LiSRA: Lightweight Security Risk Assessment for Decision Support in Information Security

Christopher Schmitz and Sebastian Pape

Goethe University Frankfurt, Germany
{christopher.schmitz, sebastian.pape}@m-chair.de

Abstract

Information security risk assessment frameworks support decision-makers in assessing and understanding the risks their organisation is exposed to. However, there is a lack of lightweight approaches. Most existing frameworks require security-related information that are not available and that are very challenging to gather. So they are not suitable in practice, especially for small and medium-sized enterprises (SMEs) who often lack in data and in security knowledge. On the other hand, other explicit SME approaches have far less informative value than the proposed framework. Moreover, many approaches only provide extensive process descriptions that are challenging for SMEs. In order to overcome this challenge, we propose LiSRA, a lightweight, domain-specific framework to support information security decision-making. It is designed with a two-sided input where domain experts initially provide domain-specific information (e.g. attack scenarios for a specific domain), whereupon users can focus on specifying their security practices and organisational characteristics by entering information that many organisations have already collected. This information is then linked to attack paths and to the corresponding adverse impacts in order to finally assess the total risk. Moreover, LiSRA can be used to get transparent recommendations for future security activities and presents detailed insights on the mitigating effects of each recommendation. The security activities are being evaluated taking into account the security activities already in place, and also considering the dependencies between multiple overlapping activities that can be of complementary, substitutive or dependent nature. Both aspects are ignored by most existing evaluation approaches which can lead to an over-investment in security. A prototype has been implemented, and the applicability of the framework has been evaluated with performance and robustness analyses and with initial qualitative evaluations.

Keywords: security risk assessment, decision support, attack trees, maturity levels, security controls, ISO/IEC 27001

1. Introduction

Frameworks for information security risk assessment play a major role in the daily routines of decision-makers in information security. They are used to systematically assess the organisational security risk and to better understand the risks an organisation is exposed to. A solid risk assessment also builds the basis for an information security management system (ISMS). Otherwise, decision-makers will not be able to allocate their finite resources efficiently.

However, security risk assessment is a challenging task that normally requires a deep understanding of the relevant attack scenarios and technical knowledge about the mitigating effects of all the implemented security measures in the organisation. This poses a challenge especially for small and medium-sized enterprises (SMEs) that often do not have the capacities to run a fully-fledged information security department. Due to smaller IT budgets they often have a lack in security expertise and security-related data. Thus, most information security risk assessment frameworks are not suitable for them. Although there exist explicit SME approaches they have far less informative value than the proposed framework [1, 2]. Besides that, many approaches only present extensive process descriptions and guidelines that are challenging for SMEs [3, 4].

To address these issues, we propose LiSRA, a lightweight, domain-specific framework for decision support in information security. LiSRA is designed with a particular focus on the special needs for SMEs. Therefore, a key requirement is to mainly use already existing data and to keep the user's input to a minimum but to en-

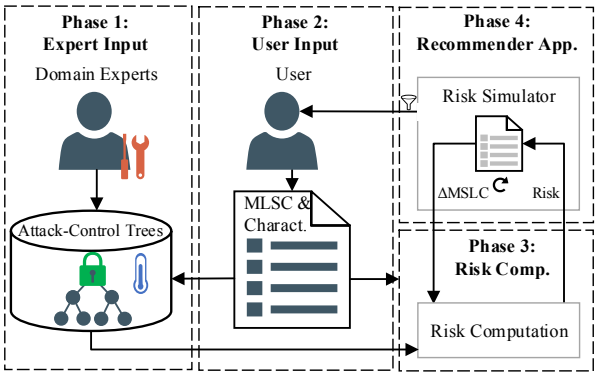


Figure 1: Overview

sure good analysis results at the same time. To meet the requirements, LiSRA expects input from both users and domain experts who are associated with the platform provider (that hosts the proposed LiSRA framework as a web-based application). The general concept is illustrated in Fig. 1. The framework assumes that organisations within a particular domain are basically exposed to similar attacks[1]. Domain experts with in-depth security knowledge for a particular domain (e. g. the electric sector) initialise the framework by providing domain-specific information (e. g. attack scenarios for a specific domain) so the user can concentrate on completing an easy-to-answer questionnaire to specify the implementation status of their security practices (that are represented by security controls).

For many organisations this only causes little extra effort because they have already collected these information. LiSRA links this information with attack trees – a well-known formalism to represent attack scenarios in a tree-based structure where high-level attack goals are decomposed into attack steps using an AND–OR tree structure [6]. They are used to calculate to which degree the implemented security controls protect against a set of attack scenarios in order to finally assess the scenario risks as well as the total risk. LiSRA can also be used to get transparent recommendations for future security activities that also provide detailed insights on their mitigating effects and how to implement them in an effective way. The term "*security activity*" is used in the sense of increasing the maturity level of a security control. Most existing approaches evaluate new security activities in isolation of security actitivies already in place, and they ignore that multiple overlapping activities can be of complementary, substitutive, or dependent nature which leads to an over-investment in security measures [7]. LiSRA explicitly addresses both aspects without bothering the user.

To further ease the data entering for the users the framework has been integrated into a web-based security management platform which eases the burden of going through a longer questionnaire. This is achieved for example by making use of small modules that are spread across the platform. They allow the users to complete or update the data needed for the risk assessment along the way when interacting with other parts of the platform [8]. Alternatively, if the data is already digitally available, e.g. as the output of an ISMS, it can also be easily imported.

The remainder of this paper is organised as follows. Section 2 presents the LiSRA framework along with a brief description of its implementation. In Section 3 an example is shown which demonstrates the framework's ease of use in the electric sector as an exemplary domain. Section 4 presents the evaluation and reports about limitations, Section 5 presents the related work, and Section 6 finally concludes and points out future research ideas.

2. LiSRA: Lightweight Security Risk Assessment

LiSRA is a lightweight security risk assessment framework for decision support in information security aiming to overcome the mentioned challenges. It models the organisation's security activities in a lightweight manner and links them with attack scenarios and their adverse impacts in order to measure the security risks. This approach can also be used to identify beneficial future security activities taking into account the effects of overlapping security activities. The framework consists of four phases:

a) *Phase 1: Expert Input.* In the first phase domain experts initially set up the framework for particular domains (e.g. the electric sector) by constructing parameterised attack trees that are linked to security controls. In a later step the user can select the domain in which his organisation operates so that the risk assessment only considers attack trees that are relevant for the respective domain. The required steps for this are illustrated in the flow chart depicted in Fig. 3 and are further described in Sect. 2.1.

b) *Phase 2: User Input.* The only user inputs required are the maturity levels of the organisation's secu-

[1]The National Electric Sector Cybersecurity Organization Resource (NESCOR) [5] for example gives an overview of domain-specific attack scenarios for the electric sector

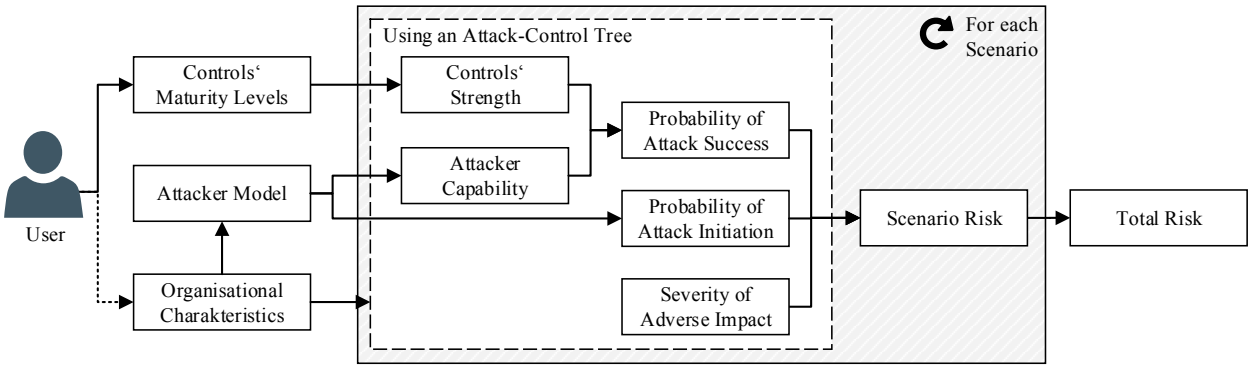


Figure 2: General Risk Computation Process

rity controls. They are used to model the implemented security practices of the organisation in a lightweight manner (see Sect. 2.2).

c) *Phase 3: Risk Computation.* Before the risk computation can start the control dependencies are resolved. This is needed because the effective maturity levels may be lower than the actual maturity levels due to control dependencies. The general risk computation process is illustrated in Fig. 2. First, the total risk is derived from scenario risks that are calculated based on both the probability of adverse impact and its severity. The probability of adverse impact is the probability that an attack is initiated and succeeds. Both factors are calculated using attack trees. The details are explained in Sect. 2.3.

d) *Phase 4: Recommender Application.* The recommender application identifies the most effective and the most cost-efficient security activities. Further information is provided in Sect. 2.4.

Since LiSRA deals with attacker behaviour, assumptions with respect to the attacker model have to be made. Here, a rational attacker is assumed to follow a best-shot strategy and always chooses the attacks and attack steps maximising his utility (according to pre-defined attacker models).

2.1. Phase 1: Expert Input

In phase 1 experts initially set up the framework for a particular domain. They gather relevant attack scenarios, transform them into a tree structure (attack trees) and link them with the respective security controls. This tree-based structure is defined as attack-control tree (ACTree) that enables determining to which extent the implemented security controls protect against attack scenarios and their associated adverse impacts. Finally, the attack-control trees are parameterised in such a way that they reflect the efficacy of controls and the attack costs. The required actions are illustrated in Fig. 3 and are described in detail in the following sections.

To make sure the system is up-to-date experts update the data at regular intervals (e.g. once per quarter) and also irregularly if the threat situation has changed significantly.

2.1.1. Identifying Attack Scenarios

The very first step is to identify the relevant attack scenarios (see A1 in Fig. 3). Experts identify both domain-independent scenarios (general attacks like malware or phishing attacks) and specific scenarios (e. g. attacking smart meters for the electric sector) for all domains that should be covered. The user can later select the domain in which his organisation operates so that the risk assessment only takes into account the relevant attack scenarios. So each domain-specific scenario has to be explicitly linked to one or more domains. It is essential to identify scenarios for both domain-specific scenarios (e. g. attacking smart meters for the electric sector) as well as for domain-independent scenarios (general attacks like malware or phishing attacks) because all organisations are exposed to general attack.

Domain experts typically already have a collection of attack scenarios because most risk assessment approaches in security management are scenario-based. So the processes will in most cases not take much time.

2.1.2. Assessing Adverse Impact

The next step is to assess the scenario's adverse impact, $I_s \in [0, 1]$ (see A2 in Fig. 3). The impact assessment is an essential factor in risk computation because it reflects the probable loss that can be expected by an attack scenario. In common and widely used frameworks

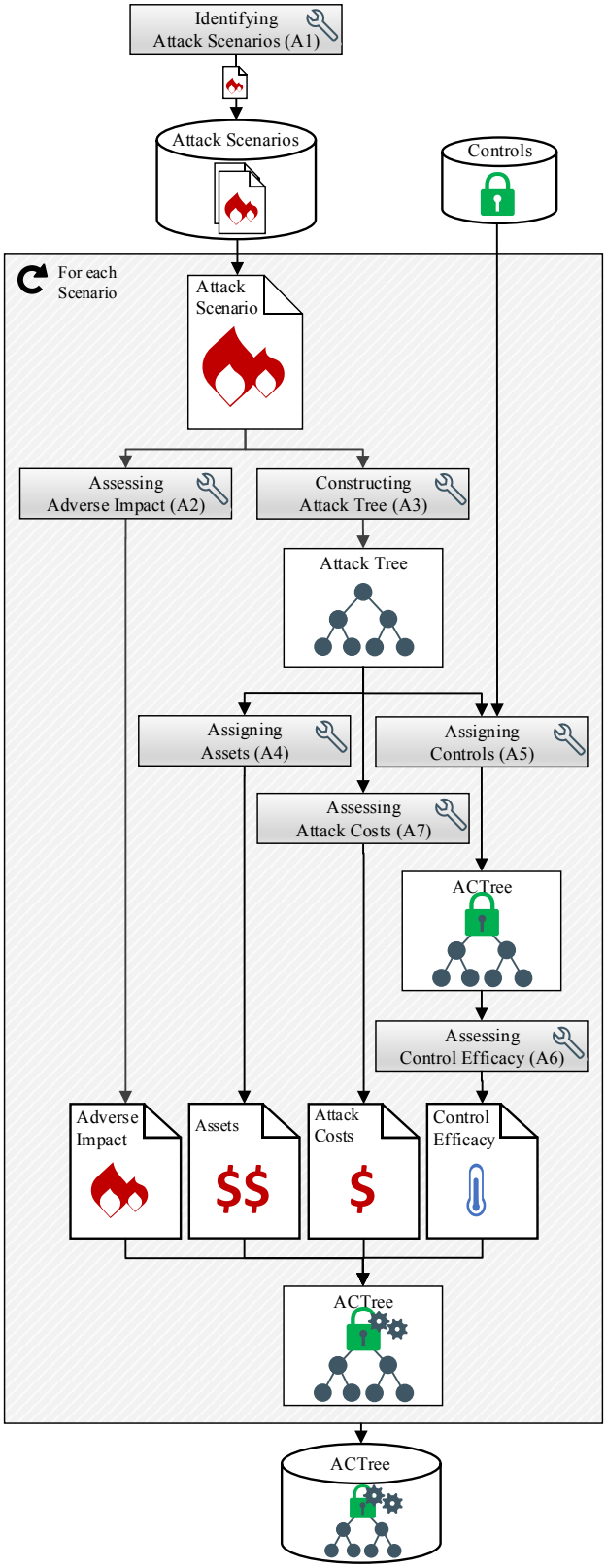


Figure 3: Expert Input – Constructing Attack-Control Trees

(such as ISO/IEC 27005 [9] and NIST SP 800-30 [10]) the impact is assessed with similar 5-point scales. Using scales the experts are familiar with eases the process of impact assessment and supports the reliability of the expert input. Besides that, it also improves the input accuracy. Therefore, the proposed LiSRA framework also uses a 5-point scale for the impact assessment. It uses the NIST impact assessment scale because it is an established scale that also contains a textual description for each impact level (in contrast to the scale used in ISO/IEC 27005). The scale is depicted in Tab. 1. For several domains there exist domain-specific, scenario-based impact assessment methods. It can make sense to combine LiSRA with one of those methods in order to further refine the analysis results.

