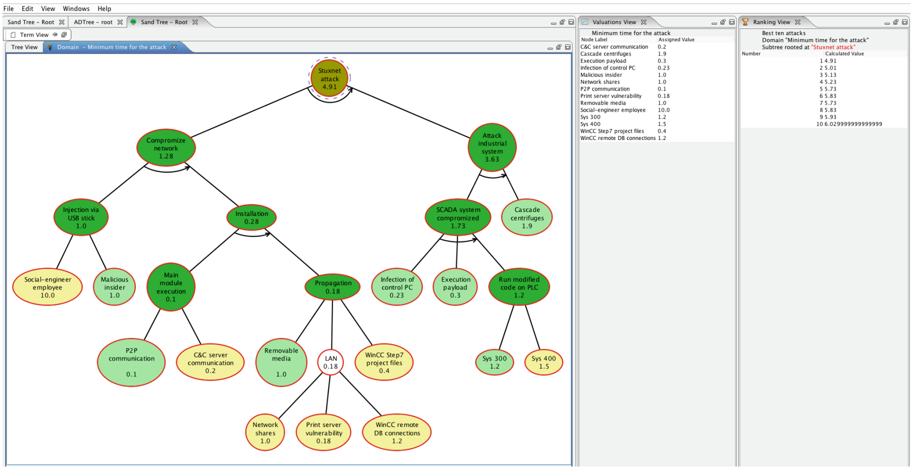
Sequential Conjunct Refinements in Attack Trees. The ADTool2.0 integrates a crucial modelling aspect: creation of attack trees with SAND refinements (consistent with the graphical language and semantics described in [4]) and their quantitative analysis. Usage of the SAND refinement allows the analyst to model and analyze attack scenarios involving several attack steps that need to be all executed in a specific order, as opposed to the standard AND refinement used to model execution of several attack steps in parallel.

After constructing a SAND attack tree, the user can assign an attribute domain (e.g., minimum time for the attack, probability of success) to the tree. Each leaf node is then initialized with a default value representing the worst case scenario (e.g., ∞ as the minimum time for the attack), and all other nodes are automatically assigned using an n-ary function, depending on the type of attribute and refinement operator, in order to evaluate the security scenario. The ADTool2.0 will automatically compute new attribute values using a bottom-up algorithm.

Ranking Attack Trees. Human ability to visualize and understand attack trees quickly decreases with the increase in size and complexity of the tree. Identifying important portions of an attack tree is therefore of paramount importance for security analysts; it allows to prioritize and focus on those branches that contribute most to the attacker goal. A systematic approach to prioritization is *ranking*, whereby a set of elements is sorted with respect to a total order. In attack graphs, a modelling language similar to attack trees, several ranking approaches have been defined [10]. In attack trees, however, ranking has been mostly neglected by both quantification methods and tools.

The ADTool2.0 implements an efficient and formal approach to rank attack scenarios. In particular, we have extended the bottom-up computation approaches proposed for attack trees [9], attack-defense trees [6], and SAND attack trees [4], in order to efficiently rank attack scenarios, where an attack scenario is either a *bundle* as in the formalisms in [6,9] or an *SP graph* as in [4]. Our approach works intuitively as follows. Given a set of quantitative values V for attack scenarios and a total order $\leq$ on V, we store at every node of the tree n least attacks with respect to the total order $\leq$, where n is a natural number representing a bound on the number of attack scenarios to be ranked.

Ranking results in the ADTool2.0 are shown in the *Ranking View* window, which can be opened from the menu Windows → Ranking View. As in the Attribute window, the Ranking window gives the option to open or create an attribute domain. By default, the ADTool2.0 uses as a total order the operator assigned to the OR gate in the attribute domain. A screenshot of the ADTool2.0


File
Edit
View
Windows
Help
Sand Tree – Root
ADTree – root
Sand Tree – Root
Term View
Tree View
Domain – Minimum time for the attack
Stuxnet attack 4.91
Compromize network 1.28
Attack industrial system 3.63
Injection via USB stick 1.0
Installation 0.28
SCADA system compromized 1.73
Cascade centrifuges 1.9
Social-engineer employee 10.0
Malicious insider 1.0
Main module execution 0.1
Propagation 0.18
Infection of control PC 0.23
Execution payload 0.3
Run modified code on PLC 1.2
P2P communication 0.1
C&C server communication 0.2
Removable media 1.0
LAN 0.18
WinCC Step7 project files 0.4
Sys 300 1.2
Sys 400 1.5
Network shares 1.0
Print server vulnerability 0.18
WinCC remote DB connections 1.2
Valuations View
Ranking View

Minimum time for the attack

Node Label	Assigned Value
C&C server communication	0.2
Cascade centrifuges	1.9
Execution payload	0.3
Infection of control PC	0.23
Malicious insider	1.0
Network shares	1.0
P2P communication	0.1
Print server vulnerability	0.18
Removable media	1.0
Social-engineer employee	10.0
Sys 300	1.2
Sys 400	1.5
WinCC Step7 project files	0.4
WinCC remote DB connections	1.2

Best ten attacks
Domain "Minimum time for the attack"
Subtree rooted at "Stuxnet attack"

Number	Calculated Value
1	4.91
2	5.01
3	5.13
4	5.23
5	5.73
6	5.83
7	5.73
8	5.83
9	5.93
10	6.029999999999999

Fig. 1. Screenshot of the ADTool2.0 with the ranking feature. The SAND attack tree used represents the Stuxnet attack, and the ranking is based on the minimal time of attack parameter. The attack scenario (all its attack nodes) with the minimal time of execution is highlighted in green by the tool.

provided in Fig. 1 shows an example of the ranking feature applied to a SAND attack tree modelling the Stuxnet attack (inspired by [7]).

