Bridging Two Worlds: Reconciling Practical Risk Assessment Methodologies with Theory of Attack Trees

Olga Gadyatskaya[1(✉)], Carlo Harpes[2], Sjouke Mauw[1], Cédric Muller[1,2], and Steve Muller[2]

[1] SnT, University of Luxembourg, Luxembourg, Luxembourg
{olga.gadyatskaya,sjouke.mauw}@uni.lu, cedric.muller.001@student.uni.lu
[2] itrust Consulting, Niederanven, Luxembourg
{harpes,cedric.muller,steve.muller}@itrust.lu

Abstract. Security risk treatment often requires a complex cost-benefit analysis to be carried out in order to select countermeasures that optimally reduce risks while having minimal costs. According to ISO/IEC 27001, risk treatment relies on catalogues of countermeasures, and the analysts are expected to estimate the residual risks. At the same time, recent advancements in attack tree theory provide elegant solutions to this optimization problem. In this paper we propose to bridge the gap between these two worlds by introducing optimal countermeasure selection problem on attack-defense trees into the TRICK security risk assessment methodology.

1 Introduction

Recent attacks, such as Ashley Madison, Sony and Target, are well-known to many of us. However, it is not only large or famous organizations that are targeted by cyber criminals. Any company can be attacked, and companies have to respond to this huge threat landscape by improving their security protection. Nowadays the ability to better identify and prioritize security risks, and to detect and mitigate incidents becomes *critical*. Companies need to look for the means to pinpoint and quantify security gaps and to eliminate them by introducing new security controls. Usually controls are selected following some established guidelines. There exist *generic* security guidelines, e.g., IT-Grundschutz Catalogues [4], ISO/IEC 27002 [13], NIST 800-53 [19], and *domain-specific* ones. Examples of the latter are PCI DSS [22] in the banking domain, the controls catalogue [6] in the air traffic management domain, ISO 27799 [10] for health informatics, ISO 27019 [14] for the energy utility industry. Furthermore, controls can be also identified by the interested parties and analysts in brainstorming [24].

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).

B. Kordy et al. (Eds.): GraMSec 2016, LNCS 9987, pp. 80–93, 2016.
DOI: 10.1007/978-3-319-46263-9_5

On the other hand, in the academic world there exist many techniques and tools to select countermeasures in an optimal way. These techniques can be roughly classified as more generic (e.g., optimal countermeasure selection on attack trees [2,25]), or more domain-specific (for example, network hardening techniques on attack graphs [1]).

These two worlds focused on the same problem of *countermeasure selection* rarely engage with each other, one of the reasons being that industrial risk treatment practices are entangled with many other practices and processes in the company (governance and compliance, but also business operations), while academic solutions tend to be more isolated and focused on particular aspects. Furthermore, design of new risk assessment methods generally follows the requirements and guidelines imposed by relevant standardization and regulation bodies [23], i.e., ISO 27001 [12] and NIST Cybersecurity Framework [20]. Academic solutions need to be introduced into risk management methodologies *on top* of these guidelines. In this position paper we propose to bridge the two worlds of practical risk management and theoretical results on optimal security control selection in attack trees. As the security risk assessment method we apply the TRICK Service framework developed and used in Luxembourg. We consider to bridge this practical assessment process with an academic result concerning the optimal countermeasure selection problem on attack trees, which is an instance of approaches proposed by Roy et al. [25] and Aslanyan and Nielson [2].

The paper is structured as follows. We give an outline of the TRICK Service in Sect. 2, and present a background on attack tree theory in Sect. 3. Our proposal for bridging these two domains in the context of optimal selection of countermeasures in risk treatment is presented in Sect. 4. We discuss possible choices for selecting countermeasures in Sect. 5, and we present the optimization problem that we solve for allocation of defensive nodes in attack trees in Sect. 6. We illustrate our current approach on a private cloud use case in Sect. 7. We then overview our next steps and conclude in Sect. 8.

2 The TRICK Service

TRICK Service (**T**ool for **R**isk management of an **I**nformation Security Management System based on a **C**entral **K**nowledge base), developed by itrust consulting in Luxembourg, is a web-based risk assessment and management tool for identification, analysis and estimation of assets, threats, vulnerabilities, risk scenarios and security measures. It helps the analyst to determine a list of security measures to be implemented in order to reduce the impact or the likelihood of possible risk scenarios.

Risk analysis in TRICK starts with establishing the context by collecting information about the type and business processes of the organisation and filling in a table, according to ISO 27005:2011 [11]. This information is used by the analyst to establish the most important assets considering the sector of the organisation.

After the context definition, a brainstorming session identifies assets and risk scenarios in the organisation. Qualitative risk assessment is performed at this

stage to allow the analyst to estimate the exposure to identified threats, vulnerabilities and risks. The next step consists in identifying the security measures that are already implemented in the organization, and assessing their current implementation rate and cost, referring to norms, such as ISO/IEC 27002 [13].