2.1.3. Constructing Attack Trees

The identified attack scenarios are then transformed into attack trees (see A3 in Fig. 3). Attack trees are an established method in threat and risk analysis to systematiclly analyse possible attack paths [11, 12]. They decomposed a high-level attack goals into single attack steps using logical AND–OR operations. Kordy et al. give a structured overview of the numerous existing variations [13].

Before constructing the trees from scratch it is recommended to follow best practices on model creation and to make use of attack pattern libraries or shared attack trees. The TREsPASS project, for instance, addressed these topics [14]. Furthermore, NESCOR provides a list of common subtrees (such as "Threat agent gains access to network") that can easily be integrated into general attack scenarios [15].

The attack trees used by LiSRA basically follow the definition of the defence trees introduced by Bistarelli et al. in 2006 [16]. The only difference is that the attack trees are extended by security controls[2] instead of concrete security measures. This modification is needed to be able to link the user's implementation status for specific security activities (defender perspective) with attacker activities (attacker perspective). For this, the vast amount of possible security measures had to be reduced by using an assessable number of roughly more than 100 security controls.

Similar to the approach by Bistarelli et al., all attacker activities are represented in leaf nodes. This does not pose a limitation because other attack tree representations where the attacker activities (and therefore the attack costs) are located in inner nodes (like ACTs by Roy

[2] A security control describes a set of security measures for the fulfillment of a security requirement.

Table 1: Impact Assessment Scale based on NIST [10]

Qualitative Values	Quantitative Values	Description
Severe	1	The attack event could be expected to have multiple severe or catastrophic adverse effects on organizational operations, organizational assets, individuals, other organizations, or the Nation.
Major	0.8	The attack event could be expected to have a severe or catastrophic adverse effect on organizational operations, organizational assets, individuals, other organizations, or the Nation.
Medium	0.5	The attack event could be expected to have a serious adverse effect on organizational operations, organizational assets, individuals other organizations, or the Nation.
Minor	0.2	The attack event could be expected to have a limited adverse effect on organizational operations, organizational assets, individuals other organizations, or the Nation.
Negligible	0	The attack event could be expected to have a negligible adverse effect on organizational operations, organizational assets, individuals other organizations, or the Nation.

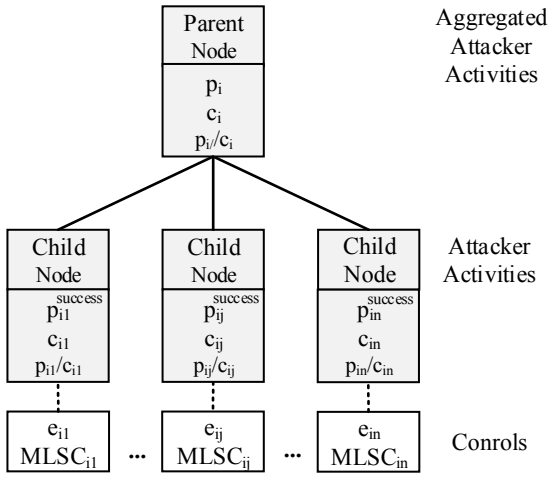


Figure 4: Parameter Notation for Attack-Control Trees

et al. [17] can easily be transformed into this representation.

The parameter notation for ACTrees, that are used in the following, is illustrated in Fig. 4 which visualises the nodes' parameters and indices. Each parent node i has a set of child nodes $j \in J$. This also holds for an attacker activity i that is assigned to j controls.

2.1.4. Assigning Assets

When the attack trees are constructed assets are assigned to corresponding nodes in the ACTree (see A4 in Fig. 3). The mapping between assets and attack steps enables to know which attacks or attack steps require the presence of which assets to be successfully performed. It is used in a later step to individualise the attack trees. We focus on the supporting assets according to ISO/IEC 27005 because "these assets have vulnerabilities that are exploitable by threats aiming to impair the primary assets of the scope (processes and information)" [9]. So attackers have to attack these "supporting assets" in the first place in order to achieve their attack goal. Therefore, an organisation that does not work with a specific asset class is not exposed to the corresponding attacks. For example, an organisation that does no work with respectively does not store any (sensitive) information on the asset class "database server" is not exposed to the attack "data theft through SQL injection". The attack step "data theft through SQL injection" can then be eliminated from the attack tree as described in detail later.

The ISO/IEC 27005 asset list is predestined for this purpose because it presents a fine-grained overview of various asset classes covering all kind of possible attack targets in information security. Besides technical categories like hardware and software it also considers non-technical categories like personnel. However, particularly for domain-specific attack scenarios it makes sense to refine these assets with respect to attack-relevant characteristics. For example, the asset class smart meter (which is relevant for the electric sector) could be differentiated with respect to the supported remote data transmission standard (GSM / GPRS, WiFi, Bluetooth, Ethernet etc.). So an energy provider that does not use any smart meter supporting a WiFi transmission is not exposed to the respective attacks.

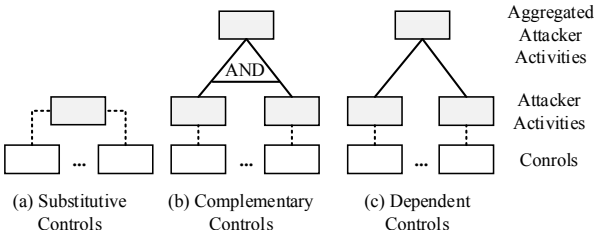


Figure 5: Modelling Rules for Controls

2.1.5. Assigning Controls

Then, security controls are assigned to the attack trees in order to get ACTrees that link the defence and the attacker perspective (see A5 in Fig. 3). ACTrees thereby enable determining to which extent the implemented security controls protect against attack scenarios and their associated adverse impacts. The starting point for assigning controls to the attacker activities is a controls list from an established standard (e. g. ISO/IEC 27002 [18]). Depending on the analysis scope a more specific standard (e. g. ISO/IEC 27019 [19] for the electric sector) can further improve the analysis results. Since the ISO/IEC 27002 standard covers roughly 110 security controls, their maturity levels can be assessed by most SMEs within a reasonable time. Although the LiSRA framework was designed with special consideration for the ISO/IEC 27000 series it also compatible with other control lists. There also exist many mappings between different control catalogues.

When assigning controls to attacker activities, the controls' relationship to each other must be considered to avoid an over-investment in security. We differentiate three relationship types: substitutive controls, complementary controls and dependent controls. The respective modelling rules, that are described in the following, are illustrated in Fig. 5.

- An example for substitutive controls for the attacker activity "password guessing" are the controls "information security awareness, education and training" (control 7.2.2) and "use of secret authentication information" (control 9.3.1). Both of them address aspects of password quality, although the latter one has a higher efficacy in this case. A set of substitutive controls is as strong as its best control because the best control takes effect. All substitutive controls are directly assigned to the correspondent attacker activity as illustrated in Fig. 5 (a).

- Complementary controls complement each other in improving the security. The highest security level can be achieved when both of them are implemented. For example, "information backup" (control 12.3.1) and "controls against malware" (control 12.2.1) complement each other in the protection against ransomware attacks. They are independent and have a multiplicative effect. Complementary controls are always linked with AND operations (to be treated multiplicative as described later) because attackers necessarily have to attack both of them for a successful attack. For this, intermediate attacker activities need to be added as shown in Fig. 5 (b).

- Dependent controls reflect a relationship type where the controls are only as good as the weakest control. An example is the relationship between a "physical security perimeter" (control 11.1.1) and an "access control policy" (control 9.1.1). A very refined and mature physical security perimeter control for instance can be useless if there is no access policy control in place, and vice versa. Dependent controls for an attacker activity are always modelled with OR operations where a rational attacker always chooses the weakest control (more details are described later). Here, intermediate nodes need to be added, too (see Fig. 5 (c)).

  There already exist lists of control dependencies for the ISO/IEC 27002 standard that can be used in the construction process of ACTrees [20].

Modelling rules for more complex relationship types like synergetic controls (that together produce an effect greater than the sum of their individual effects) are not considered here.

2.1.6. Assessing Control Efficacy

When the ACTrees are constructed they are parameterised, starting with the control efficacy (see A6 in Fig. 3). This parameter is determined for each "control to attacker activity" relation in the ACTree (see Fig. 4.

It reflects how effective a control will averagely (in the considered domain) protect against an attacker activity when it is correctly implemented.

For example, even a very mature security awareness program might be very effective in training employees to recognise phishing attacks but it might be much less effective against more specific and sophisticated attacks. This illustrates that the parameter is independent from the actual implementation level of a control. The experts assess the control efficacy based on experience and knowledge using a 3-point scale (low (L), medium (M), high (H)), which is subsequently mapped to [0,1]. *High*

is mapped to 1, *medium* to 0.67 and *low* to 0.33. A control efficacy of 0 would simply mean the control should be removed from the model. So the efficacy for an attacker activity i and a control j is $e_{ij} \in [0, 1]$.

2.1.7. Assessing Attack Costs

The same applies for the attack costs. Here, the term "attack costs" is not defined in a purely monetary sense but also in the sense of required resources. In practice, it can be a challenging task to assess the attack costs using a fine-grained scale. Therefore, the attack costs are estimated by the experts using the same 3-point scale (L/M/H) and the same mapping to a [0,1] scale that is used for the control efficacy, too (see A7 in Fig. 3). The attack costs are estimated for each attacker activity (that are defined in the leaf nodes). It is assumed that no attack can be performed for free. In the risk computation phase, the attack costs are aggregated up the tree according to the assumed attacker model. The details are described in Sect. 2.3.

2.2. Phase 2: User Input

When the framework has been set up by the domain experts users can specify their security practices and organisational characteristics.

2.2.1. Assessing Maturity Levels

The security practices are represented by the organisation's maturity levels of the security controls (MLSC). The maturity levels are used as a measure to quantify the implementation status of a security control. The higher the maturity level of a control, the higher is the chance that it is performed in an effective and secure way so that it contributes more to the organisational security. In the following, the COBIT maturity levels are used that are also defined in the ISO/IEC 15504 standard [21, 22]. Since the COBIT framework is used widespread in industry many security experts are familiar with its maturity levels and even use them in practice. For example, the information security assessment questionnaire from the German Association of the Automotive Industry (VDA) is also based on maturity levels of security controls following ISO/IEC 27002 and has a very high degree of acceptance within the German Automotive Industry [23]. So many organisations have already gathered these information. Furthermore, there also exist mappings between different control catalogues. The COBIT maturity levels are also similar to those of other prominent frameworks (e. g. NIST SP 800-30 [10], SSE-CMM (ISO/IEC 21827:2008)[24] and CMMI [25]). This also supports the reliability of the user input.

Since the ISO / IEC 27002 standard covers roughly 110 security controls their maturity levels can be assessed by most SMEs within a reasonable time. For very small organisations with less resources it can also be sufficient to concentrate on assessing entire control subcategories (34 items) or categories (14 items). Since the security controls are hierarchically structured the respective categories can easily be derived from the controls. There also exist several examples for similar high-level approaches in practice, for example Australia's framework for SMEs called "Essential Eight Maturity Model" that covers eight high-level controls [1] and the UK's Cyber Essentials scheme that focuses on five controls [2].

COBIT defines six maturity levels (from 0 to 5) that are normalised (by dividing the MLSC by 5) so that the MLSC for a control j ($MLSC_j$) $\in [0, 1]$. The maturity level assesses how mature the organisational processes of the controls are. Each maturity level can be achieved only when the level below has been achieved. The criteria for each maturity level are depicted in Tab. 2.

In larger organisations it can happen that one control has different maturity levels for different zones (e.g. in different departments). Following the weakest-link approach, the minimum maturity level for a control is chosen in this case. However, most SMEs might only rarely be affected by this. But even in this case one can easily deal with this problem by duplicating attack scenarios for another zone so that different maturity levels can be assigned to the same controls.

2.2.2. Reflecting Specific Organisational Characteristics

The optional user input described in this section is used to reflect specific organisational needs and infrastructural characteristics that have an effect on the organisational risk level (see A8 Fig. 6). Users have the option to select the organisation's domain, and to create, to adapt and/or to remove the ACTrees that are used to assess the own organisation.

a) *Selecting a Domain.* A very important way to refine the assessment results is to select the domain in which the user's organisation operates (e. g. the electric domain). Each domain-specific ACTree is associated with one or more domains so that the risk assessment only takes into account the attack scenarios that the user's organisation is exposed to. The domain can also be used to derive the attacker model. For example, for critical infrastructures one should reasonably assume attackers with many resources.