In order to rank attack scenarios up to a given node in the tree, we ought to click that node in the domain for which we want to see the ranking. Doing so, the Ranking view window will automatically update with a table containing optimal attacks with respect to the chosen attribute domain. The ADTool2.0 also offers the option to highlight those nodes that contribute most to the attack, which can be done by clicking on attack scenarios in the ranking table.

Scripting. Scriptability, whereby a tool can be run by scripts and without a GUI, is an important feature of security assessment tools. It allows sensitivity analysis (a standard technique to automatically assess how changes in some attribute values affect the overall security posture) and integration into tool chains. With the current version of the tool, it is now also possible to experiment with countermeasure selection: we can write scripts that will input several attack-defense trees with different defense scenarios applied to a particular attack, and output the best countermeasure set based on the results of the ranking.

In the scripting mode, which is typically executed from the command line[5], the ADTool2.0 supports input files of different formats (e.g., XML files) containing any of the supported attack trees (e.g., SAND trees), and provides various types of outputs such as the most critical attacks or the result of a bottom-up calculation. By using this scriptability feature, the ADTool2.0 has been integrated into the TREsPASS project tool chain[6], where it is used to visualize attack-defense scenarios and automatically or manually produced attack trees.

[5] Execute `java -jar ADTool-2.0.jar --help` from the command line for basic help.

[6] http://www.trespass-project.eu/.

Usability Features. The ADTool2.0 includes many usability features, e.g., copy-paste of subtrees, handling of multiple trees, reorder of children nodes, and extended input format (automatically generated attack trees [3] not conforming to the ADTool2.0 XML schema). The ADTool2.0 can handle and analyze *large trees* with several thousand nodes (automatically generated trees are typically of that size).

3 Conclusion

In this tool demonstration paper we presented the main features of the ADTool2.0, which is an open-source software tool for displaying, designing and analyzing attack trees in many flavors (SAND attack trees [4], attack-defense trees [6], and classical attack trees [9]). The ADTool2.0 supports ranking of attack scenarios based on the quantitative values selected by the end-user (e.g., time of attack, cost, and probability). In addition, it can be scripted for performing sensitivity analysis or running in tool chains.

References

1. Deavours, D.D., Clark, G., Courtney, T., Daly, D., Derisavi, S., Doyle, J.M., Sanders, W.H., Webster, P.G.: The möbius framework and its implementation. IEEE Trans. Softw. Eng. **28**(10), 956–969 (2002)
2. Gadyatskaya, O.: How to generate security cameras: towards defence generation for socio-technical systems. In: Mauw, S., et al. (eds.) GraMSec 2015. LNCS, vol. 9390, pp. 50–65. Springer, Heidelberg (2016). doi:10.1007/978-3-319-29968-6_4
3. Ivanova, M.G., Probst, C.W., Hansen, R.R., Kammüller, F.: Transforming graphical system models to graphical attack models. In: Mauw, S., et al. (eds.) GraMSec 2015. LNCS, vol. 9390, pp. 82–96. Springer, Heidelberg (2016). doi:10.1007/978-3-319-29968-6_6
4. Jhawar, R., Kordy, B., Mauw, S., Radomirović, S., Trujillo-Rasua, R.: Attack trees with sequential conjunction. In: Federrath, H., Gollmann, D., Chakravarthy, S.R. (eds.) SEC 2015. IFIP AICT, vol. 455, pp. 339–353. Springer, Heidelberg (2015). doi:10.1007/978-3-319-18467-8_23
5. Kordy, B., Kordy, P., Mauw, S., Schweitzer, P.: ADTool: security analysis with attack–defense trees. In: Joshi, K., Siegle, M., Stoelinga, M., D'Argenio, P.R. (eds.) QEST 2013. LNCS, vol. 8054, pp. 173–176. Springer, Heidelberg (2013)
6. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Attack-defense trees. J. Log. Comput. **24**(1), 55–87 (2014)
7. Kriaa, S., Bouissou, M., Pietre-Cambacedes, L.: Modeling the Stuxnet attack with BDMP: towards more formal risk assessments. In: Proceedings of the CRiSIS (2012)
8. LeMay, E., Ford, M.D., Keefe, K., Sanders, W.H., Muehrcke, C.: Model-based security metrics using ADversary VIew Security Evaluation (ADVISE). In: Proceedings of QEST 2011, pp. 191–200. IEEE Computer Society, Washington, DC (2011)
9. Mauw, S., Oostdijk, M.: Foundations of attack trees. In: Won, D.H., Kim, S. (eds.) ICISC 2005. LNCS, vol. 3935, pp. 186–198. Springer, Heidelberg (2006)
10. Mehta, V., Bartzis, C., Zhu, H., Clarke, E.: Ranking attack graphs. In: Zamboni, D., Kruegel, C. (eds.) RAID 2006. LNCS, vol. 4219, pp. 127–144. Springer, Heidelberg (2006)