Employing Graphical Risk Models to Facilitate Cyber-Risk Monitoring – the WISER Approach

Aleš Černivec[1], Gencer Erdogan[2], Alejandra Gonzalez[3], Atle Refsdal[2], and Antonio Alvarez Romero[4]

[1] XLAB Research, Ljubljana, Slovenia ales.cernivec@xlab.si
[2] SINTEF Digital, Oslo, Norway {gencer.erdogan,atle.refsdal}@sintef.no
[3] AON, Milan, Italy alejandra-gonzalez@alice.it
[4] ATOS, Sevilla, Spain antonio.alvarez@atos.net

Abstract. We present a method for developing machine-readable cyber-risk assessment algorithms based on graphical risk models, along with a framework that can automatically collect the input, execute the algorithms, and present the assessment results to a decision maker. This facilitates continuous monitoring of cyber-risk. The intended users of the method are professionals and practitioners interested in developing new algorithms for a specific organization, system or attack type, such as consultants or dedicated cyber-risk experts in larger organizations. For the assessment results, the intended users are decision makers in charge of countermeasure selection from an overall business perspective.

Keywords: Cyber Risk, Security, Risk Modelling, Risk Assessment, Risk Monitoring.

1 Introduction

Cybersecurity is of critical importance for small businesses, large companies, public administrations and everyone involved in the digital economy. Millions of euros are lost to cyber-crime each year. Online security is a growing concern for businesses, with attacks increasing against large corporate business and critical infrastructures, but also against small enterprises that lack the time, money and human resources to dedicate to consolidating their cyber-risk management.

There are currently a number of tools available to support cyber-risk management. Most such tools focus on the technical aspects of detecting vulnerabilities and attacks, without relating these to the wider business context of the organization. This may provide useful support for IT administrators. However, managers and decision makers need to understand the impact of cyber-risks on their business objectives in order to determine how to deal with them from a more strategic perspective. This impact can be expressed either quantitatively or qualitatively. Quantitative estimates of the likelihood of incidents and the consequence in terms of money allow risks to be weighed against the cost of

available countermeasures. Unfortunately, providing trustworthy numbers can be very difficult, as this requires access to good empirical data and statistics to serve as a foundation for quantified estimates. Such data is often unavailable. Even if we can obtain the data, analyzing it to understand its impact on the assessment is a major challenge [32]. This means that providing good quantitative assessments is not always feasible. In such cases, a qualitative approach can be a good alternative. By qualitative, we mean that we use ordinal scales, for which the standard arithmetic operators are not defined, to provide assessments. Each step is usually described by text, such as {Very low; Low; Medium; High; Very high}. More informative descriptions of each step can of course be given. Ordinal scales imply that values are ordered, thereby making it possible to monitor trends.

Cyber-risks depend on many different factors, many of which are highly technical. Moreover, the risks are continuously changing due to updates in the target ICT infrastructure or the way it supports the business, discovery of new vulnerabilities, and a rapidly evolving threat landscape. Therefore, rather than providing a snapshot representing one point in time, we want to facilitate monitoring by providing automated updates of risk level assessments based on dynamic input that captures vulnerabilities, events observed in the target ICT infrastructure, as well as the business configuration. To achieve this, we need executable algorithms that define how the risk level assessments change as a result of changes in the dynamic input, as well as an assessment infrastructure that can automatically collect the input, execute the algorithms, and present the results.

Developing risk level assessment algorithms requires a good understanding of the relevant threats, threat scenarios, vulnerabilities, incidents and assets. The CORAS risk modelling approach has proved to be well suited for establishing such an understanding and supporting risk level assessments [21, 35]. However, the CORAS approach was created to perform manual assessments representing a single point in time, rather than establishing algorithms for automated assessments.

In this paper, we present a method for developing quantitative and qualitative cyber-risk assessment algorithms by exploiting the structure of CORAS risk models, together with a cyber-risk monitoring framework that automatically collects the dynamic input, executes the algorithms, and presents the results. The method and framework were developed in the WISER project [39].

In the following, we start by presenting the cyber-risk monitoring framework in Sect. 2. We then give an overview of the method for risk modelling, which includes the algorithm development, in Sect. 3. The method consists of two main steps. Section 4 and Sect. 5 presents each step in more detail. In Sect. 6 we present related work, before discussing and concluding in Sect. 7.

2 Cyber-Risk Monitoring Framework

In this section, we present the framework developed in the WISER project to assess and monitor cyber-risk for companies from the business perspective. Figure 1 shows an overview of the essential components in the framework.