Table 2: COBIT 5 Maturity Levels [22]

Maturity Levels		Description
0	Incomplete	The control is not implemented or fails to achieve its purpose.
1	Performed	The implemented control achieves its process purpose.
2	Managed	The level 1 performed control is now implemented in a managed fashion (planned, monitored and adjusted) and its work products are appropriately established, controlled and maintained.
3	Established	The level 2 managed control is now implemented using a defined process that is capable of achieving its process outcomes.
4	Predictable	The level 3 established control now operates within defined limits to achieve its process outcomes.
5	Optimising	The level 4 predictable control is continuously improved to meet relevant current and projected business goals.

b) *Constructing New ACTrees.* The most powerful option to reflect specific organisational characteristics is to manually construct new ACTrees. For this, the ADTool [6] [3] has been modified with respect to the ACTrees used by LiSRA. After a user has constructed new ACTrees according to his organisation's needs they can be uploaded to the platform in order to individualise the risk assessment for their organisation.

c) *Manually Adapting Existing Trees.* Another option is to manually adapt the parameters or the structure of existing ACTrees. Changing default parameters makes sense if an organisation rates them differently, e. g. the impact of specific attack scenarios. Changing the tree structure makes sense if the organisation's infrastructure or processes significantly differ from the average.

d) *Disabling Trees.* Some of the existing trees might not be relevant for the user's organisation or they might become obsolet due to the construction of new trees or the adaptation of already existing trees. For this reason it is important that users can disable ACTrees so they are not considered for the assessment of their organisation.

e) *Semi-Automatic Adaptation of Trees.* Smaller organisations might struggle to individualise the ACTrees on their own. The most suitable way for those organisations is to make use of a semi-automatic adaptation of the ACTrees based on a short questionnaire. This questionnaire presents a hierarchical overview of asset classes (following ISO/IEC 27005) where the user marks the asset classes that do not exist in the considered risk assessment scope of their organisation (e. g. a smart meter supporting a remote data transmission over WiFi) [9]. LiSRA uses this information to automatically update the ACTrees by eliminating those attacks (trees) or attack steps (subtrees) targeting asset classes that do not exist in the scope. In case of OR operations only the respective subtree is eliminated, whereas for AND operations the parent node is eliminated because logically it cannot be successfully performed, too. The rationale behind the elimination is that attacks or attack steps that require the existence of certain assets cannot be performed without them. So the update is necessary to more precisely reflect the actual attack surface of the user's organisation.

Adaptations made by organisations can also be examined by domain experts in order to enable new or modified trees for other organisations, too. So there is an iterative improvement process in place ensuring a good quality.

2.3. Phase 3: Risk Computation

The detailed risk computation process is visualised in Fig. 6. The total risk is derived from scenario risks that are calculated based on both the probability of adverse impact and its severity.

2.3.1. Resolving Control Dependencies

As described in the previous section, the organisation's security measures are represented by security

[3] The ADTool is an open source software used for graphical modeling of attack-defense trees.

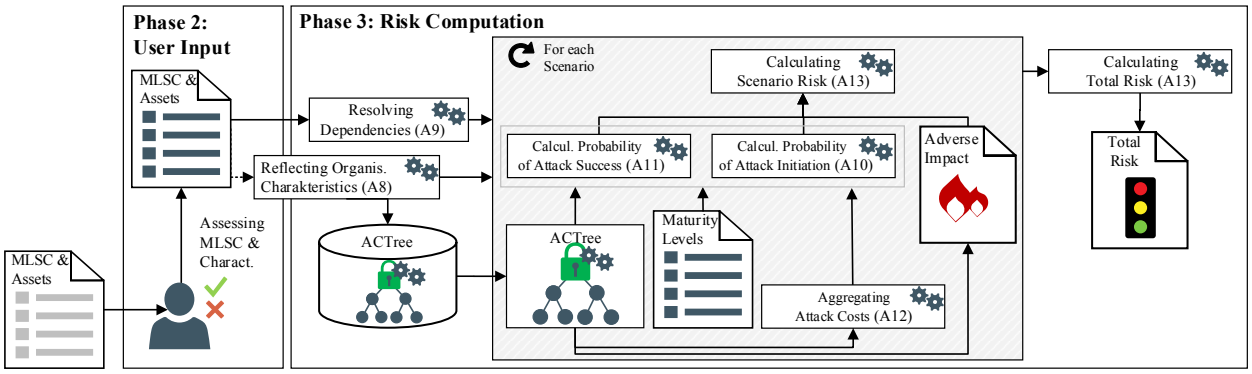


Figure 6: Phase 2 (User Input) and Phase 3 (Risk Computation)

controls following the ISO/IEC 27001. However, many of the controls are dependent on each other so that their effect cannot be assessed independently. Thus their dependencies need to be resolved (see A9 Fig. 6). If a dependent control is not mature enough it might stop other, more mature, controls from being more effective. For example, a very refined and mature physical security perimeter control can be useless if there is no access policy control in place. Sengupta systematically analysed the dependencies between all controls of the ISO/IEC 27002:2013 [20]. The results for the dependencies at the group level are visualised in Fig. 7. In the

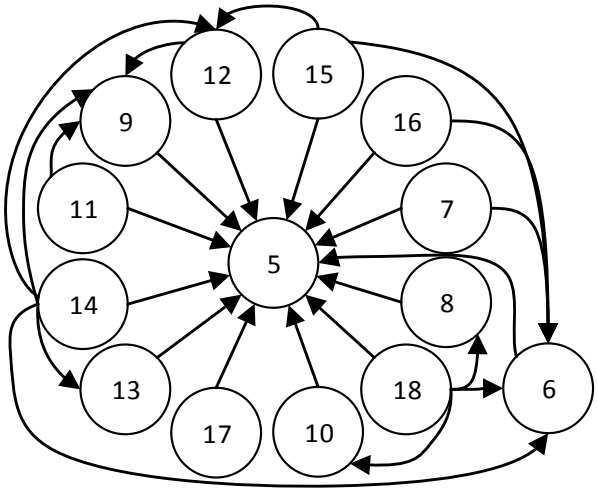


Figure 7: Visualisation of the ISO/IEC 27002 group dependencies, own figure based on [20]

following, we distinguish between strong and weak dependencies. In a strong dependency, one control strictly requires the implementation of another control. For example, the prerequisite to protect an area with a physical security perimeter (control 11.1.1) is the implementation of an access control policy (control 9.1.1). Therefore, it is a strong dependency. On the other hand, dependencies on the organisation's policy for information security (control 5.1.1) for instance, are typically weak dependencies because this policy, which should be defined and approved by the management, influences other controls to a lesser extent.

To resolve these control dependencies the dependency function $d(i)$ is applied (see Eq. (1)). Here, control i depends on the set of the controls $k \in K$. In case of a strong dependency, the MLSC of the dependent control results from the minimum MLSC of both controls. So it follows a weakest link approach. For example, a missing access control policy (MLSC=0) decreases the maturity level of the dependent physical security perimeter control to 0, even if the physical security perimeter control is implemented in a very mature way. In case of weak dependencies a control is supported by its depending controls but they are not necessarily required. So even if the other controls are not in place (MLSC=0) the dependent control can still achieve a good maturity level. This is reflected by $\Delta_{ik} = 3$ in Eq. (1).

$$d(i) = \min_{k \in K}(MLSC_i, MLSC_k + \Delta_{ik}) \qquad (1)$$

with

$$\Delta_{ik} = \begin{cases} 0 & \text{if controls i and k have a strong dependency,} \\ 3 & \text{if controls i and k have a weak dependency.} \end{cases}$$

In order to resolve all dependencies the dependency function is defined recursively and is applied to all dependent controls in the ACTrees, following the dependencies identified by Sengupta [20].

2.3.2. *Assessing the Probability of Attack Initiation*

When the dependencies are resolved, the risk computation starts. The first step is to assess the probability of

attack initiation, $PI \in [0, 1]$ (see A10 Fig. 6). It reflects the selection probability for a specific attack option because (in case of OR operations) an attacker can choose between different attack options. We assume a rational attacker who always chooses the attack option maximising his utility. Some exemplary attacker models are defined in (Eq. 3 to Eq. 6). Here, one-shot attacks are assumed where the attacker only performs the best attack. This is modelled by the following constraint defining that the sum of the weighed decisions for a subtree is 1 (Eq. 2).

$$\sum_{j \in J} PI_{ij} = 1 \tag{2}$$

The first attacker model describes an attacker with very limited resources and a strong cost focus, e.g. script kiddies. In this case, the attacker always chooses the cheapest option.

$$PI_{ij}^{ScriptKiddie} = \begin{cases} 1 & \text{for } j = min_{j \in J} c_{ij}, \\ 0 & \text{else .} \end{cases} \tag{3}$$

The next one defines an attacker who is only interested in the attack option that maximises the probability of success, e.g. nation-state attacker. The attack decisions are not influenced by costs.

$$PI_{ij}^{Nation-StateAttacker} = \begin{cases} 1 & \text{for } j = max_{j \in J} PS_{ij}, \\ 0 & \text{else .} \end{cases} \tag{4}$$

The third attacker model represents an attacker who considers both costs and attack success and concentrates on the cost efficiency of an attack. The model determines the most cost-efficient attack decision. Since it is a good trade-off between probability of success and attack costs it might be an attacker model representing many attackers.

$$PI_{ij}^{EfficiencyMaximiser} = \begin{cases} 1 & \text{for } j = max_{j \in J} \frac{PS_{ij}}{c_{ij}}, \\ 0 & \text{else .} \end{cases} \tag{5}$$

The next model equally covers all attack options ($j \in J$) by assuming a random-shot attacker. It measure the average security.

$$PI_{ij}^{RandomShotAttacker} = \frac{1}{|J|} \tag{6}$$

Apart from those simple attacker models it is also possible to model more sophisticated ones by considering probability distributions (e.g. the standard normal distribution depending on the attacker's success chances) or by more refined utility functions, e. g. following the ideas by Ingoldsby [26].

2.3.3. Assessing the Probability of Attack Success

We define the probability of attack success, $PS \in [0, 1]$, as the probability that an attack (or an attack step), once initiated, succeeds. Thus, it is also determined by the probability of attack initiation. It is calculated using a tree-based algorithm aiming to determine to which degree the implemented security controls protect against attack scenarios or attack steps once an attack is initiated (see A11 Fig. 6).

First, the probability of attack success is calculated for the attacker activities (that are always located in the leaf nodes of the trees). The attacker's probability of success is derived by the strength of the assigned controls. It is determined by the strongest control – so a maximum function is applied (see case 3 in Eq. (7)).

$$PS_i = \begin{cases} \prod_{j \in J} PS_{ij} & \text{for inner nodes with AND,} \\ \sum_{j \in J} (PI_{ij}\, PS_{ij}) & \text{for inner nodes with OR,} \\ CF(1 - \max_{j \in J} CS_{ij}) & \text{for leaf nodes.} \end{cases} \tag{7}$$

The rationale for this is that (following the modelling rules for controls) only in case of substitutive controls more than one control can be assigned to an attacker activity; therefore only the strongest controls takes effect. Then, the probability of attack success is subsequently aggregated up the tree until the final attack goal is reached.

The control strength, $CS_i \in [0, 1]$, measures the ability of the controls j to resist against a specific attacker activity i. In general, the more mature and effective a control is the better it protects against attacks. So a control's strength is defined by the product of a control's maturity and its efficacy (Eq. (8)).

$$CS_i = \min_{j \in J}(e_{ij} \times\ MLSC_j, r) \tag{8}$$

The *min* function is used to model that a control strength of 1 (100 % security) can normally not be achieved. So the residual value is set to $r = 0.99$.

Since the probability of success also depends on the attacker model the control strength is weighted with a capability factor, $CF \in [0, 1]$ (see case 3 in Eq. (7)). It expresses how capable an attacker is in performing a specific attack scenario. It assumes that less capable attackers (like script kiddies) are less successful in performing complex attack scenarios than more capable attackers (like nation-state attackers), whereas they might be equally successful in performing very simple attacks.

Table 3: Attacker Capability

Attacker Model	Attacker Capability		
Nation-State Attacker	unlimited	=	∞
Average Attacker	$3 \times$ high	=	3
Script Kiddy	$1 \times$ low	=	0.33

The capability factor is defined as follows:

$$CF = min(1, \frac{AC^{attacker}}{c_s}) \tag{9}$$

The attacker's capability $AC^{attacker}$ describes how expensive an attack scenario can be for a specific attacker so that he can still effectively cope with it. These costs are not interpreted in a purely monetary sense but also in the sense of required resources which includes factors like attacker skills. Tab. 3 illustrates exemplary input values for different attacker models. Script kiddies have very limited resources and know-how so it is assumed that they might only be capable to effectively perform one attacker activity with low costs, whereas nation-state attackers potentially have unlimited resources. The quantification of cost values is the same as described in Section 2.1.7. (low=0.33; medium=0.67; high=1). NIST SP 800-30 provides additional information for quantifying attacker capabilities that can be used to further refine the input [10]. The attacker's capability is then divided by a reference value measuring the attack costs for an average attacker (like the efficiency maximiser) to execute the entire scenario. These reference value is calculated without considering the capability factor because it is only used to compare the capabilities for different attacker model with each other. These scenario costs can directly be derived from the costs for single attacker activities. More detailed information are provided in the next section (see Eq. (10).

When the weighted control strength is determined for all leaf nodes, they are aggregated up the tree in order to determine the probability of attack success for an entire attack scenario. For this, it is differentiated between inner AND nodes and inner OR nodes[4].

In case of parent nodes with AND operations the attacker does not have any choice, both attack steps have to be performed. To aggregate the probability of success all steps are multiplied by each other (see case 1 in Eq. (7). In case of parent nodes with OR nodes the attacker can choose between different attack options. For each option $j \in J$ the probability of attack success PS is weighted with the corresponding probability of attack initiation PI (see case 2 in Eq. (7). The aggregation process continues until the root node is finally reached.

[4]A parent node with only one child yields the same result a as hypothetical AND- or OR-node with one sub-node.