The analyst then estimates the *annual loss expectancy* (ALE) of each asset-scenario pair, by multiplying the impact (in euros) that a scenario could have, with the annual expected probability that a scenario could occur on the asset.

A *risk reduction factor* (RRF) parameter is associated to each asset-scenario-countermeasure triple. The RRF is a coefficient that expresses the negative influence of a security control on the ALE generated by the occurrence of a scenario on an asset. For a given security control in relation to a given scenario acting on an asset, its RRF is a value between 0 and 1, where RRF=0 means that the countermeasure is useless, and RRF=1 signifies perfect protection.

Implementation (or partial implementation) of a security control results in an ALE reduction, based on the RRF and the implementation rate. For the sake of simplicity we will not take the implementation rate of a security measure into account, assuming that any countermeasure is fully implemented.

As we have seen from the description, in order to ensure that the overall risk assessment, analysis and treatment process is correct, the analyst needs to come up with a (sufficiently) complete list of scenarios and evaluate their respective probabilities. If scenarios are too generic, it is very challenging to evaluate their probabilities (or occurrence rates). At the same time, for simpler attack steps, e.g., vulnerability exploitation, it might be more easy to evaluate their chances to occur by relying, e.g., on the available statistics in the sector. To better estimate the residual ALE, we proposed to apply the attack tree formalism summarized in the next section.

3 Attack Tree Theory Background

Attack trees [26] are a graphical model useful for threat modelling and risk assessment [18,21]. They are comprehensible to stakeholders with different backgrounds and expertise, and they also enjoy various formal semantics [17] that allow for qualitative and quantitative analysis of attack scenarios. In a typical attack tree, the top node (the root) represents the goal of the attacker. For instance, a possible goal is *entering the system to manipulate the integrity* (risk scenario) *of financial transactions* (asset) *by arranging a money transfer to the attacker* (impact).

The root is *refined* into a set of child nodes that represent the different ways to achieve the goal. An **or**-refinement means that any child is sufficient to achieve the parent goal, and an **and**-refinement states that all children need to be achieved before the parent is achieved. Consequently, each child node can be further refined, until the remaining nodes are *simple* enough and do not require further refinement. These simple attack nodes are also called *atomic* attacks, and they are leaf nodes of the attack tree.

Probability computations on attack trees. For the scope of this paper we assume that all atomic attacks in the tree are *independent*, and that all attack nodes are unique in the tree. Then for two attack leaf nodes x and y that represent independent events, with respective probabilities $\mathbf{Pr}(x)$ and $\mathbf{Pr}(y)$, we can calculate their composed probability by $\mathbf{Pr}(x \wedge y) = \mathbf{Pr}(x)\mathbf{Pr}(y)$; and $\mathbf{Pr}(x \vee y) = \mathbf{Pr}(x) + \mathbf{Pr}(y) - \mathbf{Pr}(x)\mathbf{Pr}(y)$. A bottom-up evaluation can be further continued on intermediate nodes until the probability of the root node of the attack tree at, denoted as $\mathbf{Pr}(at)$, is computed. This evaluation can be done in, e.g., the ADTool [8,15].

Attack-defense trees. Attack trees consider the situation only from the perspective of the attacker. However, the main goal of using attack trees in practice is to systematize threat identification in order to improve risk treatment, i.e. identification of relevant countermeasures. Therefore, extensions of attack trees with defensive nodes emerged as a way to explicitly tackle the security control problem. Notable extensions include defense trees [3], protection trees [5], attack-countermeasure trees [25], and *attack-defense* trees [16]. In this work we focus on attack-defense trees as this formalism integrates attacks and countermeasures in the least restrictive way (i.e., defense nodes can be interleaved with attack nodes, while in other formalisms they are typically only leaf nodes).

The problem of countermeasure selection is not novel in the context of attack trees. Roy et al. considered the problem of optimal countermeasure selection for attack-countermeasure trees in [25], and Aslanyan and Nielson investigated optimal probability-cost balances on attack defense trees in [2]. Both of these works consider a tree with already pre-selected countermeasures, and the solution of the optimization problem is to find the subset of *already placed* countermeasures, such that the probability of attacker's success and the cost of selected controls are minimal (a set of Pareto-efficient solutions is offered in [2]). Our goal is to introduce optimal countermeasure selection akin to [2,25] into the TRICK risk assessment methodology.

4 Proposal for Bridging the Gap

We consider that the analyst who is using TRICK will now express threat scenarios as attack trees, and will perform the subsequent risk treatment steps using these trees.