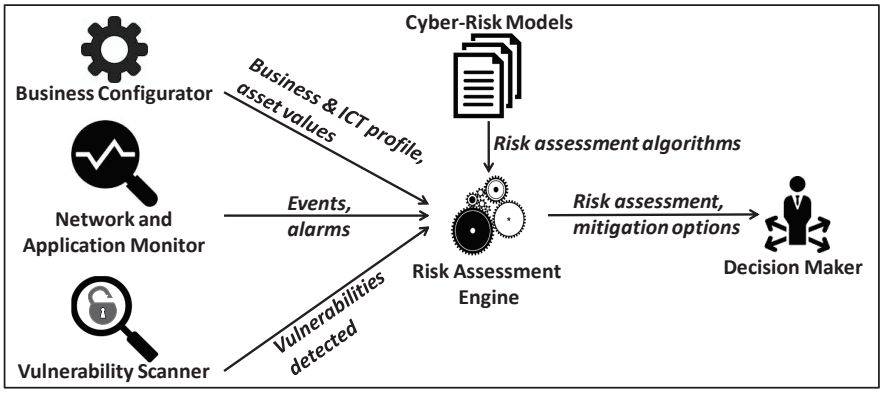

Fig. 1. Overview of the WISER framework.

The brain of the WISER framework is the Risk Assessment Engine. It produces an assessment of the risk the company faces, based on a cyber-risk model with an associated risk assessment algorithm. As illustrated by the left-hand side of Fig. 1, there are three different types of dynamic input to this algorithm. By "dynamic" we mean that the input values may change between each execution of the algorithm. The first input type is provided by the user configuring the framework for the company via a business configuration interface. The second input type is events and alarms obtained from monitoring the network and application layers of the ICT infrastructure, while the third input type is information about detected vulnerabilities provided by vulnerability scanners. We use the term *indicator* to denote the inputs to the algorithm. The output of the Risk Assessment Engine is an assessment of quantitative and/or qualitative risk levels. In addition, a proposal for mitigation options will also be triggered if risk levels exceeds a set threshold. However, mitigation options are beyond the scope of this paper. The reader is referred to [31] for further details on this.

In the remainder of this section, we explain how the input to the assessment algorithms, i.e. the indicator values, are obtained.

2.1 Business Configurator

As already noted, the first input type in Fig. 1 is provided by the user configuring the framework for a client/organization. This is done through a Business Configuration interface by answering general questions about the business, ICT profile

and security management of the organization. Furthermore, the user is asked to provide information about the machines and applications to be protected, which we refer to as the targets of analysis.

For each target of analysis, the user assigns a level of importance with respect to confidentiality, integrity, and availability, which will depend on the way in which this target supports the business processes. The user also characterizes each target based on the ACM Computing Classification System [1], which serves as the de facto standard classification system for the computing field.

Finally, for clients who wish to obtain quantitative cyber-risk assessments, the user configuring the framework is asked to provide, for each target of analysis, a typical and worst-case loss potentially resulting from a successful cyber-attack. If these values are not provided, the framework resorts to using default values defined for a typical European SME, as estimated by the WISER Consortium [31].

2.2 Network and Application Monitor

The Network and Application Monitor module in the WISER framework provides the most dynamic input to the cyber-risk assessment, as the monitoring occurs in real time. The module consists of sensors which generate events (messages announcing unusual activity or values of observed metrics) and a Monitoring Engine, which generates alarms by correlating and combining several events. Various sensors, installed on the client's infrastructure, continuously observe numerous network and application-level parameters to provide input values for risk assessment indicators. Both events and alarms are fed into the Risk Assessment Engine. Alarms can also be used independently from the Risk Assessment Engine to notify the responsible users about ongoing attacks or emerging security threats.

The monitoring architecture incorporates two layers: the Resource Layer and the Provider Layer. The Resource Layer consists of WISER Agents and sensors installed on the client's infrastructure, providing data about the infrastructure to the Provider Layer. The Provider Layer supports the monitoring capabilities with back-end core services for event aggregation and correlation and a central data storage facility, serving monitoring data to the Risk Assessment Engine.

Monitoring sensors are able to detect several types of attacks and anomalies in the network infrastructure as well as in applications installed on the client's premises. The following sensors, which are further described in [36], are employed by WISER:

- DNS Traffic Sensor: monitors DNS requests to detect patterns of traffic that potentially belong to botnets.
- Snort: a network-based intrusion detection system, detecting network reconnaissance attempts, malware signatures and denial of service attacks.
- OSSEC: a host-based intrusion detection system, monitoring application-level activity and detecting anomalies in operation of core operating system services and user applications, recognizing viruses and attackers.

– Cowrie: an SSH-based honeypot used to attract attackers and detect their presence while averting attacks from other machines on the network.

The WISER Agents are responsible for the collection, normalization and transfer of data of the events to the provider layer. The Agents gather the data from sensors installed on the same network through encrypted syslog channels, adds common information about the organization, and forwards the messages in a common format to the provider layer communication bus, supported by the AMQP-based RabbitMQ server. RabbitMQ is responsible for message queuing and distribution to other components on the monitoring provider layer.

The Monitoring Engine, part of the Provider layer, is composed of a SIEM (Security Information and Event Management) solution and a correlation engine that provides continuous analysis of security-related events, aggregating data from the sensors and generating reports and alarms, taking a predefined set of correlation rules and security directives into consideration.