2.3.4. Aggregating Attack Costs

In most cases attack decisions are influenced by attack cost (e. g. for script kiddies or efficiency maximisers). So there is the need to assess the attack costs for each attack step in the attack tree (see A12 Fig. 6). For this, the initially gathered attack costs for the attacker activities are aggregated up the tree. In case of inner nodes with AND operations the attacker has to perform both attack steps so the attack costs are added up. In case of OR operations the expectation of the attack costs for a successful attack are calculated by weighting the the attack costs with the probability of initiation. So the attack costs are aggregated in the same way as the probability of attack success.

$$c_i = \begin{cases} \sum_{j \in J} c_{ij} & \text{for AND nodes,} \\ \sum_{j \in J} (PI_{ij}\ c_{ij}) & \text{for OR nodes.} \end{cases} \tag{10}$$

2.3.5. Assessing the Risk

The risk for a single scenario, $R_s \in [0, 1]$, is defined as product of the probability of attack success and the magnitude of adverse impact for a scenario s. PS_s and I_s refer to the root node of scenario s.

$$R_s = PS_s\ I_s \tag{11}$$

Finally, the total risk, $R \in [0, 1]$, adds up the weighted risk for each scenario (see A13 Fig. 6).

$$R = \sum_{s \in S} (PI_s\ R_s) \tag{12}$$

2.4. Phase 4: Recommender Application

The next step, when the risk has been computed, is to identify the most beneficial security activities.

One option is to manually inspect the results of the risk analysis. If the total risk indicates the need for action one can go through the list of scenarios to identify the high-risk scenarios. Then, users can manually inspect the respective ACTrees, e. g. to identify the most influential controls for these high-risk scenarios. A manual inspection also enables the risk assessment for very specific attack steps.

However, a faster and more objective approach for comprehensive analyses is to use the recommender application that automatises the inspection process. It can be used to get recommendations for the most effective and the most cost-efficient security activities that

are represented by MLSC increases. To further operationalise the process of improving the maturity levels, there exist mappings between the high-level ISO/IEC 27002 controls and concrete security measures (e. g. the mapping from the German Federal Office for Information Security between ISO/IEC 27002 controls and the security measures listed in the IT baseline protection [27]). Those mappings are especially helpful for MLSC increases from level 0 ("Incomplete") to 1 ("performed"). Fig. 1 visualises how these recommender application interacts with the other components. It receives the MLSC from the user and identifies beneficial security activities by simulating the corresponding risk and costs with the risk computation component.

2.4.1. Most Effective Security Activities

The first recommender application identifies the most effective security activities. It concentrates on a short-term perspective and therefore analyses the effects of incremental MLSC increases by one. The rationale for this is that improvements of organisational routines is a time-consuming process which needs to be conducted stepwise. This is also explicity pointed out in the related Capability Maturity Model (CMM) standard. They argue that skipping maturity levels is counter-productive because each level forms a necessary foundation for the next higher level which also holds for the COBIT maturity levels [28].

To identify the most effective security activities they are ranked according to the effect they have on the risk level. This measures is also known as Birnbaum measure [29]. So LiSRA increments each control's MLSC one after the other and calculates the risk reduction for each MLSC increase. Finally, all security controls with an expected risk reduction above a defined threshold are listed and sorted by the achieved risk reduction.

2.4.2. Most Cost-Efficient Security Activities

The second recommender application is based on a cost-benefit analysis and therefore relates the resulting list of the first recommender application (containing the most effective security activities) with the corresponding security costs. So cost estimations for information security costs are required for this. Here, the term "security costs" is not defined in a purely monetary sense but also in the sense of required resources.

In the following, the security costs are differentiated into the control-specific costs and the step-specific cost factor. Both of them are described below.

- The control-specific cost factor (CC) can be derived from a study by the Software Engineering Institute (SEI) in which they have empirically analysed the time needed to move up to the next MLSC. The data have been gathered with SCMAPI (Standard CMMI Appraisal Method for Process Improvement) that was conducted from 2006 to 2008 with almost 3,500 organisations. The results show that the maximum cost factor[5] for an MLSC increase is 16 [30]. This factor is reflected by the scale for control-specific cost factor depicted in Tab. 4a. For this, a geometric progression with a maximum factor of 16 and a factor to the next level of 2 is used. Brecht et al. have analysed the information security cost ratio for the ISO/IEC 27002 control categories. They can be used as rough default values[6] to estimate the security costs [31].

  However, the study refers to CMMI maturity levels that slightly differ from COBIT maturity levels in the way that COBIT level 1 ("performed") is between the CMMI's level 1 ("initial") and 2 ("managed") – it is assumed that it is exactly between level "initial" and "managed" in terms of time. The other maturity levels are basically the same [22, 25]. This has a negligible effect on the chosen cost factors.

- The security costs do not only depend on the characteristics of a specific security control but also on the concrete MLSC increase which is modelled by the step-specific cost factor (SC).

  Here, it is assumed that the time to move up from CMMI level 0 ("not performed") to 1 ("initial") is similar to the time to move up from CMMI level 1 ("initial") to level 2 ("managed").

Table 4: Costs for an MLSC Increase

(a) Control-Specific Cost Factor (CC)

Costs	Factor
Very High	4
High	2
Medium	1
Low	0.5
Very Low	0.25

(b) Step-Specific Cost Factor (SC)

Step	Factor
0→1	0.4
1→2	0.13
2→3	1
3→4	0.93
4→5	0.6

[5]The cost factor refers to the smallest and the largest observed value that is not an outlier

[6]The control categories 5,6 and 16 are associated with very high costs; category 9 with high costs; 8,11,13,14,17 and 18 with medium costs and 7 with low costs.

Accordingly, it takes 6 months to move from COBIT level 0 to 1, 2 months from level 1 to 2, 15 months from level 2 to 3, 14 months from level 3 to 4, and 9 months from level 4 to 5 [30]. This indicates how much effort MLSC improvements take and how time-consuming they are.

These effort values are now used as a weighting factor w for the security costs SC.

The step-specific cost factor is then normalised so that $SC \in [0, 1]$. Thus, an MLSC increase from 0 to 1 yields $\frac{6}{15} = 0.4$, from 1 to 2 yields $\frac{2}{15} = 0.13$, from 2 to 3 yields $\frac{15}{15} = 1$, from 3 to 4 yields $\frac{14}{15} = 0.93$, and from 4 to 5 yields $\frac{9}{15} = 0.6$. An overview is shown in Tab. 4b.

The next step is to calculate the cost efficiency CE for each MLSC increase of a control i by using Eq. (13).

$$CE_{i,MLSC} = \frac{RR_i}{CC_i \times SC_{MLSC}} \tag{13}$$

It divides the received risk reduction RR by the step-specific security costs SC that arise from an MLSC increase for control i. Then, all controls with an cost efficiency above a defined threshold are sorted and displayed. An example is shown in Sect. 4.4.

2.4.3. Providing Transparent Recommendations

Transparent recommendations are of crucial importance for the acceptance of recommender systems such as LiSRA. It describes to which extent users understand why a particular item is recommended to them [32]. Therefore, besides the recommendations themselves, also the rationale behind the recommendations is presented to the user by a graphical explanation interface.

The mitigating effects of the recommendations are presented to the user in different ways. He can choose between the scenario-centric and the recommendation-centric perspective. The scenario-centric perspective contrasts the effects of all recommendations for a specific scenario, whereas the recommendation-centric perspective illustrates the mitigating effects of a specific recommendation for each scenario. All nodes (attack steps) in the ACTrees are coloured according to the reduced probability of attack success caused by the recommended control increase. The colour coding ranges from red (no effect) to green (very high effect). The user can navigate through the trees to review the mitigating effects on each attack or attack step for each recommendation. The graphical explanation interface presents the mitigating effects of a control in the context of concrete attack steps. This serves to implement the given recommendations in a more effective way. By looking into the ACTrees decision-maker might learn that the security control "information security awareness, education & training" should be implemented with a stronger focus on phishing attacks than on other attacker activities.

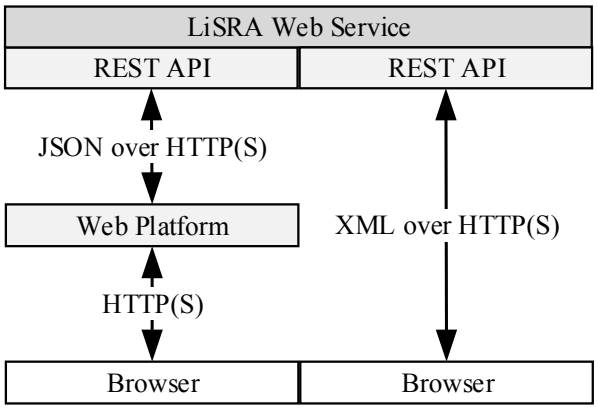


Figure 8: Architecture

2.5. Implementation

The LiSRA framework has been implemented as a RESTful web service in Java so it can easily be imported and used by other projects as well (LiSRA-as-a-Service). The high-level architecture is illustrated in Fig. 8.

The web service can for example be called over HTTP(S) with a simple browser GUI where a user uploads an XML file containing his MLSC. As return he gets back another XML document presenting the total risk as well as the specific risks for each attack scenario.

Additionally, the LiSRA framework has been integrated into the SIDATE security management web platform which has been developed in Liferay 7.0 [33]. The user enters the organisation's maturity levels in the data input section (see Fig. 9), whereupon all the risks are graphically represented in the risk representation section (see Fig. 10). For this, the web portal transmits the user's MLSC to the web service (in JSON) that returns back all the risk levels. For the sake of transparency, the corresponding ACTrees are visualised, too. The purpose of the integration was to further ease the process of going through a longer questionnaire. It aims to ease the burden of going through a longer questionnaire by enabling and motivating the user to complete or to update the MLSC along the way when interacting with other parts of the platform [8].

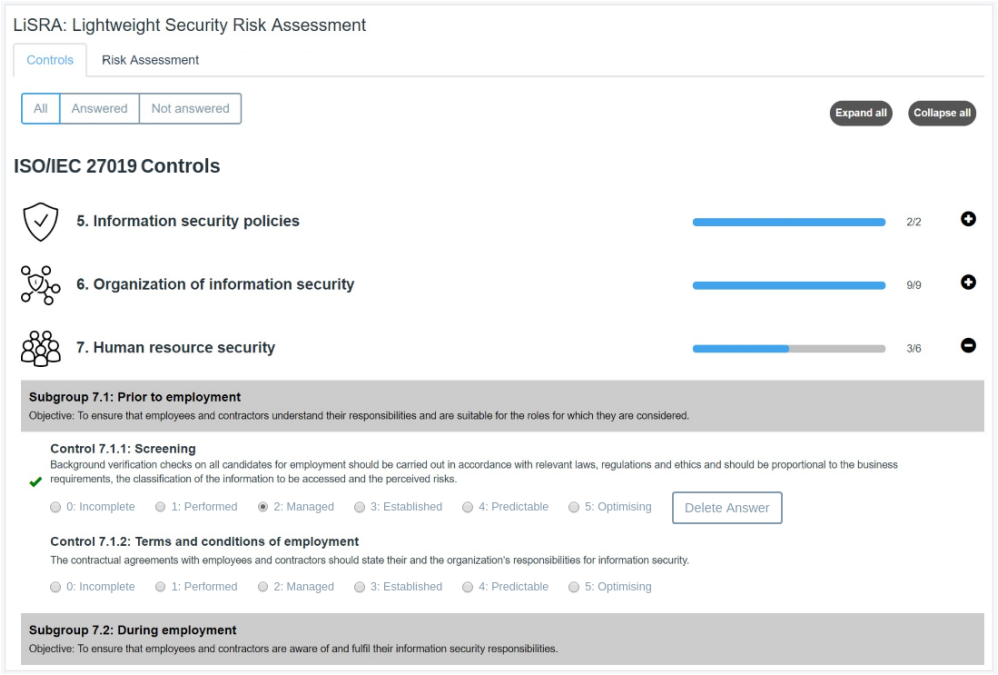


Figure 9: Data Input Section

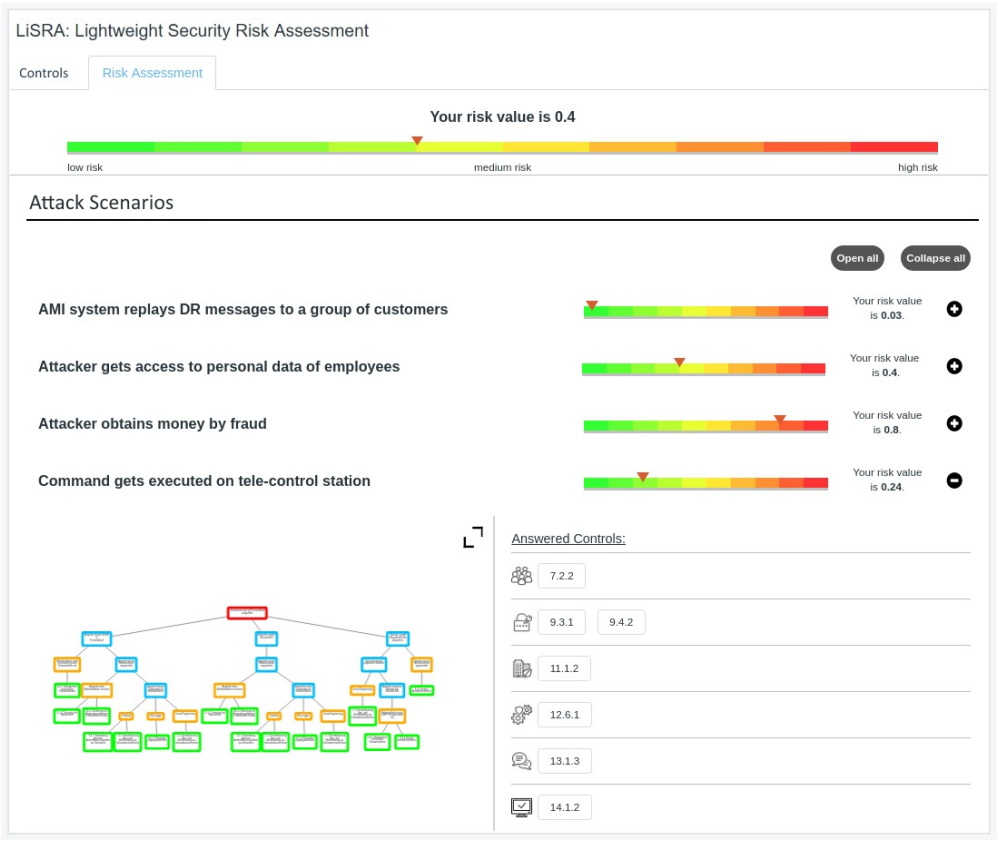


Figure 10: Risk Representation Section

3. Example

In this section we demonstrate for the exemplary domain of the electric sector that LiSRA can be used with little extra effort.