The ROSI Function. The *return on security investment* (ROSI) function evaluates the investment made into security controls versus the obtained security improvement [9]. The average yearly cost of implementing a set of new countermeasures M (denoted as $\mathbf{cost}(M)$) corresponds to the investment, and the total ALE reduction obtained as a result of implementing these new countermeasures (denoted $\Delta ALE(M)$) corresponds to the yearly gains. Thus, for a set of controls M, $ROSI(M) = \Delta ALE_M - \mathbf{cost}(M)$.

Considering that the **Risk** equals **Impact** multiplied by **Probability** [3], we set the difference in the annual loss expectancy $\Delta ALE(M)$ as the product

of the **Impact** times the difference of yearly probability of occurrence without and with implementation of the set of countermeasures M [25].

The probability for the attacker to reach the goal and to implement the threat scenario can be evaluated through probabilities of atomic attack steps, as discussed in Sect. 3. At the same time, the impact of the attack tree (i.e., the impact in case the attacker reaches his/her goal and the threat scenario expressed in the attack tree has occurred) can be estimated independently from the tree. Thus we focus only on probability values and the selection of countermeasures based on how well they can reduce the attack success probability.

We consider that each countermeasure t has a possible effect on each attack node x. This effect is described by an **effectiveness** parameter, $\mathbf{eff}(t, x) \in [0,]$, with $\mathbf{eff}(t, x) = 0$ corresponding to a useless countermeasure for x, and $\mathbf{eff}(t, x) = 1$ defining perfect protection against x.

The effectiveness is defined so that the overall probability of the attack node x mitigated by t, which we denote as x_t, is defined as $\mathbf{Pr}(x_t) = \mathbf{Pr}(x)(1\text{-}\mathbf{eff}(t, x))$. Thus, the higher the effectiveness parameter of the countermeasure in the given context, the lower the resulting probability of attack.

The step of evaluating the security posture by considering already implemented countermeasures in TRICK, can be directly executed on the attack tree. The analyst will now place the existing countermeasures as defense nodes in the attack tree. Computation of probabilities in presence of countermeasures and their effectiveness can be done via the bottom-up evaluation algorithm; just like for attack trees. As a result of this step of considering already existing countermeasures, the analyst will obtain an attack-defense tree *adt* and will evaluate the overall probability of the considered attack scenario as $\mathbf{Pr}(adt)$. For simplicity, in this paper we consider that the analyst "starts from scratch", i.e., the infrastructure does not have any security controls implemented yet, and the analyst starts from an attack tree.

An important distinction of effectiveness from RRF in the context of TRICK is that RRF measures global influence of the countermeasure on the particular scenario occurring with the asset (i.e., on the whole attack tree), while effectiveness is more localized as it applies to an attack node (sub-scenario) in the attack tree, and reduces the probability of occurrence of only this node. The RRF in the TRICK context could be further defined for a set of countermeasures as a non-linear combination of their effectiveness parameters in the attack-defence tree. Thus the process of creating an attack tree, estimating the effectiveness parameters of available countermeasures, and selecting the optimal subset of countermeasures can in the future serve as a methodology to better estimate RRFs in the TRICK Service.

New Countermeasures From Catalogues. As we have mentioned, the de-facto standard for risk treatment is to use catalogues of appropriate security mechanisms, such as [4,6,13,19,22]. TRICK also implements the catalogue of standard security controls defined by ISO/IEC 27002 [13], and others. Therefore, a straightforward way to implement optimal countermeasure selection is to

consider such a catalogue of countermeasures, and to define an optimization problem on an attack-defense tree that maximizes the ROSI function.

Indeed, in practice an organization cannot implement all potential countermeasures, and often even implementation of the most critical security controls needs to be prioritized due to budget restrictions. Therefore, countermeasure selection needs to be guided by the *cost-benefit analysis*, in which we will consider costs of countermeasures versus their respective benefit (how well they can reduce attack probabilities).

5 Choices for Countermeasure Selection

Several choices are possible for selecting countermeasures. In this section we discuss these options in more detail.

Locality/Universality of Countermeasures. A countermeasure can be *local*, i.e., it has effect only on the attack node it has been applied to in the tree. In this case, if t is selected as a countermeasure for the attack node x, then it reduces the probability of occurrence of the sub-tree x, but does not influence the probability of occurrence of other attack nodes. However, this assumption does not preclude t from being selected as a countermeasure at another applicable attack node y, where it can then reduce the probability of occurrence (while inducing also extra cost of a separate countermeasure). This solution will work well for the cases when indeed separate security controls with the same name need to be introduced in different locations of the infrastructure. For instance, if there are two vulnerable doors that can be used by the attacker to get in, we will be able to propose two door locks as separate protection mechanisms.

Yet, if, for example, an attack tree has the attack nodes "infiltrate the network" and "probe the ports", and the countermeasure "firewall" is applicable to both of them, this countermeasure could be selected as a defense node twice in our solution (so the approach could propose to pay twice for the same firewall). Thus, an alternative is to assume countermeasures to be *universal*, meaning that they are applied once to the entire tree and affect all attack nodes, unless the effectiveness of a countermeasure on a given node has been set to zero (in this case this countermeasure is not shown in the tree). It is also possible to consider the combined approach, when some security controls are local, and some – universal.