For each generated alarm, the Monitoring Engine computes a risk score, which combines measures of potential attack damage, likelihood of the attack and the value of assets compromised. If the risk score is high enough, the WISER Framework can automatically notify a responsible person. Notice that these risk scores should not be confused with the risk level assessments provided by the Risk Assessment Engine when executing the algorithms addressed in Sects. 3–5. The former provides low-level assessments according to fixed rules to facilitate a quick response by an ICT administrator. The latter provides more high-level assessments, where several events and alarms from the Monitoring Engine can be considered in conjunction with other types of indicators to provide more ICT and business context, and where the algorithms can be tailored to a specific organization or system.

2.3 Vulnerability Scanner

The Vulnerability Scanner module automatically identifies security vulnerabilities in the client's web applications. The Vulnerability Scanner acts similarly as a monitoring sensor, installed at the clients premises. It scans specified target websites periodically (in a configured time interval) and reports the results to the Risk Assessment Engine, like monitoring sensors. The vulnerabilities found by each scan can also be seen in the WISER Dashboard along with their mitigation suggestions. The vulnerability scanner in this mode of operation can scan public websites as well as web applications that are only accessible from inside the organization's network. This means that it can be used to test specialized web applications and also websites during their development process.

The Vulnerability Scanner is based on combining results from several tools for vulnerability scanning, such as W3af [38] and OWASP ZAP [29]. These tools automatically gather responses from the targeted website and compare them to their databases of known vulnerabilities to find which vulnerabilities might be present in the web application. The vulnerability databases contain explanations of the vulnerabilities and their mitigation proposals, which are included in the reports.

3 Method for Cyber-Risk Modelling

In this section, we explain the overall method for risk modelling. The risk modelling described here will typically be a part of a wider risk management process, such as ISO 31000 [17], and can be "plugged into" any such process. We focus on the methodological aspects that are special in our context, which are the following:

- For risk level assessment, the goal is not to perform an assessment representing a snapshot of one particular point or period in time, but rather to develop algorithms for automated assessment. The algorithms can be either qualitative or quantitative.
- Identification of threats, vulnerabilities, threat scenarios, incidents and risks is done using CORAS diagrams (models).
- For the elements of a risk model described above, we also identify the dynamic factors that can be provided by the framework, i.e. the indicators, as explained in Sect. 2. The indicators serve as input for the assessment algorithms.

Figure 2 shows the overall method used for cyber-risk modelling, considering these aspects. The outcome of the first step is a validated CORAS diagram with

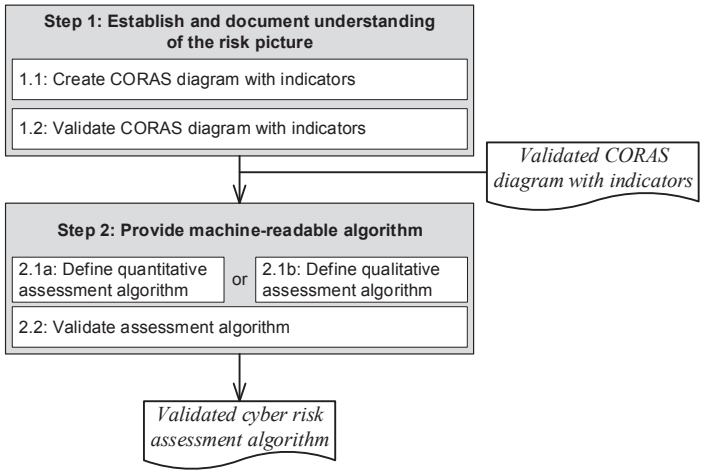

Fig. 2. Method for cyber-risk modelling.

indicators. This diagram captures the relevant assets, risks, the ways in which these risks may materialize, and the relation between these elements and the available business configuration indicators, vulnerability scan indicators, network monitoring indicators and application monitoring indicators that can be employed to assess the risk and the involved threats, vulnerabilities and threat scenarios.

The outcome of the second step is a machine-readable algorithm for risk level assessment (and mitigation proposals) that can be automatically executed by the Risk Assessment Engine. The dynamic input for this assessment algorithm consists of the indicators identified in the first step. In the next two sections, we explain the steps in further detail.

4 Step 1: Establish and Document Understanding of the Risk Picture

4.1 Step 1.1: Create CORAS diagram with indicators

As illustrated by Fig. 2, this step is the same irrespective of whether the aim is to develop a qualitative or a quantitative assessment algorithm. The reason is that the purpose of this particular step is not to assess risk levels or define an assessment algorithm, but to identify the potential chains of events that may lead to risks materializing. This includes identifying all the threats, vulnerabilities, threat scenarios and incidents involved in such chains. Moreover, we identify the indicators that can provide information about all risk elements that can serve as useful input for the assessment algorithm to be developed in Step 2.

For creating security risk models, we use CORAS [21], which is a graphical risk modeling language that has been empirically shown to be intuitively simple for stakeholders with very different backgrounds [35]. Moreover, CORAS comes with a method that builds on established approaches (in particular ISO 31000 [17]), and includes detailed guidelines for creating CORAS models, which can be applied to carry out Step 1.