3.1. Phase 1: Expert Input

In phase 1 the experts construct and parameterise the ACTrees.

3.1.1. Identifying Attack Scenarios

The first step is to identify relevant attack scenarios. For the elctric sector there exist a well elaborated collection of attack-defense trees including the corresponding impact categories that can be used as initial input for the framework. They are provided by the National Electric Sector Cybersecurity Organization Resource (NESCOR) [15]. Although their trees are represented in a different way (so they need to be transformed), it makes much sense to use them as a starting point. Generally, it is recommended to built on already established material in order to save time and costs and to improve quality.

For the exemplary application of the model we use the simple attack scenario illustrated in Fig. 11 where the attacker tries to steal a server.

3.1.2. Assessing Adverse Impact

The impact assessment scale is illustrated in Tab. 1. The impact assessment always depends on the specific context of the scenarios (e.g. the assets at stake). For the given scenario we assume a severe adverse impact with $I_s = 1$.

To further refine the results it can make sense to use a domain-specific method. For the electric sector there is an impact scoring model proposed by the National Electric Sector Cybersecurity Organization [5] where the experts score the impact of scenarios based on 15 criteria[7]. For each criterion they can select one out of four choices. Depending on their answer, the criterion is scored with 0, 1, 3 or 9. The overall sum (between 0 and 135) reflects the scenario's impact. For the reason of s implicity, the scoring model is not used in the example.

3.1.3. Constructing Attack Trees

The ACTree used in the exemplary attack scenario (see Fig. 11) is a simplified tree only used for demonstration purposes and to explain how LiSRA works. As

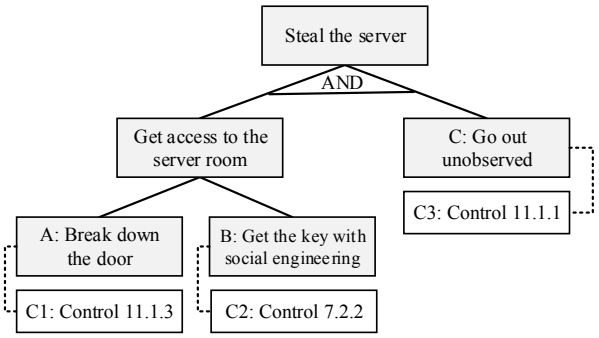


Figure 11: Exemplary Attack-Control Tree

defined in Section 3, the root node of the tree represents the attack goal of the scenario and all attacker activities are located in the leaf nodes. The presented scenario is inspired by Bistarelli et al. [16]. The attack goal is to steal a server. To achieve this, the attacker must have access to the server room and must go out unobserved (attacker activity C). There are two option to get access to the server. He can either break down the door (attacker activity A) or he can get the key using social engineering (attacker activity B).

3.1.4. Assigning Assets

In the given example the rood node is obviously associated with the general asset class "server". A high-level perspective is sufficient in this case because the attack scenario is very high-level, too.

3.1.5. Assigning Controls

When the attack trees have been constructed the corresponding security controls are assigned. As recommended above, the control list from ISO/IEC 27002 can be used respectively the more specified ISO/IEC 27019 which addresses the special needs for the electric sector.

In the given simplified example three controls are assigned. A protection against attacker activity A ("break down the door") is control C1 (11.1.3) which addresses "securing offices, rooms and facilities". The second attacker activity "get the key with social engineering" can be mitigated by control C2 (7.2.2) which is about "information security awareness, education and training". Control C3 (11.1.1) is about "physical security perimeter" which comprises for instance video surveillance. So it protects again the attacker activity of "going out unobserved"

3.1.6. Assessing Control Efficacy

Next, the ACTrees are parameterised. The control efficacy depends on the context so it is individually as-

[7]Examplary criteria are "negative impact on customer service", "negative impact on billing functions" or "restoration costs".

sessed for each associated attacker activity. For example, the control "securing offices, rooms and facilities" is assumed to be effective against breaking down a door so its efficacy is assessed as "high" ($e_{C1} = high \Leftrightarrow e_{C1} = 1$), whereas the general control "awareness, education and training" is assumed to be less effective against specific social engineering attacks ($e_{C2} = medium \Leftrightarrow e_{C2} = 0.67$).

3.1.7. Assessing Attack Costs

The attack costs are gathered for each attacker activity using the 3-point scale defined in Section 3. In the present example it is assumed that the costs to get the key with social engineering are significantly higher ($c_B = high \Leftrightarrow c_B = 1$) than to break down a door ($c_A = medium \Leftrightarrow c_A = 0.67$) which is again assumed to be more expensive than going out unobserved ($c_C = low \Leftrightarrow c_C = 0.3$).

3.2. Phase 2: User Input

3.2.1. Assessing Maturity Levels

After the initialisation phase the user enters his organisation's MLSC. For control C1 it is assumed that there are established processes that are performed in the entire organisation to make sure that offices, rooms and facilities are protected. Therefore, $MLSC_{C1} = 3$. Information security awareness trainings (control C2) are irregularly performed but not in a managed way so $MLSC_{C2} = 1$. The processes addressing physical security perimeters (control C3) are systematically monitored and measured at an organisational level so $MLSC_{C3} = 4$. Finally, all maturity levels are normalised between 0 and 1 (by division by 5) so that $MLSC \in [0, 1]$.

3.2.2. Reflecting Specific Organisational Characteristics

Since it is assumed that the given organisation has servers in place (which is the only asset class associated with $Scenario_1$) the tree is fully considered in the risk assessment. Otherwise, it the entire scenario or attack steps would be excluded from the analysis.

3.3. Phase 3: Risk Computation

The risk computation process, visualised in Fig. 6, starts with resolving the control dependencies. Afterwards, the risk is computed based on attack scenarios.

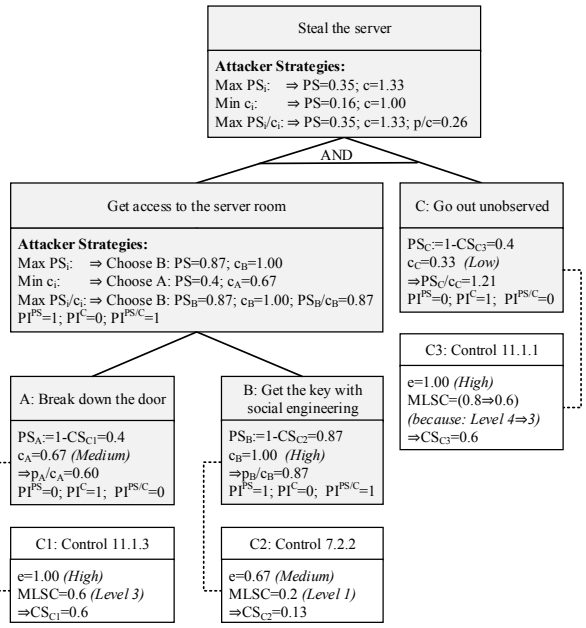


Figure 12: Exemplary Attack-Control Tree with Parameters

3.3.1. Resolving Control Dependencies

According to Sengupta's list of control dependencies, there is a strong dependency in the present example. Control C1 (11.1.3) depends on control C3 (11.1.1) [20]. Inserting their MLSC into the dependency function (Eq. 1) yields $min(3, 4) = 3$ wherefore the *effective* MLSC for control C3 is decreased by one ($MLSC_{C3} = 4 \rightarrow MLSC_{C3} = 3$). For reasons of simplicity, only the controls depicted in the ACTree are considered. Otherwise the control 11.1.2 would have to be analysed as well as the dependent controls in group 5 and 9 that are indicated in Fig. 7

3.3.2. Assessing the Probability of Attack Initiation

The probability of attack initiation reflects the selection probability for a specific attack options. In this example, we consider an attacker who always chooses the attack options with the maximum efficiency that is represented by $PI^{EfficiencyMaximiser}$.

3.3.3. Assessing the Probability of Attack Success

The probability of attack success for an attack scenario is derived from the attacker's probability of attack success for each attacker activity which is calculated by the strength of the assigned security controls. The results are also graphically illustrated in the ACTree depicted in Fig. 12. The first attacker activity A ("break down the door") is associated with one control: control C1 ("securing offices, rooms and facilities") that

Table 5: Input Parameters for Attack-Control Trees

Attacker Activity			ISO/IEC 27002 Security Controls				
ID	Description	Costs	ID	Description	Efficacy	Maturity	Strength
A	Break down the door	Medium	11.1.3	Securing offices, rooms and facilities	High	3	0.4
B	Get the key with social engineering	High	7.2.2	Information security awareness, education and training	Medium	1	0.87
C	Go out unobserved	Low	11.1.1	Physical security perimeter	Medium	$4 \rightarrow 3$	$0.2 \rightarrow 0.4$

includes for instance burglar resistant doors. So the control's efficacy for this attack is assumed to be high ($e = high \Leftrightarrow e = 1$) and the organisation's MLSC in the scenarios is 3 ($MLSC = 3 \Leftrightarrow MLSC = 3/5 = 0.6$). Then, the efficacy and the MLSC are used to calculate the controls strength ($CS_{C1} = min(0.6 \times 1), 0.99) = 0.6$.

To determine the probability of attack success the capability factor has to be assessed first. Assuming an average attacker with an attacker capability of $AC^{EfficiencyMaximiser} = 3$ (see Tab. 3) and average attack costs (for the efficiency maximiser) to perform the scenario of $c_s = 1.33$ (see Fig. 12) the capability factor yields $CF = min(1, 3/1.33) = 1$. Therefore, the probability of attack success is $PS_A := 1(1 - 0.6) = 0.4$. The same is done for attacker activity B ($PS_B := 1(1 - CS_{C2}) = 1(1 - min(0.2 \times 0.67), 0.99)) = 1(1 - 0.13 = 0.87$), and for attacker activity C (whose maturity level was decreased due to the control dependencies) ($PS_C = 1(1 - CS_{C3}) = 1(1 - min(0.6 \times 1, 0.99)) = 1(1 - 0.6) = 0.4$)

When the probability of attack success has been calculated for each attacker activity, the values for the parent nodes are calculated. The first parent node ("Get access to the server room") uses an OR operation so an attacker can decide between the attack steps A and B. The decisions is made based on the considered attacker model. In case of the efficiency maximiser (Eq. 5) activity B is chosen (because $0.87 > 0.60$). So in this case the parent node ("Get access to the server room") continues with the values for attack step B. The next parent node uses an AND operator. The attacker has to perform both attack steps so the respective probabilities are multiplied with each other. For the efficiency maximiser the probability of attack success for the scenario ("steal the server") is $PS_s = 0.87 \times 0.4 = 0.35$ and and the corresponding attack costs are $c = 1.33$.

3.3.4. Aggregating Attack Costs

The costs that are aggregated using Eq. (10) are presented in Fig. 12, following the same aggregation logic as in the previous section.

Table 6: Effects of the MLSC Increase for Control 7.2.2 (C2) from $MLSC = 1$ to $MLSC = 2$

	$Scenario_1$	$Scenario_2$	...
Prob. of Success	0.35	0.08	...
Attacker Costs	1.33	0.5	...
Attack Efficiency	0.26	0.16	...
Prob. of Initiation	1	0	...
Impact	1	1	...
Scenario Risk	0.35	0	...
Total Risk		0.35	

(a) Before MLSC Increase

	$Scenario_1$	$Scenario_2$	...
Prob. of Success	0.29	0.08	...
Attacker Costs	1.33	0.5	...
Attack Efficiency	0.22	0.16	...
Prob. of Initiation	1	0	...
Impact	1	1	...
Scenario Risk	0.29	0	...
Total Risk		0.29	

(b) After MLSC Increase

3.3.5. Assessing the Risk

Since LiSRA is a scenario-based approach the risk is first calculated for each scenario, whereupon the risk are aggregated. For a better illustration the hypothetical attack scenario 2 is added. The risk scores for $scenario_1$ and $scenario_2$ are depicted in Tab. 6. Inserting them in Eq. 11 yields $Risk_1 = 0.35 \times 1 = 0.35$ and $Risk_2 = 0.08 \times 0 = 0$. The procedure is repeated for each scenario. Finally, the organisation's total risk is calculated by adding up the weighted scenario risks according to the considered attacker model. The efficiency maximiser would choose $Scenario_1$ which has the best cost-success ratio, so the total risk is 0.35.

3.4. Phase 4: Recommender Application

The recommender application recommends the most effective and the most cost-efficient security activities in a short-term perspective.

Table 7: Simulation of Incremental MLSC Increases

	C1↑	C2↑	C3↑	...
Before MLSC Increase	3	1	4	...
After MLSC Increase	4	2	5	...
Total Risk Reduction	**0.18**	0.06	0	...
Security Costs	0.93	0.07	0.6	...
Cost Efficiency	**0.19**	0.12	0	...