Unique/Multiple Countermeasures of the Same Type. One option is to consider that each countermeasure can be applied to an attack node at most once. An alternative solution is to allow multiple identical countermeasures to be applied to the same node. Considering that each countermeasure is unique and can be applied at most once allows to avoid trivial solutions when cheap controls are applied several times. Furthermore, for the countermeasures defined in the ISO/IEC 27002 standard, it makes sense to only apply them once in a given context. Yet, certain defensive mechanisms can in fact improve protection if

applied multiple times (e.g., several security guards may be better than one, several locks on a door can be better than a single one).

Combinations of Defense Nodes. In general, catalogues suggest multiple countermeasures against a single attack node. However, the semantics of attack-defense trees only allow one single defense node per attack node [16]. To address this limitation, one can aggregate several applicable countermeasures into a meta-defense node for a given attack node. For example, we consider a combination of defense nodes, expressed as an **and**-refinement, to be added to the tree. Considering t and q to be two countermeasures (extension to the general case of k applicable countermeasures is trivial), we can add to the tree the defense node $t \wedge q$, with $\mathbf{eff}(t \wedge q) = 1$ - $\mathbf{eff}(t)\mathbf{eff}(q)$. Intuitively it means that both t and q simultaneously provide protection, but their effectiveness may not be fully independent. Furthermore, $\mathbf{cost}(t \wedge q) = \mathbf{cost}(t)+\mathbf{cost}(q)$.

Alternatively, meta-defense nodes can be expressed as an **or**-refinement. In this case, considering two applicable security controls t and q, the aggregated meta-defense node $t \vee q$ can be added to the tree, with $\mathbf{eff}(t \vee q) = 1-(1-\mathbf{eff}(t)) \cdot (1 - \mathbf{eff}(q))$. Again, $\mathbf{cost}(t \wedge q) = \mathbf{cost}(t)+\mathbf{cost}(q)$. The choice between these two types of aggregated meta-nodes depends on the interpretation one has for the defense nodes in the attack tree [16].

Defense Location-Sensitivity. If one considers countermeasures to be local, then actual position of the countermeasure in the tree becomes an important factor further contributing to the complexity of the considered problem. We can demonstrate that if a countermeasure t is applicable to both attack nodes x and $x \vee y$ (what is very likely for attack trees expressed in natural language), then assigning t to the parent node provides a better reduction of the risk. Indeed, with the countermeasure assigned to the parent node $x \vee y$, $\mathbf{Pr}(c^p(x \vee y, t)) = \mathbf{Pr}(x \vee y)(1\text{-}\mathbf{eff}(t)) = (\mathbf{Pr}(x) + \mathbf{Pr}(y)$ - $\mathbf{Pr}(x)\mathbf{Pr}(y))(1\text{-}\mathbf{eff}(t))$. In case t is allocated with the child node x, we have $\mathbf{Pr}(c^p(x, t) \vee y) = \mathbf{Pr}(x)(1\text{-}\mathbf{eff}(t)) + \mathbf{Pr}(y)$ - $\mathbf{Pr}(x)(1\text{-}\mathbf{eff}(t))\mathbf{Pr}(y)$. It is evident that $\mathbf{Pr}(c^p(x \vee y, t))$ - $\mathbf{Pr}(c^p(x, t) \vee y) =$ -$\mathbf{Pr}(y)\mathbf{eff}(t) \leq 0$, given that $0 \leq \mathbf{eff}(t), \mathbf{Pr}(y) \leq 1$. Therefore, the closer to the root we place a defense, the better it can reduce the overall probability of the considered attack.

We discuss the choices we have made for our implementation and the optimization problem to be solved in the following section.

6 Attack Tree Refinement and Optimization Problem

Assumptions Made on Countermeasure Selection. In our current implementation we assume each security control to be universal. Thus, for each attack node x and each countermeasure t, such that t is applicable to x ($\mathbf{eff}(t, x) > 0$), we consider that t can be applied to x as a defense node everywhere it is applicable. Furthermore, we consider that each countermeasure, if selected, is applied exactly once. These considerations imply that the total cost of each countermeasure is not affected by the number of times this countermeasure appears in the

attack-defense tree (it is counted only once). We also consider that aggregated meta-nodes are composed by the $\vee$-refinement.

The process to refine an estimation of probability for an asset-scenario pair and to find the optimal set of countermeasures is as follows.