Figure 3 gives an overview of the CORAS notation. Threats, threat scenarios, unwanted incidents, assets, relations and vulnerabilities are collectively used to create CORAS risk models, which document risks as well as events and circumstances that can cause risks. Notice that the different relations are used to connect different nodes: the *initiates* relation goes from a threat to a threat scenario or an unwanted incident. The *leads-to* relation goes from a threat scenario or an unwanted incident to a threat scenario or an unwanted incident. The *impacts* relation goes from an unwanted incident to an asset. The indicator construct, which is not part of the standard CORAS notation, is introduced to capture dynamic factors that are obtained by the framework, as explained in Section 2.

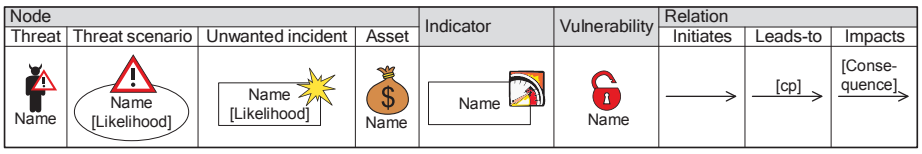

Fig. 3. CORAS notation. cp=conditional probability.

To support risk estimation, CORAS uses likelihood values, conditional probabilities, and consequence values (enclosed in brackets) on certain nodes and relations, as illustrated in Fig. 3. These will be represented by variables in the risk assessment algorithms. It is therefore useful to establish a naming convention for the variables, as well as for the nodes and indicators in the diagram. Table 1 shows our naming convention.

Table 1. Naming conventions for defining likelihood and consequence variables. The letters x and y represent integers.

Name	Meaning
Ax	Asset x
Sx	Scenario x ("S" means threat scenario)
Ux	Incident x ("U" means unwanted incident)
IN-x	Indicator x
l_Ux	Likelihood of Ux
l_Sx	Likelihood of Sx
c_Ux_Ay	Consequence of Ux for Ay
cp_Sx_to_Sy	Conditional probability of Sx leading to Sy
cp_Sx_to_Uy	Conditional probability of Sx leading to Uy

Figure 4 shows a CORAS risk model for a session hijacking in the context of web-applications. This risk model is (a slightly simplified version of) one of 10 risk models we developed in the WISER project [39]. These risk models were not developed for a particular target of analysis, but primarily intended for an arbitrary European SME. Notice that some of the indicators appear more than once, as they are attached to more than one element.

Indicators are normally identified after the assets, threats, threat scenarios, unwanted incidents and vulnerabilities. Indicator identification is not covered by the standard CORAS method [21]. We therefore present here the guiding questions used for this purpose:

- What observable events at the network layer could give useful information about the likelihood/frequency of attacks? (Network monitoring indicators.) This question should be asked for each identified threat scenario and incident.
- What observable events at the application layer could give useful information about the likelihood/frequency of successful or unsuccessful attacks? (Application monitoring indicators.) This question should be asked for each identified threat scenario and incident.
- What information can we get from vulnerability scanners or security tests? (Test result indicators). This question should be asked for each identified vulnerability.
- What do we otherwise know about the threats, vulnerabilities, threat scenarios, incidents or assets that could help us assess the level of cyber-risk? (Business configuration indicators.) These questions should be asked for each element of the risk model.

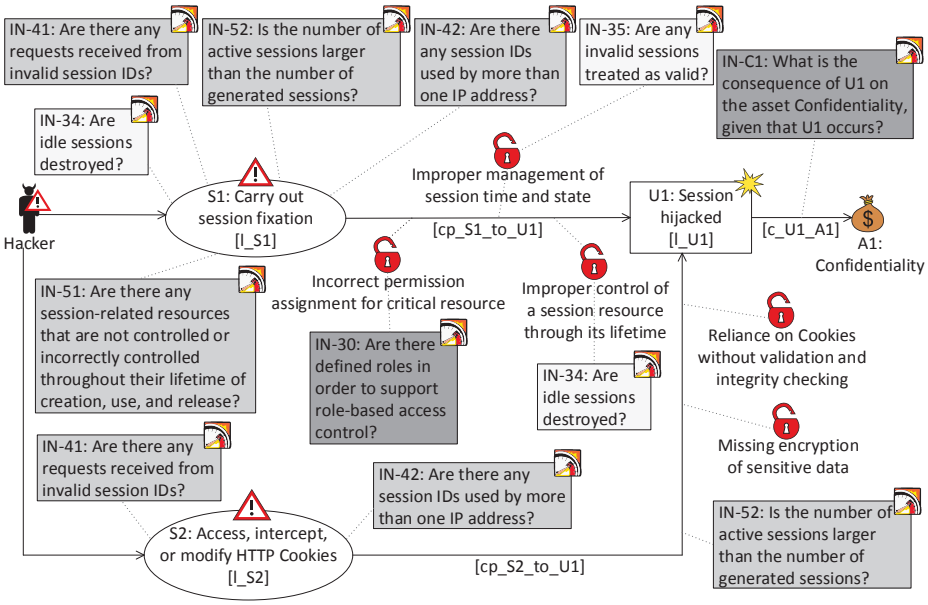

Fig. 4. CORAS risk model for *Session hijacking*. *IN-34* and *IN-35* are vulnerability scan indicators. *IN-30* and *IN-C1* are business configuration indicators. *IN-41*, *IN-42*, *IN-51*, and *IN-52* are application monitoring indicators.