3.4.1. Most Effective Security Activities

To determine the most effective security activities each control's MLSC is one after another incremented by one in order to simulate the caused risk reduction. The result is shown in Tab. 7. Improving control C3's MLSC does not cause any risk reduction because the dependency with control C1 stops C3 from being more effective. On the other hand, an MLSC increase of C1 also has a positive effect on the MLSC of C2 because C2 is not limited anymore from C1. So an increase of C1 causes the highest risk reduction.

Tab. 6 shows the effects of an MLSC increase of C2 in detail. It is assumed that C2 is not covered by the second scenario. The MLSC increase significantly reduces the probability of attack success ($PS_{S1} = 0.35 \rightarrow PS_{S1} = 0.29$) and the attack efficiency for the first scenario. The same holds for scenario risk ($R_{S1} = 0.35 \rightarrow R_{S1} = 0.29$) and for the resulting total risk ($R = 0.35 \rightarrow R = 0.29$).

3.4.2. Most Cost-Efficient Security Activities

The recommender application also identifies the most cost-efficient security activities. It takes the list with the achieved risk reduction (from most effective security activities) as basis and relates it with the arising control-specific costs CC_i and the step-specific cost factor SC_{MLSC} to reflect the MLSC increase. The simulated efficiency per MLSC increase is depicted in Tab. 7.

Low security costs are assumed for control C1 ($CC_{C1} = medium \Leftrightarrow CC_{C1} = 1$) with a step-specific cost factor for the MLSC increases from 3 to 4 of $SC_3 = 0.93$ which makes total costs of 0.93. Low security costs are assumed for C2 ($CC_{C2} = low \Leftrightarrow CC_{C2} = 0.5$) with a step-specific cost factor for MLSC increases from 1 to 2 of $SC_1 = 0.13$ which results in total costs of around 0.07. The same is done for C3 which causes costs of 0.6.

After dividing the risk reduction by the total security costs, it can be seen that an increase of C1 is the most cost-efficient security activity (see Tab. 7).

3.4.3. Providing Transparent Recommendations

In order to implement the recommended MLSC increases more effectively users can navigate through the tree and compare the mitigating effects (measured in risk reduction) for the recommended security activities. The visualisation in Fig. 13 illustrates the recommendations in a scenario-centric perspective that indicates the effects of the MLSC increases for $Scenario_1$. The graphical representation also shows the indirect effect of control C1 to the attacker activity C that is caused by a dependency. Besides that, it indicates that decision-makers should implement C1 with a special emphasis on the protection of doors (see Fig. 13a). It also highlights the importance for control C2, that normally covers very general trainings and awareness activities, to explicitly address social engineering issues (see Fig. 13c).

4. Evaluation

The evaluation of security management frameworks is a challenging task, especially because there does not exist any gold standard that could be used to conclude validity. Verendel surveyed 90 papers on quantified security where he systematically analysed which methods have been used for validation. He points out that in most cases an explicit empirical validation is missing (except for vulnerability discovery models) [34]. This is because "measuring security is hard" as Pfleeger et al. state [35]. This holds in particular for risk assessment at an organisational level because it typically deals with very complex targets of evaluation and a large scope.

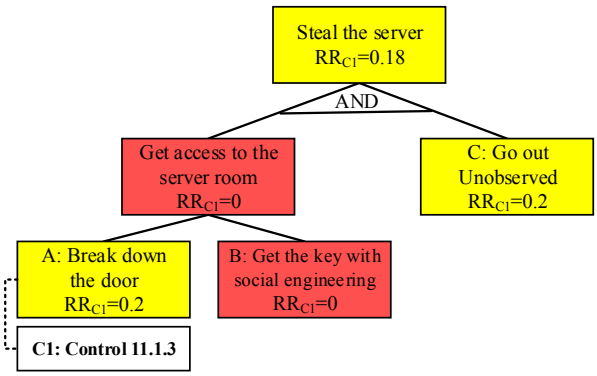
However, various important aspects of the framework have been evaluated like its applicability which has been analysed by performance tests, by analyses of robustness, and in initial qualitative evaluations. Moreover, we have examined the perceived usefulness as well as the concerns of sharing sensitive data .

4.1. Robustness

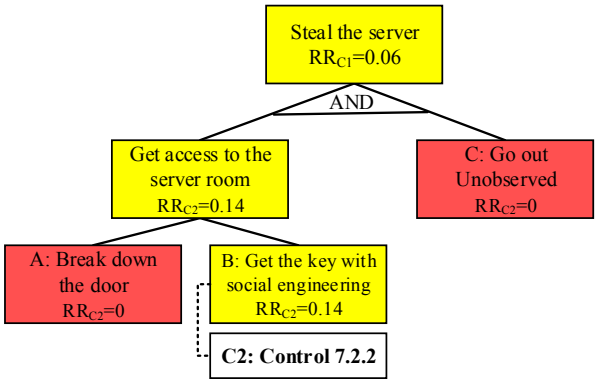
The quality of the risk assessment strongly depends on the robustness of the ACTrees. It is essential that the computed risk is robust against logical transformations (e.g. with respect to the associative or the distributive law) of the tree structures. The mathematical proofs that the computation of the probability of attack success is robust against logical transformations are presented in Appendix A.

Another aspect that is related herewith is the robustness with regard to the abstraction level of attack scenarios. It is possible to merge independent attack trees

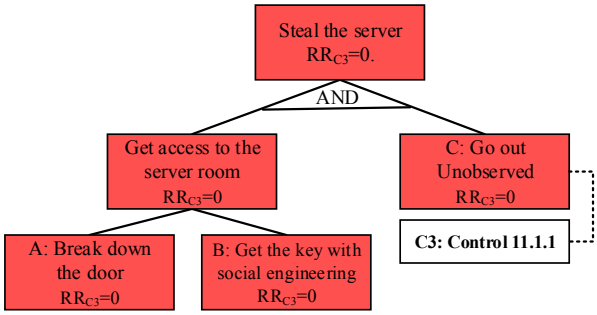
Figure 13: Visualisation of the Risk Reduction (RR) for the Recommended Controls



(a) Effects of the MLSC Increase for Control 11.1.3 (C1)



(b) Effects of the MLSC Increase for Control 7.2.2 (C2)



(c) Effects of the MLSC Increase for Control 11.1.1 (C3)

with OR operations in order to construct a larger tree at a higher abstraction level. Both equivalents must produce the same risk. This is the case because the PI (probability of initiation) function is applied to both. It does not matter if a tree is represented as a single tree or as a subtree, as long as the impact I is assessed correctly.

4.2. Performance

The practical applicability of the framework is an essential factor to be used in practice. An implementation-independent measure for this is the time complexity of an algorithm. Let i be the maximum number of attacker activities in an ACTree, then the tree consists of i leaf nodes and of maximum i inner nodes. In total, it makes a maximum of $2i$ nodes for each tree. For each of the $2i$ nodes some parameters (PI, PS, c, AC and CS) are calculated. Even though PS and c are defined recursively they only need to be calculated once for each node. The parameters are calculated with cheap operations like multiplications, additions or comparisons (for a constant set of controls) that require constant time. The nodes' parameters directly result from their child nodes. Since a node cannot have more than i child nodes the worst-case time complexity to assess a scenario's risk is $\in O(i^2)$. For all scenarios the worst-case time complexity is $\in O(s\ i^2)$. Since it is not possible that a tree has a height of i and each node has i child nodes at the same time the presented complexity analysis is very conservative so that the complexity might be even better with less strict constraints. However, the number of scenarios and the number of attacker activities in a tree are typically not very high so their risk can be assessed in a reasonably time, even for the worst-case.

The performance of the framework has also been tested with several performance tests. We have analysed the performance of the web service (for both local and remote calls) and for the web platform (for local calls). The performance tests were conducted for different numbers of realistic ACTrees (20, 50, 100, 200, 500). The ACTrees had an average number of 36.06 nodes and an average depth of 4.94 nodes. The performance tests for the web service have been automatically executed by a script logging the mean value, the median and the standard deviation (SD) for 100 service calls. The web platform has been manually tested with 5 calls using the Chromium browser version 71. The performance tests for the local web service and the web platform were conducted on a laptop with a 1.8 GHz processor and 8 GB RAM. For the remote web service tests the web service has been installed on a Tomcat server being located on a virtual private server in the same country.

It has a dual-core processor with 2 GHz and 4 GB RAM (without hyperthreading).

Tab. 8 presents the measured times which demonstrates the practical applicability of the framework from a performance perspective. The remote web service can easily handle even a vast amount of 500 attack scenarios. Applying the risk computation of phase 3 takes around 0.225 seconds (median value) for 500 scenarios. For the application scenario of recommending security activities more iterations are needed. Considering the 110 security controls of ISO/IEC 27002 and 500 ACTrees, it would approximately take 24.75 sec. However, these computations could be computed in parallel, i.e. by running several instances of the web service in parallel.

Expectedly, the page load time in a browser is significantly higher than when accessing the web service directly. The most time-consuming factors are the rendering and the scripting.

However, in practice one would expect a significantly lower number of attack scenarios. Furthermore, the source code was developed in a prototypical way without focussing on time efficiency so there is much potential to reduce the performance time.

The performance tests have also shown that the tree structure has no influence on the performance of the algorithm. This has been tested by simulations with a number of ACTrees and their transformed equivalents (n=100). The performance was directly measured in the web service. The median values were 103.4 ms and 103.9 ms.

4.3. Perceived Usefulness

The perceived usefulness of the SIDATE security management platform has been evaluated in a workshop [8]. One central part of the platform is the LiSRA framework which has been evaluated in a focus group of ten experts from eight small or medium-sized energy providers. Most of them had a profound security background and have gained experiences with ISO/IEC 27001 certification as auditor or customer.

First, a live demo of the web platform has been presented. Due to time limitations it has been focused on the conceptual ideas and is has not been gone in-depth. The attendees could interrupt at all point in time to ask any kind of questions. Afterwards, a moderated discussion was initiated where the experts were asked for general feedback and for suggestions for improvement based on their own experiences.

A central aspect of the discussion was the relevance for the ISO/IEC 27001 certification process. The experts agreed that the framework would be helpful for an internal pre-audit that takes place before the official ISO/IEC 27001 audit starts. They also emphasised that it would make a lot of sense to go through the ISO/IEC 27002 respectively 27019 controls because this would reflect what the auditor checks in the end.

In terms of suggestions for improvement they mentioned the idea to add a recommender feature that was not implemented at this time.

4.4. Concerns of Sharing Sensitive Information

For another study, the concerns of sharing sensitive data in the security management platform have been analysed, including the implemented LiSRA framework [33]. Two workshops have been conducted with experts from small and medium-sized energy providers (seven experts from six energy providers in the first workshop; six experts from five energy providers in the second workshop). The only, but sensitive, user input of the LiSRA framework are the maturity levels of the security controls. The experts did not have any concerns with sharing their maturity levels with the platform provider as long as they get a benefit out of it. Similar insights can be derived from the acceptance of the TISAX (Trusted Information Security Assessment Exchange) platform in the German automotive industry. TISAX is a sector-specific exchange platform for the German automotive industry where the results of a standardised security self-assessment (VDA-ISA) can be shared with other companies[36]. However, the data processing could also be done locally so that there would not be the need to transfer the maturity levels to an external server.

4.5. Limitations

The framework is not without limitations. First, the modelled attacker strategies only reflect one-shot attacks, that is an scenario where the attacker attempts to attack an organisation only once. He performs the best attack strategy (maximising his utility) and he does not try the second or the third-best option if he was not successful. Especially attackers with unlimited resources might follow a multiple-shot strategy.

Another limitation is that the framework is designed in particular for SMEs where a control is typically assigned to one maturity level only. In larger organisations it can happen that one control has different maturity levels in different zones. However, LiSRA can deal with this problem by duplicating attack scenarios for another zone where different maturity levels can be assigned the same controls.

Table 8: Performance tests (measured in ms)

ACTrees	Web Service (local call)			Web Service (remote call)			Page Load Time (browser)		
Quantity	Mean	Median	SD	Mean	Median	SD	Mean	Median	SD
20	95.4	93.93	5.25	137.2	134.9	6.61	5,983	5,865	294.14
50	97.81	96.27	4.30	149	144.6	24.28	7,010	7,114	450.84
100	101.28	99.83	3.63	155.5	151.4	19.39	9,505	9,516	517.9
200	110.3	107.9	6.56	192.8	190.1	12.71	12,739	12,318	819.56
500	132.8	129	7.72	227.4	224.8	18.96	26,243	25,994	701.29

5. Related Work

LiSRA is an information security risk assessment framework that also gives recommendations on future security activities. Related work for both fields of research are presented in the following.

5.1. Information Security Risk Assessment

Many literature reviews on risk assessment methodologies have been conducted in the last years [37, 38, 4, 39, 40, 41, 42]. They demonstrate that there is a lack of lightweight and reasonable frameworks that can be applied by SMEs. They provide evidence that most approaches require security-related information that are not available and that are very challenging to gather, especially for SMEs. It also becomes clear that other explicit SME approaches have far less informative value than LiSRA. An example is the model proposed by Bojanc et al. that asks for concrete values for the threat probabilities, the asset vulnerabilities and for the quantification of different loss factors [43]. It is similar for the FAIR framework that aggregates input parameters following a risk taxonomy in order to derive an asset's risk [44]. This requires the user to first define individual aggregation rules for each children-to-parent relation in the taxonomy because they strongly depend on organisational characteristics. Besides that, it is also not defined how to apply the model in order to assess the entire organisational risk. Another example is the approach by Pieters et al. that assesses the adversarial risk for an attack scenario on the basis of complex functions that are used to derive the attack success. It is very difficult to parameterise the functions, particularly for SMEs. Their approach also does not consider which security controls are in place, let alone how mature they are. On the other hand, it is one of the few models that explicitly takes into account the attacker knowledge level [45]. Karabacak et al. propose ISRAM (information security risk analysis method) – a risk assessment framework that aims to improve the quality of inaccurate input data using a survey-based method where the probability of occurrence and the consequence of occurrence are assessed for each attack scenario in two independent surveys. Although this method can improve the quality of non-available input data it still requires a sufficient number of experts with good "knowledge and awareness on the information security problem, its effects and its probable causes" [46]. They are necessary to identify and to adequately evaluate all relevant attack scenarios. So for most SMEs who typically lack in security experts it is not a suitable solution, also because of the organisational overhead that might exceed their security capacities [39].