A. Assess Input Parameters.

1. *Create an attack tree.* Model the step or variant of the attack and describes them in a pure attack tree at that does not contain any defence notes. Estimate the success probability of each leaf node. Let n be the number of attack nodes in the initial attack tree. Let a_j denote the j-th node in this attack tree. The ADTool [8,15] can be used to compute $\mathbf{Pr}(at)$, which is the success probability of the root note; it depends on the attack tree and the probabilities of the leaf nodes.
2. *Identify countermeasures.* Prepare the list of potentially applicable countermeasures from catalogues. Let m is the number of countermeasures in this list. For each countermeasure, estimate the security implementation costs.
3. *Estimate effectiveness values.* Estimate the value of the $(m \times n)$ *effectiveness matrix* $\mathbf{E}$ indicating the effectiveness of a countermeasure i on an attack node j. We define $\mathbf{E}[i, j] = \mathbf{eff}(i\text{-th countermeasure}, j\text{-th attack})$.

B. Solve the Optimization Problem. A possible solution of the problem is described by $d = (d_1, d_2, ..., d_m)$, an m-tuple indicating for each countermeasure whether each corresponding countermeasure i will be implemented (if $d_i = 1$) or not ($d_i = 0$). The cost of such a solution is given by $\mathbf{cost}(d) = \sum_{i=1}^{m}(d_i \times \mathbf{cost}(\text{countermeasure } i))$.

Remark that we can have meta-defense nodes. Let the meta-defense node t expresses the combined defenses applicable to the node a_k. Then $\mathbf{eff}(t, a_k) = 1 - \prod_{j=1}^{m}(1 - d_j \times \mathbf{E}[j, k])$. In the attack tree language, this defense node is a node consisting of an $\vee$-refinement of the selected countermeasures ($d_j = 1$ and $\mathbf{E}[j, k] > 0$).

The Return On Security Investment (of the list of selected countermeasures d) is defined as follows.

$$ROSI(d) = \mathbf{Impact} \cdot (\mathbf{Pr}(at) - \mathbf{Pr}(adt_d)) - \mathbf{cost}(d),$$

where the **Impact** is the loss achieved if the attack succeed (i.e., if the root node of the attack tree occurs), adt_d is the new attack-defense tree in which the countermeasures selected by d have been added to the nodes according to the effectiveness matrix $\mathbf{E}$. Notice that adt can be constructed from at, d, and $\mathbf{E}$. The ADTool can be now used to compute $\mathbf{Pr}(adt_d)$.

> Our optimisation problem consists in finding the list of the selected countermeasures d that maximizes ROSI(d).

Note that instead of maximizing ROSI(d), we can as well minimize $\mathbf{Impact} \cdot \mathbf{Pr}(adt_d) + \mathbf{cost}(d)$.

Current Implementation. Our current implementation uses a branching algorithm based on multiple parameters. We use a brute-force algorithm to find the optimal d, by trying all 2^m possible sets of countermeasures to implement. Our tool called *ADTop* will be published as open-source.

General Optimization Problem. Notice that the generalized optimization problem for selecting countermeasures (considering various assumptions discussed in Sect. 5) can be also solved by applying the approaches from [2,25]. To apply these algorithms under the assumption of *local* countermeasures, we can consider all security controls that have positive effectiveness and their combinations as candidate defense nodes. Furthermore, in case of [2], we will also need to evaluate the resulting set of Pareto-efficient trees, and to select the one that gives the global optimum to the ROSI function.

7 Illustration on a Use Case

We have applied our approach to a use case scenario of a private cloud attack. The target of this scenario is a small/medium size enterprise (SME) with ten employees sharing confidential documents, such as audit reports, studies, and internal documents of customers. To allow continuous remote access to all documents, they are made available on a private cloud accessible via VPN and installed in the SME's IT room. Suppose that stealing these documents will create a damage of 100.000 €.

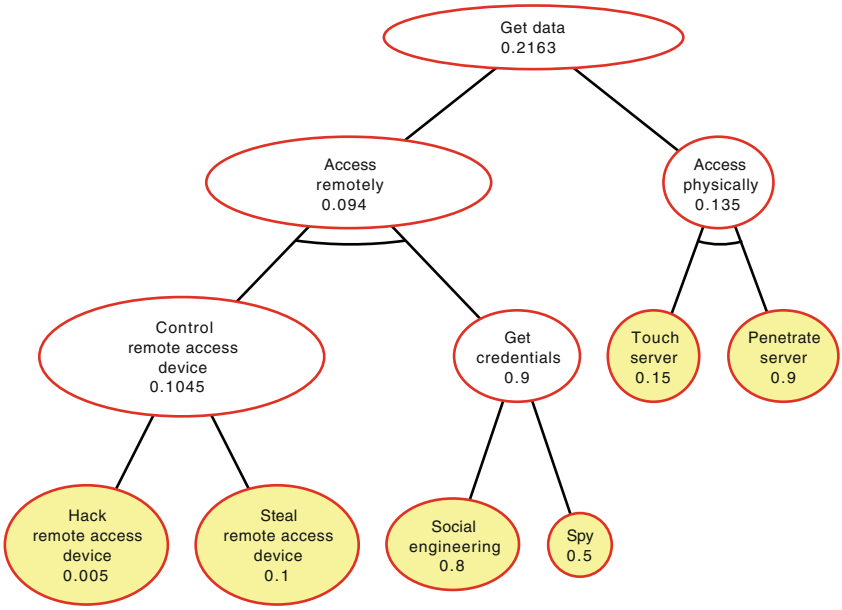

Fig. 1. Initial attack tree with success probabilities for our private cloud attack use case.