4.2 Step 1.2: Validate CORAS Diagram with Indicators

The CORAS diagram provided in Step 1.1 serves as the basis for developing the machine-readable algorithm in Step 2. Therefore, before moving on, it is essential to ensure that the CORAS diagram reflects, as far as possible, the actual reality with respect to potential threats, vulnerabilities, threat scenarios and risks. Of course, as risk assessments concern what might happen in the future, there is no way we could ensure that a CORAS diagram (or any other form of risk model) is objectively correct and complete with respect to reality. Instead, what we aim for here is a convincing argument that the diagram reflects available knowledge and beliefs among qualified cybersecurity experts. Such an argument can be established, for example, by showing that the CORAS diagram faithfully captures information available from well-reputed standards, repositories, text books, research papers or similar sources; some examples include ISO 27001 [15], ISO 27005 [18], ISO 27032 [16], CAPEC [23] and OWASP [28]. If possible, the validation of the CORAS diagram should be carried out by a group of cybersecurity experts who, after relevant information sources have been identified and obtained, go through each part of the diagram in a systematic manner to identify elements that need to be added, removed, or otherwise improved. The validation terminates when no such elements are found.

5 Step 2: Provide Machine-Readable Algorithm

In the following, we explain how to define quantitative (Sect. 5.1) and qualitative (Sect. 5.2) assessment algorithms based on a CORAS model, before briefly discussing how to validate the results (Sect. 5.3). Notice that the two types of algorithms are independent alternatives, and readers who are only interested in one type can skip the other section. For both alternatives, our focus here is on establishing the algorithm structure. Therefore, we do not go into details regarding estimations that must be done in order to complete the algorithms, but provide references where further information can be found where relevant.

5.1 Step 2.1, alt. a: Define quantitative assessment algorithm

For defining quantitative assessment algorithms we follow an actuarial approach, where the likelihood (frequency) and consequence (economic loss) of unwanted incidents are modelled separately through the probabilistic framework of *Bayesian Networks* (BN) [25]. More specifically, we use hybrid BNs [24], which means that the random variables are not bound to be discrete or (conditionally) Gaussian.

The algorithm is implemented using the **R** programming language [34] for statistical computing. The underlying calculations are performed by Markov-Chain-Monte-Carlo simulation. However, in this paper we focus on the exploitation of CORAS diagrams to establish the BN structure. Understanding this does not require prior knowledge about the **R** programming language, hence we do not show any **R** code. Detailed guidelines on how to create an **R** script from a CORAS model are provided in [31].

BN skeleton The first step is to define a BN skeleton based on the structure of the CORAS model. Figure 5 shows the BN skeleton reflecting the CORAS model in Fig. 4. A risk captured in a CORAS diagram (by an *impacts* relation from an unwanted incident to an asset) is represented by a childless node in the BN (*R1* in Fig. 5). The overall goal is to compute a risk level for risk nodes, as a function of indicators. Any risk node has two parent nodes: one representing the frequency of the unwanted incident and another representing the consequence for the asset. In our example, the risk node *R1* has the parent nodes *l_U1* and *c_U1_A1*, representing the frequency of the incident *U1* and its consequence for the asset *A1*, respectively.

Nodes representing the frequency of an unwanted incident The frequency of unwanted incidents is calculated following the underlying logic of the CORAS model. The frequency node of an unwanted incident has a parent node for each incoming *leads-to* relation to the incident in the CORAS diagram, representing the likelihood contribution from each incoming relation. For example, node *l_U1* in Fig. 5 has two parent nodes: *l_U1S1* and *l_U1S2*.

The likelihood contribution from each *leads-to* relation depends on the likelihood of its threat scenario (the source node) and the conditional probability

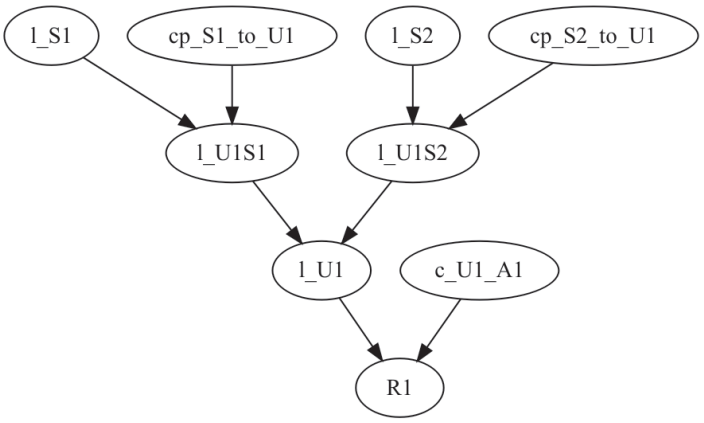

Fig. 5. BN skeleton for CORAS model in Fig. 4.