Apart from that, many frameworks only provide extensive process descriptions and guidelines. This holds for example for OCTAVE-S [3] but also for numerous other approaches [4]. This can be challenging in particular for SMEs that usually have less capacities to become acquainted with comprehensive frameworks.

But there do exist other approaches that are designed for SMEs aiming to explicitly address their special needs. Two of the most prominent examples are Australia's framework for SMEs called "Essential Eight Maturity Model" and the UK's Cyber Essentials scheme [1, 2]. However, they only cover eight respectively five high-level security controls which makes clear that their informative value is far less than LiSRA's. The same also applies to other approaches like the analytic hierarchy process (AHP) based approach by Schmid and Pape that provide less informative value [47].

5.2. Economics of Security Activities

Since Ross Anderson argued for the importance of the economic perspective in information security in 2001 [48] and the Gordon–Loeb model raised interest in 2002 [49], extensive work has been done in the area of economic evaluation of information security activities. A literature review from 2017 on the economics of security investments systematically documents the challenges for many existing evaluation approaches [50]. It shows that many evaluation approaches for security ac-

tivities use information risk assessment approaches as a basis. So the limitations of general risk assessment approaches also apply for many evaluation approaches. So most approaches require non-available data that is hard to estimate and require in-depth knowledge in security, and can therefore not be applied by SMEs. A similar picture is also drawn in both survey paper by Neubauer [51] and by Ruan [52]. This documents that designing a lightweight framework with low requirements on the expected user input is a hard problem and still a challenging task. Good examples for this are the approach by Benaroch that expects probability distributions of investment outcomes as input data [7], and the approach by Manusco et al. where one first has to model the conditional probability tables for each scenario as basis for Bayesian networks [53].

There are also many approaches in literature that are defined very high-level. This applies for several RoSI (return on security investment) approaches that ask for high-level parameters like the annualised rate of occurrence that is challenging to estimate. This applies to the approach by Bistarelli et al. that evaluates and compares different security measures based on their return on security investment (RoSI) and their return on attack (ROA) [16]. Another common issue is that the status of high-level security controls descibing complex processes (e. g. ISO/IEC 27002 controls) is represented using a binary scale asking only for its presence [54]. This does not reflect the large spectrum of the possible implementation level at all.

Another crucial weakness of many existing approaches is that they evaluate security measures in isolation of measures already in place and that the effects of overlapping measures are often ignored by assuming they are independent from each other. They also do not reflect that different measures can be of complementary, substitutive or dependent nature which leads to an over-investment in security. This shortcoming becomes evident from broad literature reviews on security investment models [50, 51, 55]. Benaroch points out this weakness very clearly [7] referring to a number of existing work. Sawik, for example, writes that "The blocking effectiveness of each countermeasure is assumed to be independent whether or not it is used alone or together with other countermeasures" [56]. Tsalis et al. explain that "an asset is protected by multiple controls, but these may mitigate the same threats or incidents. [...] For simplicity reasons, we will assume that the controls mitigate threat independently" [57].

The same holds for the approach by Bistarelli at al. that also neglects any direct effect between different security measures, and thus implicitly assumes substitutive controls [16]. Although most attack tree approaches strictly assume complementary effects like Mancuso et al. [53, 6], others additionaly allow to model weak dependencies between measures [54]. Apart from that, there also exist more elaborated approaches aiming to precisely model the interacting effects between different security activities. However, these models typically require non-available information [7].

It is also important to consider the dependencies between security controls when identifying the most beneficial security activities. Gadyatskaya, for instance, refers to the ISO/IEC 27002 controls but neglects their dependencies when identifying the most optimal security measures [54]. This is problematic as shown by Sengupta [20].

Furthermore, most approaches do not differentiate between different attacker models. They assume an average attacker type (with average resources and average strategies) and neglect that the probability of attack success, and thus the risk, can strongly vary between different attacker types. For critical infrastructures, for instance, one should reasonably assume more powerful attackers with more resources than for other organisations. A universally applicable framework should meet this requirement. The authors are not aware of any other economic evaluation approach for security activities that enables the user to choose between different attacker models [50, 51, 55].

A major advantage of attack tree-based approaches over other methods is that they can provide detailed information why a security activity is as good or bad as it is claimed to be, and how they can be implement in the most effective manner. They are predestined for this because the intermediate results, i.e., the (reduced) probability of attack success, are calculated for each node in the tree. This makes it possible to navigate through the tree and to compare the mitigating effects of the recommended security activities in the context of concrete attack steps. However, a basic problem with attack tree-based approaches is that the quality of the assessment results strongly depend on the assumption that the underlying algorithm is robust against logical tree transformations. However, the authors are not aware of any other tree-based evaluation approach that provide evidence for this key requirement.

Although LiSRA is a universal framework that can be individualised for different domains (as shown for the electric sector) there also exist more specialised approaches addressing technical domain-specific challenges, i. e., to take into account individual client-specific security requirements in cloud computing [58].

6. Conclusion and Outlook

Assessing information security risks is one of the core duties for decision-makers in information security. In order to allocate their finite resources efficiently they need to understand the risks their organisation is exposed to. However, there is a lack of lightweight and reasonable frameworks that can be applied by SMEs. Most approaches either require too many information or their informative value is far less than LiSRA's.

Therefore, we propose LiSRA, a lightweight framework for decision support in information security. Due to the two-sided input users can focus on specifying their security practices by entering information that many organisations have already collected. These information are linked to attack paths and to the corresponding adverse impacts in order to finally assess the total risk. Apart from that, LiSRA can also be used to identify the most effective and the most cost-efficient future security activities. It provides detailed insights on their mitigating effects that also supports decision-makers in implementing the given recommendations in an effective manner. In contrast to most existing approaches, it also explicitly considers the security activities that are already implemented, and it takes into account that multiple overlapping security activities can affect each other in a complementary, substitutive or dependent way. The framework has been implemented in a prototype and its applicability has been evaluated in quantitative and qualitative analyses.

The next step is to extend the recommender application so that it identifies the optimal security activities given a limited budget. Furthermore, concrete distribution function need to be specified and empirically tested for the attacker models.

Based on the attack-control trees already constructed it is planned to conduct a case study with real-world data to evaluate how well LiSRA performs in practice and to get firsthand feedback from the organisation's experts.

Acknowledgments

This research was funded by the German Federal Ministry of Education and Research (BMBF). Grant number: 16KIS0240. We thank Leon Alexander Hermann and Ehud Cseresnyes for their contribution to the prototype implementation.

Appendix A. Proofs for Robustness

The following section contains proofs for robustness for logical transformations of the ACTree structure. The most important rules for logical transformations are the associative and the distributive law. Proofs for both are presented in the following. Proofs for more trivial rules like the commutative law are not covered here. All possible logical transformations are based on these basic transformations. It is shown that the probability of attack success for a scenario is independent from the representation of equivalent tree structures. Because the probability of attack success (PS) depends on the probability of attack initiation (PI) (see Eq. 7), for each proof it is first shown that PI is the same for different equivalent tree structures; then, the same is done for PS.

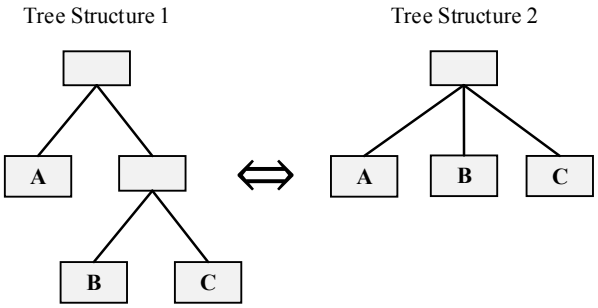


Figure A.14: Logical Transformation with Respect to the Associative Law

PI functions reflect different attacker models. They come into place in case of OR operations where an attacker can choose between different attack options. To proof the robustness for any PI function they are modelled with the generic function g. On the other hand, AND operations are modelled with function f.

The proofs also make use of the fact that, due to the logical transformations, the parameters of the nodes (here A, B and C) are the same for different equivalent tree representations.

1. **First Proof for Equivalence of Logical Tree Transformations with Respect to the Associative Law**

   First, the robustness of logical transformations is shown for the first variant of the associative law. Both equivalent tree structures are presented in Fig. A.14.

   (a) Probability of Initiation:
   According to Eq. 7, PI for tree structure 1 is represented by (A.1).

$$PI^1 = g(PI_A, g(PI_B, PI_C)) \qquad \text{(A.1)}$$

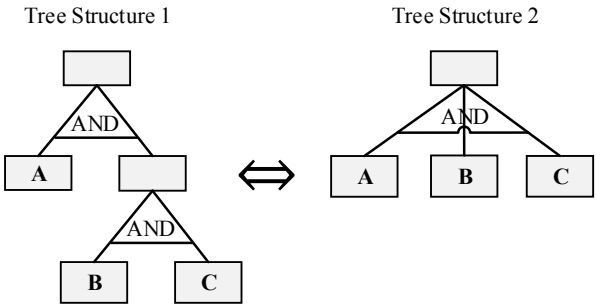


Figure A.15: Logical Transformation with Respect to the Associative Law

PI for tree structure 2 is represented by (A.2).

$$PI^2 = g(PI_A, PI_B, PI_C) \tag{A.2}$$

Therefore, assuming that function g is associative, PI^1 and PI^2 are the same for both tree structures.

(b) Probability of Attack Success:
According to Eq. 7, PS for tree structure 1 is calculated as shown in (A.3).

$$PS^1 = PI_A PS_A + \sum_{j \in J} (PI_j PS_j) \tag{A.3}$$

$$PS^1 = PI_A PS_A + PI_B PS_B + PI_C PS_C \tag{A.4}$$

PS for structure 2 is represented by (A.5).

$$PS^2 = \sum_{j \in J} (PI_j PS_j) \tag{A.5}$$

$$PS^2 = PI_A PS_A + PI_B PS_B + PI_C PS_C \tag{A.6}$$

Because $PS^1 = PS^2$, the probability of attack success is the same for both equivalent tree structures.

2. **Second Proof for Equivalence of Logical Tree Transformations with Respect to the Associative Law**

The second possible logical transformation with regard to the associative law is depicted Fig. A.15. Because the tree does not contain any OR operations the attacker does not have any attack decision. Therefore, the probability of attack initiation is 1 for all nodes.

According to Eq. 7, PS for tree structure 1 is calculated as shown in (A.7).

$$PS^1 = PS_A + \sum_{j \in J} PS_j \tag{A.7}$$

$$PS^1 = PS_A + PS_B + PS_C \tag{A.8}$$

PS for structure 2 is represented by (A.9).

$$PS^2 = \sum_{j \in J} PS_j \tag{A.9}$$

$$PS^2 = PS_A + PS_B + PS_C \tag{A.10}$$

Because $PS^1 = PS^2$, the probability of attack success is the same for both equivalent tree structures.

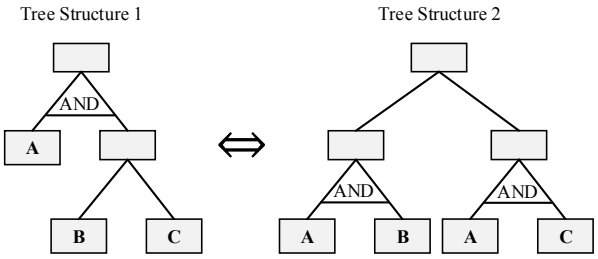


Figure A.16: Logical Transformation with Respect to the Distributive Law

3. **First Proof for Equivalence of Logical Tree Transformations with Respect to the Distributive Law**

The same is done for the distributive law. The equivalent tree structures are illustrated in Fig. A.16.

(a) Probability of Initiation:
PI for tree structure 1 is represented by (A.11) where the AND operations are modelled with function f and OR are modelled with function g.

$$PI^1 = f(PI_A, g(PI_B, PI_C)) \tag{A.11}$$

PI for structure 2 is represented by (A.12).

$$PI^2 = g(f(PI_A, PI_B), f(PI_A, PI_C)) \tag{A.12}$$

For any function g that is distributive in respect to a function f, (A.13) applies.

$$PI^2 = f(PI_A, g(PI_B, PI_C)) \tag{A.13}$$

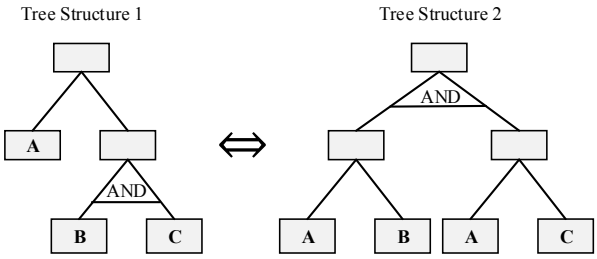


Figure A.17: Logical Transformation with Respect to the Distributive Law

Because $PI^1 = PI^2$, the probability of attack initiation is the same for both equivalent tree structures.