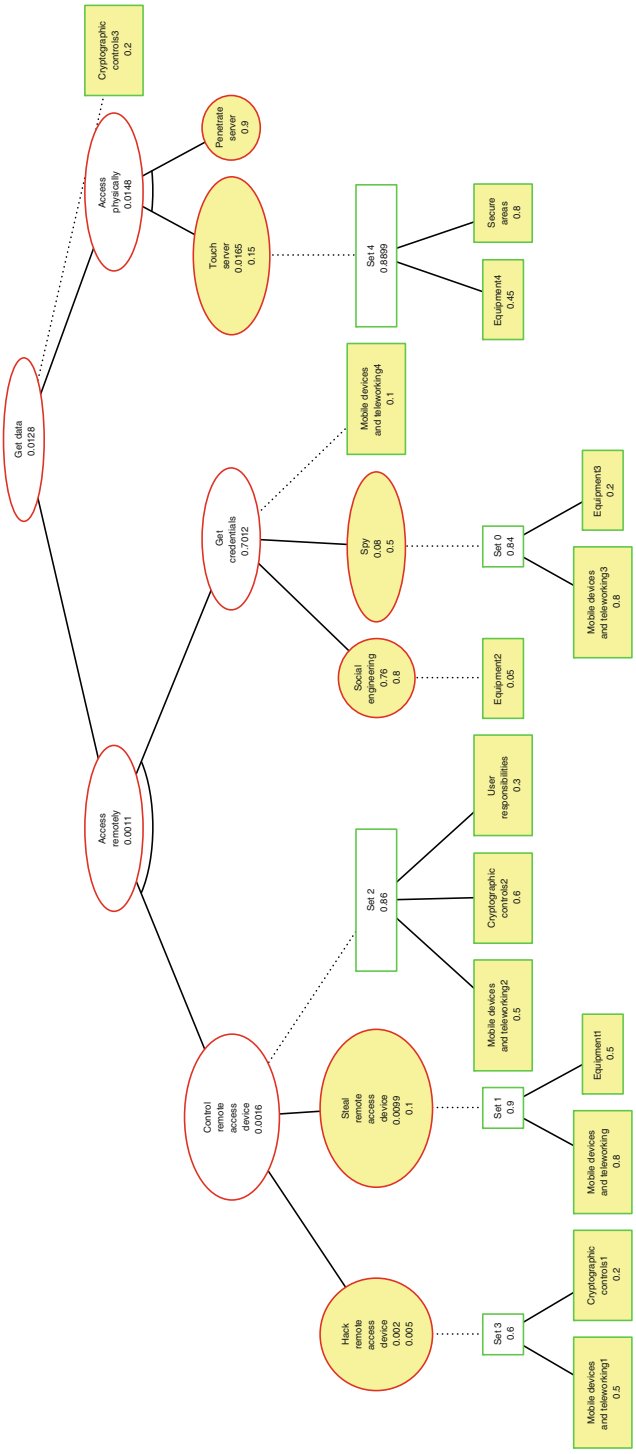

Fig. 2. Optimal attack-defense tree of our use case of the private cloud attack.

Table 1. The effectiveness values and implementation costs of countermeasures.

	Control remote access device	Get credentials	Get data	Social engineering	Penetrate server	Access physically	Access remotely	Hack remote access device	Spy	Steal remote access device	Touch server	Cost in k€
Management direction for inf. security	0.8			0.8					0.2	0.8		3.3
Internal organization			0.3			0.1						0.66
Mobile devices and teleworking	0.5	0.1						0.5	0.8	0.8		0.45
Prior to employment												0.1
During employment												0.2
Termination and change of employment												0.05
Responsibility for assets												0.375
Information classification												0.33
Media handling								0.15				0.4
Business requirements of access control	0.1	0.2						0.05		0.05	0.4	0.96
User access management							0.2			0.2	0.3	2.16
User responsibilities	0.3											
System and application access control	0.7	0.55		0.1				0.7	0.5			1.86
Cryptographic controls	0.6		0.2					0.2				0.5
Secure areas											0.8	0.27
Equipment				0.05					0.2	0.5	0.45	0.53
Operational procedures and responsibilities												3.96
Protection from malware	0.4	0.05		0.1				0.5				0.8
Backup												
Logging and monitoring												0.18
Control of operational software	0.1							0.15				0.2
Technical vulnerability management	0.35	0.05						0.3	0.05			0.66
Information systems audit considerations												0.312
Network security management								0.5				0.225
Information transfer												0.156
Sec. requirements of information systems	0.04	0.04		0.05				0.05	0.05			0.4
Sec. in development and support processes								0.05				0.78
Test data												
Inf. security in supplier relationships												0.8
Supplier service delivery management												
Management of incidents												0.6
Information security continuity												1.2
Redundancies												
Compliance with laws and contracts												0.3
Information security reviews												2.4