that an occurrence of this threat scenario will lead to the unwanted incident. Therefore, the node *l_U1S1* depends on the parentless nodes *l_S1* (likelihood of scenario *S1*) and *cp_S1_to_U1* (the conditional probability that an occurrence of *S1* will lead to *U1*). Similarly, *l_U1S2* depends on *l_S2* and *cp_S2_to_U1*. In our example, the probability distribution of the frequency node *l_U1* is defined as follows[5]:

$$\begin{aligned} l\_U1 &= l\_U1S1 + l\_U1S2 \\ &= l\_S1 \cdot cp\_S1\_to\_U1 + l\_S2 \cdot cp\_S2\_to\_U1 \end{aligned}$$

Notice that node *l_U1* is deterministic, since its value at each step of the simulation is calculated by a formula from the values of its parent nodes.

The indicators, which represent the input to the final algorithm, are not included in the BN structure. They will be used to compute the values for the parentless ancestor nodes of *l_U1* (*l_S1*, *l_S2*, *cp_S1_to_U1* and *cp_S2_to_U1* in Fig. 5). These nodes are represented by uniform distributions whose extremes depend on the indicators affecting the nodes. The functions from indicator values to the extremes of the distributions are defined based on expert estimates and available empirical data. We chose uniform distributions in order to ease the estimate elicitation. CORAS uses intervals for the same reason. Since our focus here is on the algorithm structure, we refer to [13] for further details on the estimation.

Nodes representing the consequence of an unwanted incident Consequence nodes model the consequence, in terms of economic loss per occurrence

[5] This formula assumes that the scenarios for the incoming *leads-to* relation are separate, as further explained in [21, p. 224].

of an unwanted incident (node *c_U1_to_A1* in Fig. 5). Here we follow an approach typically adopted in scenario analysis for operational risk management. For a given risk, a two-parameter distribution is chosen for the consequence, and experts are requested to provide a *typical case* loss and a *worst case* loss. This provides the minimal amount of information required to describe the main features of the distribution, that is a value which is experienced frequently and a value which is extreme (experienced rarely). Usually, the typical case loss is identified with a location index, such as the median of the distribution. The worst case loss is identified with a suitably large quantile of the distribution. This gives a nonlinear system of two equations in two variables (the parameters of the distribution), which can be solved by a Newton-like numerical approximation method [10, 2, 26]. We adopted the lognormal distribution for modeling consequence nodes. In modelling loss data, the lognormal distribution is observed to provide good fits in many cases; for this reason it is often used for modelling consequence in operational risk and particularly for the scenario analysis component, see e.g. [19, 11].

Nodes representing risk level Risk level nodes model the yearly aggregate loss distribution, which depend on the frequency and consequence of the unwanted incident. In our example, the risk level is represented by node *R1* in the BN. The probability distribution assigned to *R1* is defined as follows:

$$R1 = l\_U1 \cdot c\_U1\_to\_A1$$

5.2 Step 2.1, alt. b: Define qualitative assessment algorithm

For defining qualitative assessment algorithms we use DEXi [12], which is a computer program for development of multi-criteria decision models and the evaluation of options. We briefly present DEXi before explaining how to create a DEXi model from a CORAS diagram. For a detailed description, we refer to the DEXi User Manual [6].

A multi-attribute model decomposes a decision problem into a tree (or graph) structure where each node in the tree represents an attribute. The overall problem is represented by the top attribute, also called the root. All other attributes represent sub-problems, which are smaller and less complex than the overall problem. Each attribute is assigned a value. The set of values that an attribute can take is called the *scale* of the attribute. DEXi supports definition of ordinal scales; typically, each step consists of a textual description.

Every attribute is either a basic attribute or an aggregate attribute. *Basic attributes* have no child attributes. This means that a basic attribute represents an input to the DEXi model, as its value is assigned directly, rather than being computed from child attributes.

Aggregate attributes are characterized by having child attributes. The value of an aggregate attribute is a function of the values of its child attributes. This function is called the *utility function* of the attribute. The utility function of each

aggregate attribute is defined by stating, for each possible combination of its child attribute values, what is the corresponding value of the aggregate attribute. The DEXi tool automatically computes the value of all aggregate attributes as soon as values have been assigned to the basic attributes. Hence, a DEXi model can be viewed as an algorithm where the basic attribute values constitute the input and the values of the aggregate attributes constitute the output. A java library and a command-line utility program for DEXi model execution is available [12], meaning that functionality for executing DEXi algorithms can be easily integrated in software systems. This, combined with the fact that DEXi has been designed to produce models that are comprehensible to end users [9], was our reason for choosing DEXi. Its comprehensibility seems to be confirmed by its application in several different domains, involving a wide range of stakeholders [7–9].

Figure 6 shows an example of a DEXi model which consists of three aggregate attributes and three basic attributes; the latter are shown as triangles. The top attribute, which is an aggregate attribute, is named *Risk* and has two child attributes (*Likelihood* and *Consequence*) that are also aggregate attributes. The *Likelihood* attribute has in turn two basic attributes as child attributes (*Likelihood indicator 1* and *Likelihood indicator 2*), while the *Consequence* attribute has one basic attribute as child attribute (*Consequence indicator 1*).