(b) Probability of Attack Success:
PS for tree structure 1 is calculated in (A.14).

$$PS^1 = PS_A \sum_{j \in J} (PI_j PS_j) \quad \text{(A.14)}$$

$$PS^1 = PS_A(PI_B PS_B + PI_C PS_C) \quad \text{(A.15)}$$

$$PS^1 = PI_B PS_A PS_B + PI_C PS_A PS_C \quad \text{(A.16)}$$

(A.16) can be transformed in the way that the nodes A and B resp. A and C are merged. This transformation demonstrates that PS^1 equals PS^2. The notation PI_{AB} refers to PI for the parent node of node A and B. The same holds for PS_{AB}.

$$PS^1 = PI_{AB} PS_{AB} + PI_{AC} PS_{AC} = PS^2 \quad \text{(A.17)}$$

4. **Second Proof for Equivalence of Logical Tree Transformations with Respect to the Distributive Law**

The second possible logical transformation with regard to the distributive law is depicted Fig. A.17.

(a) Probability of Initiation:
PI for tree structure 1 is represented by (A.18).

$$PI^1 = g(PI_A, f(PI_B, PI_C)) \quad \text{(A.18)}$$

PI for structure 2 is represented by (A.19).

$$PI^2 = f(g(PI_A, PI_B), g(PI_A, PI_C)) \quad \text{(A.19)}$$

For any function f that is distributive in respect to a function g, (A.20) applies.

$$PI^2 = g(PI_A, f(PI_B, PI_C)) \quad \text{(A.20)}$$

Because $PI^1 = PI^2$, the probability of attack initiation is the same for both equivalent tree structures.

(b) Probability of Attack Success:
PS for tree structure 1 is calculated in (A.21) and is transformed into (A.23).

$$PS^1 = PI_A PS_A + PI_{BC} \sum_{j \in J} PS_j \quad \text{(A.21)}$$

$$PS^1 = PI_A PS_A + PI_B PS_B \times PI_C PS_C \quad \text{(A.22)}$$

$$PS^1 = PI_A PS_A + PI_{BC}(PS_B PS_C) \quad \text{(A.23)}$$

PS for tree structure 2 is represented by (A.24) and is transformed into (A.26)

$$PS^2 = (PI_A PS_A + PI_B PS_B)(PI_A PS_A + PI_C PS_C) \quad \text{(A.24)}$$

$$PS^2 = PI_A PS_A \times PI_A PS_A + PI_A PS_A \times PI_B PS_B + PI_A PS_A \times PI_C PS_C + PI_B PS_B \times PI_C PS_C \quad \text{(A.25)}$$

$$PS^2 = PI_A PS_A(PI_A PS_A + PI_B PS_B + PI_C PS_C) + PI_B PS_B \times PI_C PS_C \quad \text{(A.26)}$$

The present equations represent logical statements. Therefore, (A.26) can be simplified into (A.27).

$$PS^2 = PI_A PS_A + PI_B PS_B \times PI_C PS_C \quad \text{(A.27)}$$

Because $PS^1 = PS^2$, the probability of attack success is the same for both equivalent tree structures.

References

[1] Australien Cyber Security Centre (ACSC), Essential eight maturity model, https://www.cyber.gov.au/publications/essential-eight-maturity-model, 2019.

[2] National Cyber Security Centre, Uk's cyber essentials scheme, https://www.cyberessentials.ncsc.gov.uk/, 2019.

[3] C. Alberts, A. Dorofee, J. Stevens, C. Woody, OCTAVE-S implementation guide, version 1.0, Pittsburgh, PA, Carnegie Mellon University (2005).

[4] A. Shameli-Sendi, R. Aghababaei-Barzegar, M. Cheriet, Taxonomy of information security risk assessment (isra), Comput. Secur. 57 (2016) 14–30.

[5] National Electric Sector Cybersecurity Organization Resource (NESCOR), Electric Sector Failure Scenarios and Impact Analyses, Technical Report, 2013.

[6] B. Kordy, P. Kordy, S. Mauw, P. Schweitzer, Adtool: Security analysis with attack–defense trees, in: K. Joshi, M. Siegle, M. Stoelinga, P. R. D'Argenio (Eds.), Quantitative Evaluation of Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 173–176.

[7] M. Benaroch, Real options models for proactive uncertainty-reducing mitigations and applications in cybersecurity investment decision making, Information Systems Research 29 (2018) 315–340.

[8] C. Schmitz, A. Sekula, S. Pape, V. Pipek, K. Rannenberg, Easing the burden of security self-assessments, in: 12th International Symposium on Human Aspects of Information Security & Assurance, HAISA 2018 ,Dundee, Scotland, August 29-31, 2018, Proceedings.

[9] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), ISO/IEC 27005 Information technology – Security techniques – Information security risk management, Technical Report, 2008.

[10] National Institute for Standards and Technology (NIST), 800-30 Rev. 1: Guide for Conducting Risk Assessments, Technical Report, 2012.

[11] B. Schneier, Attack trees, Dr. Dobbs journal 24 (1999) 21–29.

[12] S. Mauw, M. Oostdijk, Foundations of attack trees, in: International Conference on Information Security and Cryptology, Springer, pp. 186–198.

[13] B. Kordy, L. Piètre-Cambacédès, P. Schweitzer, DAG-based attack and defense modeling: Don't miss the forest for the attack trees, Computer Science Review 13-14bb (2014) 1–38.

[14] M. Davarynejad, M. Ford, D. Hadziosmanovic, O. Gadyatskaya, R. Hansen, D. Ionita, H. Jonkers, A. Lenin, Z. Lukszo, S. Mauw, B. Othman, W. Pieters, C. Probst, A. Tanner, R. Trujillo, J. van den Berg, J. Willemson, Technology-supported Risk Estimation by Predictive Assessment of Socio-technical Security, Best practices for model creation and sharing (Deliverable D5.3.2), Technical Report, 2015.

[15] National Electric Sector Cybersecurity Organization Resource (NESCOR), Analysis of Selected Electric Sector High Risk Failure Scenarios, Technical Report, 2013.

[16] S. Bistarelli, F. Fioravanti, P. Peretti, Defense trees for economic evaluation of security investments, in: Availability, Reliability and Security, 2006. ARES 2006. The First International Conference on, pp. 8 pp.–.

[17] A. Roy, D. S. Kim, K. S. Trivedi, Cyber security analysis using attack countermeasure trees, in: F. T. Sheldon, S. J. Prowell, R. K. Abercrombie, A. W. Krings (Eds.), Proceedings of the 6th Cyber Security and Information Intelligence Research Workshop, CSIIRW 2010, Oak Ridge, TN, USA, April 21-23, 2010, ACM, 2010, p. 28.

[18] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), ISO/IEC 27002:2013, information technology – security techniques – code of practice for information security controls (2013).

[19] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), ISO/IEC 27019:2017, information technology – security techniques – information security controls for the energy utility industry (2017).

[20] A. Sengupta, Modeling dependencies of iso/iec 27002:2013 security controls, in: J. H. Abawajy, S. Mukherjea, S. M. Thampi, A. Ruiz-Martínez (Eds.), Security in Computing and Communications, Springer International Publishing, Cham, 2015, pp. 354–367.

[21] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), ISO/IEC 15504-5:2012, information technology – process assessment - part 5: An exemplar software life cycle process assessment model, 2012.

[22] Information Systems Audit and Control Association (ISACA), CobiT 5: A Business Framework for the Governance and Management of Enterprise IT, Rolling Meadows, 2012.

[23] Verband der Automobilindustrie (VDA), Information security assessment, https://www.vda.de/de/services/Publikationen/information-security-assessment.html, 2019, Accessed 26 February 2019.

[24] International Organization for Standardization (ISO), International Electrotechnical Commission (IEC), ISO/IEC 271827:2008, information technology – systems security engineering - maturity model (sse-cmm) (2013).

[25] M. B. Chrissis, M. Konrad, S. Shrum, CMMI for Development: Guidelines for Process Integration and Product Improvement, Addison-Wesley Professional, 3rd edition, 2011.

[26] T. R. Ingoldsby, Attack tree-based threat risk analysis (2013).

[27] G. F. O. for Information Security (BSI), Zuordnungstabelle ISO 27001 sowie ISO 27002 und IT-Grundschutz, Technical Report, 2011.

[28] M. C. Paulk, B. Curtis, M. B. Chrissis, C. V. Weber, Capability maturity model, version 1.1, IEEE software 10 (1993) 18–27.

[29] Z. W. Birnbaum, On the importance of different components in a multicomponent system, Technical Report, Washington Univ Seattle Lab of Statistical Research, 1968.

[30] Carnegie Mellon University - Software Engineering Instituteee, Process maturity profile cmmi for development scampi class a appraisal results - 2008 end-year update, Presentation, 2009.

[31] M. Brecht, T. Nowey, A Closer Look at Information Security Costs, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 3–24.

[32] P. Pu, L. Chen, R. Hu, A user-centric evaluation framework for recommender systems, in: Proceedings of the fifth ACM conference on Recommender systems, ACM, pp. 157–164.

[33] J. Dax, B. Ley, S. Pape, C. Schmitz, V. Pipek, K. Rannenberg, Elicitation of requirements for an inter-organizational platform to support security management decisions, in: 10th International Symposium on Human Aspects of Information Security & Assurance, HAISA 2016 ,Frankfurt, Germany, July 19-21, 2016, Proceedings.

[34] V. Verendel, Quantified security is a weak hypothesis: A critical survey of results and assumptions, in: Proceedings of the 2009 Workshop on New Security Paradigms Workshop, NSPW '09, ACM, New York, NY, USA, 2009, pp. 37–50.

[35] S. Pfleeger, R. Cunningham, Why measuring security is hard, IEEE Security Privacy 8 (2010) 46–54.

[36] ENX Association, Trusted information security assessment exchange (tisax), http://enx.com/tisax/tisax-en.html, 2019, Accessed 26 February 2019.

[37] European Union Agency for. Network and Information Security (ENISA), Risk Management: Implementation principles and Inventories for Risk Management/Risk Assessment methods and tools, Technical Report, 2006.

[38] S. Fenz, J. Heurix, T. Neubauer, F. Pechstein, Current challenges in information security risk management, Information Management & Computer Security 22 (2014) 410–430.

[39] A. Behnia, R. A. Rashid, J. A. Chaudhry, A survey of information security risk analysis methods, SmartCR 2 (2012) 79–94.

[40] D. Ionita, Current established risk assessment methodologies and tools, Master's thesis, University of Twente, 2013.

[41] S. M. Sulaman, K. Weyns, M. Höst, A review of research on risk analysis methods for it systems, in: Proceedings of the 17th International Conference on Evaluation and Assessment in Software Engineering, EASE '13, ACM, New York, NY, USA, 2013, pp. 86–96.

[42] C. Harpes, G. Schaff, M. Martins, B. Kordy, R. Trujillo, D. Ionita, Technology-supported Risk Estimation by Predictive Assessment of Socio-technical Security, Currently established risk-assessment methods (Deliverable D5.2.1), Technical Report, 2014.
[43] R. Bojanc, B. Jerman-Blažič, A quantitative model for information-security risk management, Engineering management journal 25 (2013) 25–37.
[44] J. Jones, FAIR - ISO/IEC 27005 Cookbook, Technical Report, The Open Group, 2010.
[45] W. Pieters, M. Davarynejad, Calculating adversarial risk from attack trees: Control strength and probabilistic attackers, in: Data Privacy Management, Autonomous Spontaneous Security, and Security Assurance, Springer, 2015, pp. 201–215.
[46] B. Karabacak, I. Sogukpinar, Isram: information security risk analysis method, Computers & Security 24 (2005) 147–159.
[47] M. Schmid, S. Pape, A structured comparison of the corporate information security maturity level, in: G. Dhillon, F. Karlsson, K. Hedström, A. Zúquete (Eds.), ICT Systems Security and Privacy Protection, Springer International Publishing, Cham, 2019, pp. 223–237.
[48] R. Anderson, Why information security is hard-an economic perspective, in: Computer security applications conference, 2001. acsac 2001. proceedings 17th annual, IEEE, pp. 358–365.
[49] L. A. Gordon, M. P. Loeb, The economics of information security investment, ACM Trans. Inf. Syst. Secur. 5 (2002) 438–457.
[50] D. Schatz, R. Bashroush, Economic valuation for information security investment: a systematic literature review, Information Systems Frontiers 19 (2017) 1205–1228.
[51] T. Neubauer, C. Hartl, On the singularity of valuating it security investments, in: Proceedings of the 2009 Eigth IEEE/ACIS International Conference on Computer and Information Science, ICIS '09, IEEE Computer Society, Washington, DC, USA, 2009, pp. 549–556.
[52] K. Ruan, Introducing cybernomics: A unifying economic framework for measuring cyber risk, Computers & Security 65 (2017) 77–89.
[53] A. Mancuso, P. Zebrowski, A. C. Vieira, Risk-based selection of mitigation strategies for cybersecurity of electric power systems, IEEE TRANSACTIONS ON DEPENDABLE AND SECURE COMPUTING (2019).
[54] O. Gadyatskaya, C. Harpes, S. Mauw, C. Muller, S. Muller, Bridging two worlds: reconciling practical risk assessment methodologies with theory of attack trees, in: International Workshop on Graphical Models for Security, Springer, pp. 80–93.
[55] L. Demetz, D. Bachlechner, To invest or not to invest? assessing the economic viability of a policy and security configuration management tool, in: The Economics of Information Security and Privacy, 2013, pp. 25–47.
[56] T. Sawik, Selection of optimal countermeasure portfolio in it security planning, Decis. Support Syst. 55 (2013) 156–164.
[57] N. Tsalis, M. Theoharidou, D. Gritzalis, Return on security investment for cloud platforms, in: Proceedings of the 2013 IEEE International Conference on Cloud Computing Technology and Science - Volume 02, CLOUDCOM '13, IEEE Computer Society, Washington, DC, USA, 2013, pp. 132–137.
[58] A. M. Nhlabatsi, J. B. Hong, D. S. D. Kim, R. Fernandez, A. Hussein, N. Fetais, K. M. Khan, Threat-specific security risk evaluation in the cloud, IEEE Transactions on Cloud Computing (2018).