Figure 1 presents the initial attack tree we produced for the considered use case. It can be read as follows. To steal data, the attacker can remot7ely or physically access the cloud file server. To access remotely, the attacker needs to gain control of the remote access device and get the credentials to connect. To gain control of the device, the attacker can hack it (which happens at a success probability of 0.5 % within a timeframe of one year), or he/she can steal it (success probability of 10 %). To get credentials, the attacker can make the user to disclose them via social engineering (80 %), or, additionally to the

hacking, he/she can spy on the privileged user, e.g., by installing a key logger, or, before stealing the remote access device, by spying on the keyboard, e.g., via shoulder surfing (50 %). To access physically, the attacker needs to touch the server(15 %) and to penetrate it, e.g., by plugging a USB stick or accessing the hard disk (90 %). The probabilities were estimated by the customer, for her implementation. The overall success probability of the root node "Get data" is 21.63 %, computed in the ADTool.

We consider as potential countermeasures the objectives taken from the ISO/IEC 27002 standard (see Table 1). The customer has partially implemented them, and has estimated the security implementation costs to achieve full compliance to these objectives. We have evaluated the effect of these countermeasures on each attack node of the initial attack tree by filling the matrix **E**, which was filled for the eleven attack nodes and the thirty-five countermeasures, i.e., the thirty-five objectives of the ISO/IEC 27002 standard. We identified the objectives without any effect on the attack nodes and removed them, reducing the complexity of the algorithm from 2^{35} to 2^{17} attack-defense trees to consider. The optimal attack-defense tree adt_{opt} found by our implementation is presented in Fig. 2.

Figure 3 shows a screenshot of our ADTop tool that implements the approach described in this paper. The optimal attack-defense tree adt_{opt} found by ADTop has the residual success probability for the attacker reduced to 1.28 % (instead of the initial 21.63 %). The optimization function is computed as **Impact**· **Probability**(adt_{opt}) + **cost**(selected countermeasures). For the optimal attack-defense tree it is 100,000€ · 0.0128 + 1750 = 3030. The corresponding ROSI is **Impact**· (**Probability**(at) - **Probability**(adt_{opt})) - **cost**(selected countermeasures) = 100,000€ · (0.2163 - 0.0128) - 1750€ = 18,600€.

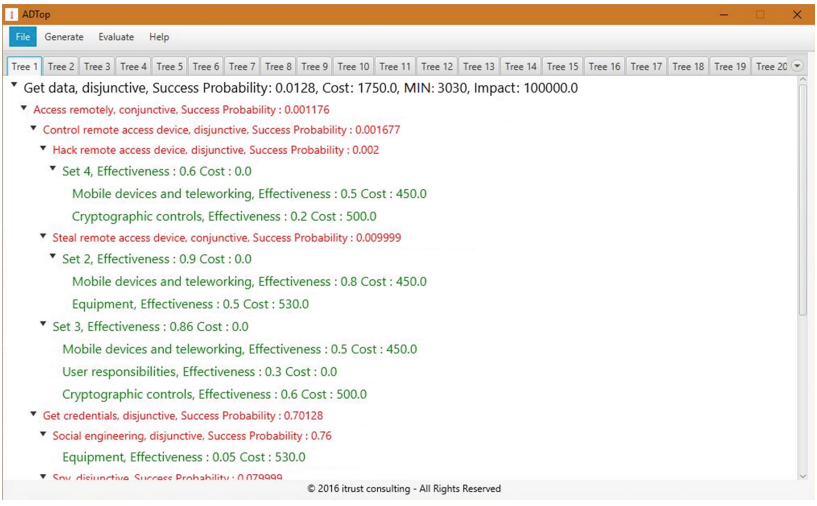

Fig. 3. Screenshot of the ADTop tool.

8 Next Steps and Conclusions

In this position paper we have argued that there is a gap between practical risk assessment methods and academic research. This gap explains why, on the one hand, the practical impact of academic results is somewhat limited, while, on the other hand, practical risk assessment methods do not include state-of-the-art scientific results. Various factors influence this discrepancy. An example is the use of different ontologies, leading to different interpretations of used notions, such as combined defensive mechanisms (meta-defense nodes). Another possible factor is implied by the fact that practical risk assessment methodologies often have a wider scope than specific academic developments, which leads to an interfacing problem between the two.