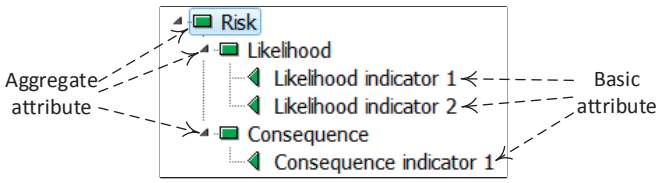

Fig. 6. DEXi model.

We now show how to build a security risk assessment algorithm, in the form of a DEXi model, based on a CORAS model. We use the model in Fig. 4 as an example. This means that the decision problem represented by the top attribute in the DEXi model concerns deciding the risk level. We start by explaining how each fragment of the CORAS model can be schematically translated to a corresponding fragment of the DEXi model. Since our focus here is on the algorithm structure, we do not address the definition of scales and utility functions, but refer to [14] for further discussion on this.

Risk In the CORAS model, a risk corresponds to an *impacts* relation from an unwanted incident to an asset. The risk level depends on the likelihood of the incident and its consequence for the asset, as represented by *l_U1* and *c_U1_A1* in Fig. 4. In the DEXi model, a risk is therefore represented as a top (i.e. orphan) attribute that has two child attributes, one representing the likelihood

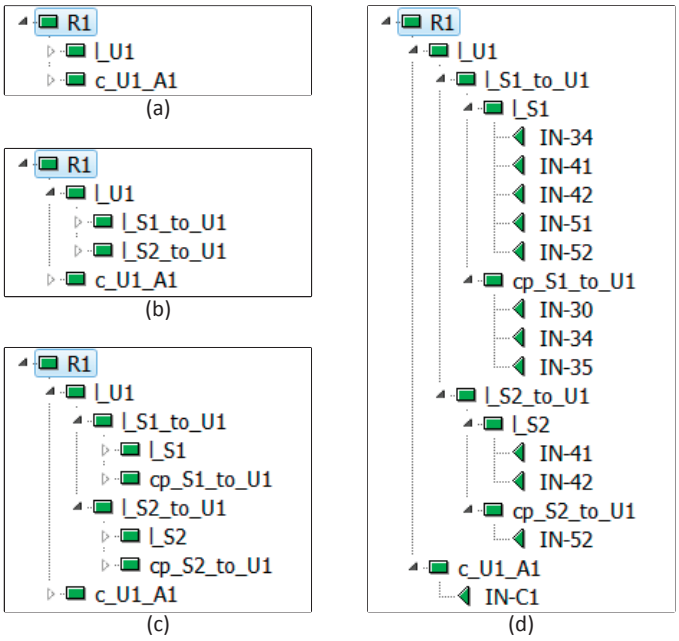

Fig. 7. Screenshots from the DEXi tool.

of the incident and one representing its consequence for the asset. Figure 7(a) shows the DEXi-representation of the (only) risk shown in Fig. 4. The value of the top attribute *R1* represents the risk level. Notice that *R1* does not occur as a separate name in the CORAS diagram, as a risk is represented by the combination of the incident, the asset, and the relation between them, rather than by a separate node.

Node with incoming *leads-to* relations In a CORAS model, the likelihood of a node with incoming *leads-to* relations[6] depends on the likelihood contributions from each relation. In the DEXi model, such a node is therefore represented by an attribute with one child attribute for every incoming *leads-to* relation. The attribute *l_U1* in Fig. 7(b), which represents the likelihood of *U1*, therefore has two child attributes, *l_S1_to_U1* and *l_S2_to_U1*, representing the likelihood contributions from *S1* and *S2* via their outgoing *leads-to* relations.

Node with outgoing *leads-to* relation The contribution from a *leads-to* relation to its target node depends on the likelihood of the source node and the conditional probability that an occurrence of the source node will lead to an occurrence of the target node. The latter is assigned to the *leads-to* relation in

[6] Recall from Sect. 4 that threat scenarios and unwanted incidents are the only node types that may have incoming *leads-to* relations.

a CORAS model. In the DEXi model, a source node with an outgoing *leads-to* relation is therefore represented by an attribute with two child attributes, one representing the likelihood of the source node and one representing the conditional probability that an occurrence of the source node will lead to the target node. As illustrated in Fig. 7(c), the attribute *l_S1_to_U1* representing the likelihood contribution from *S1* to *U1* therefore has two child attributes, *l_S1* representing the likelihood of *S1* and *cp_S1_to_U1* representing the conditional probability of *S1* leading to *U1* (and similarly for *l_S2_to_U1*).

Node with attached indicators In a CORAS model, indicators can be attached to a node to show that the indicators are used as input for assessing the likelihood of the node. In the DEXi model, indicators attached to a node are therefore represented as basic attributes under the attribute representing the node. Figure 7(d) shows the complete DEXi tree structure derived from the CORAS risk model in Fig. 4. The basic attributes in Fig. 7(d) correspond to the indicators in Fig. 4. Here we see that *l_S1* has a child attribute for each of the indicators *IN-34*, *IN-41*, *IN-42*, *IN-51* and *IN-52* attached to *S1* in Fig. 4, while *l_S2* has child attributes representing the indicators *IN-41* and *IN-42*, which are attached to *S2* in Fig. 4.