We argue that an important step forward can be made by bridging this gap through extending practical methods with recent academic results. As an example, we have looked at the extension of the TRICK methodology with recent results on optimal countermeasure selection. In order to do so, we had to agree on a common terminology and had to relate practical design details (like countermeasure catalogues) to academic concepts (like attack-defence trees). In this paper we provided a high-level description of the proposed extension of TRICK and a special algorithm which has been implemented and is being tested in the context of cloud security.

The next steps will focus on improving scalability by designing a better optimization algorithm, and assessing whether the attack-defense-refined risk assessment can be considered more reliable by the risk managers than the established ALE in TRICK Service. Another future extension of this work is to consider attack trees and attack-defense trees automatically generated from some system model [7] as the starting point, instead of manually designed attack trees. This will allow us to integrate the system also with the recent TREsPASS methodology for assisted risk assessment [27].

References

1. Albanese, M., Jajodia, S., Noel, S.: Time-efficient and cost-effective network hardening using attack graphs. In: 2012 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 1–12. IEEE (2012)
2. Aslanyan, Z., Nielson, F.: Pareto efficient solutions of attack-defence trees. In: Focardi, R., Myers, A. (eds.) POST 2015. LNCS, vol. 9036, pp. 95–114. Springer, Heidelberg (2015)
3. Bistarelli, S., Fioravanti, F., Peretti, P.: Defense trees for economic evaluation of security investments. In: The First International Conference on Availability, Reliability and Security, 2006 ARES 2006, pp. 8-pp. IEEE (2006)
4. Bundesamt fur Sicherheit in der Informationstechnik: IT-Grundschutz-Catalogues, 13th version (2013)
5. Edge, K.S., Dalton, G.C., Raines, R.A., Mills, R.F., et al.: Using attack and protection trees to analyze threats and defenses to homeland security. In: Military Communications Conference 2006. MILCOM 2006, pp. 1–7. IEEE (2006)

6. European Organization for Safety of Air Navigation: Threats, Pre-controls and post-controls catalogues (2009)
7. 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
8. Gadyatskaya, O., Jhawar, R., Kordy, P., Lounis, K., Mauw, S., Trujillo-Rasua, R.: Attack trees for practical security assessment: ranking of attack scenarios with ADTool 2.0. In: Agha, G., Van Houdt, B. (eds.) QEST 2016. LNCS, vol. 9826, pp. 159–162. Springer, Heidelberg (2016). doi:10.1007/978-3-319-43425-4_10
9. Harpes, C., Adelsbach, A., Zatti, S., Peccia, N.: Quantitative risk assessment with ISAMM on ESA's operations data system. In: Proceedings of TTC (2007)
10. ISO: 27799:2008 Health Informatics - Information security management in health using ISO/IEC 27002 (2008)
11. ISO, IEC: 27005:2011 Information technology Security techniques Information security risk management (2011)
12. ISO, IEC: 27001:2013 Information technology - Security techniques - Information security management systems - Requirements (2013)
13. ISO, IEC: 27002:2013 Information technology Security techniques Code of practice for information security controls (2013)
14. ISO, IEC: TR 27019:2013 Information technology Security techniques Information security management guidelines based on ISO/IEC 27002 for process control systems specific to the energy utility industry (2013)
15. 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)
16. Kordy, B., Mauw, S., Radomirović, S., Schweitzer, P.: Attack-defense trees. J. Logic Comput. **24**(1), 55–87 (2014)
17. 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)
18. NATO Research and Technology Organisation (RTO): Improving common security risk analysis (2008)
19. NIST: Special Publication 800–53 Revision 4. Security and privacy controls for federal information systems and organizations (2013). http://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-53r4.pdf
20. NIST: Framework for Improving Critical Infrastructure Cybersecurity (2014)
21. OWASP: CISO AppSec Guide: Criteria for managing application security risks (2013)
22. PCI Security Standards Council: Payment Card Industry Data Security Standards (PCI DSS) (2016). https://www.pcisecuritystandards.org/
23. PWC: The global state of information security survey (2016). http://www.pwc.com/gx/en/issues/cyber-security/information-security-survey.html
24. Refsdal, A., Solhaug, B., Stølen, K.: Cyber-Risk Management. Springer Briefs in Computer Science. Springer International Publishing, Heidelberg (2015)
25. Roy, A., Kim, D.S., Trivedi, K.S.: Scalable optimal countermeasure selection using implicit enumeration on attack countermeasure trees. In: Proceedings of the 42nd Annual IEEE/IFIP International Conference on Dependable Systems and Networks, pp. 299–310. IEEE (2012)
26. Schneier, B.: Attack trees. Dr. Dobb's J. Softw. Tools **24**, 21–29 (1999)
27. TREsPASS: Technology-supported Risk Estimation by Predictive Assessment of Socio-technical Security (2016). http://www.trespass-project.eu/