Notice that we may have cases where a node has incoming *leads-to* relations in addition to attached indicators, although this is not the case in the example. In such cases, the attribute representing the node can have child attributes representing the incoming branches in addition to the child attributes representing indicators.

***Leads-to* relation with attached indicators** In a CORAS model, indicators can be attached to a *leads-to* relation from one node to another, or on a vulnerability attached to such a relation, to show that the indicators are used as input for assessing the conditional probability of an occurrence of the source node leading to the target node. In the DEXi model, indicators attached to a *leads-to* relation (or vulnerability) are therefore represented by basic attributes under the attribute representing the conditional probability assigned to the relation. Therefore, in Fig. 7(d) we see that *cp_S1_to_U1* has a child attribute for each of the indicators *IN-30*, *IN-34* and *IN-35*, while *cp_S2_to_U1* has one child attribute representing *IN-52*.

Other CORAS model fragments We have not provided separate guidelines for threats, *initiates* relations, and indicators attached to *impacts* relations. For the latter, the reason is that a CORAS model does not provide any support for consequence assessment beyond the assignment of a consequence value to the *impacts* relation from an unwanted incident to an asset. All indicators relevant for consequence assessments are therefore represented as basic attributes directly under the attribute representing the consequence, as illustrated by *c_U1_A1* in Fig. 7(d). In our example, the single indicator *IN-C1* attached to the *impacts*

relation from *U1* actually provides the consequence value directly, which means that the *IN-C1* attribute could have been attached directly under *R1*, without the intermediate *c_U1_A1* attribute. We chose to include *c_U1_A1* to illustrate the general structure.

Concerning threats and *initiates* relations, we rarely assign likelihoods to these CORAS elements in practice, as estimating threat behavior is very difficult. Instead, we assign a likelihood directly to the target node of the *initiates* relation. An indicator assigned to a threat or to an *initiates* relation can therefore be handled as if it was assigned directly to the target node.

5.3 Step 2.2: Validate Assessment Algorithm

Before putting the algorithm in operation, it should be validated to verify that its output can reasonably be expected to reflect reality. When dealing with the kind of cyber-risk assessment addressed in this paper, we typically need to rely on expert judgment for this. We first select a set of validation scenarios and then validate the output from the algorithm for each scenario with a team of experts. As the CORAS model does not provide any support for consequence assessment beyond annotation on *impacts* relations, we focus here primarily on the likelihood assessment.

A validation scenario is a set of indicator values representing one possible snapshot of the dynamic factors that influence the likelihood assessment (and hence also the risk level). Thus, the number of possible scenarios is the product of the number of possible values for each such indicator. This often results in many possible scenarios, which may be infeasible to validate. Our example in Fig. 4 includes 7 different Boolean indicators affecting the likelihood assessment (as well as one affecting the consequence). This gives 128 possible scenarios.

We therefore need to select a reasonable number of scenarios depending on the available effort. As a minimum, we suggest selecting validation scenarios based on the following two criteria: (1) cover the extreme scenarios where none or all of the indicators are triggered (yielding the minimum and maximum frequency values), and (2) cover each path in the CORAS risk model, meaning that for each path p (from the threat to the unwanted incident) in the risk model, there must be a scenario where one or more indicators along the path is triggered and the indicators for all other paths are not triggered unless these indicators also affect path p. By triggered, we mean that the indicator value contributes to the increase of likelihood. For example, for *IN-34* (Are idle sessions destroyed?), the value False (=No) would imply a higher likelihood than True (=Yes), since idle sessions can be exploited by an attacker.

For validating the output of the algorithm with the experts, we recommend using a well-established approach, such as the Wide-band Delphi method [4]. This is a forecasting technique used to collect expert opinion in an objective way, and arrive at consensus conclusion based on that. Another similar estimation approach is the Constructive Cost Model (COCOMO) [5].

6 Related Work

Most security risk approaches aim to provide either quantitative or qualitative assessments capturing the risk level at a single point in time, rather than continuous monitoring. However, there are also approaches that address dynamic aspects and offer support for updating assessments based on new information.

Poolsappasit et al. [30] propose an approach for dynamic security risk management using Bayesian Attack Graphs (BAGs). This approach is dynamic in the sense that it allows system administrators to tweak the probability of events captured by a BAG in order to see how this propagates in the complete risk picture. While their approach facilitates manual update of probability of events, our method facilitates both manual and automatic update of the likelihood of events indirectly through the different types of indicators. The manual update in our method is based on input provided by representatives of the target under analysis (business configuration indicators), while the automatic update is based on input collected from vulnerability scanning, application-layer monitoring, and network-layer monitoring.

The first use of measurable indicators as dynamic input to provide risk level assessments based on CORAS was presented in [33]. This is a quantitative approach where indicators are represented by variables in arithmetic formulas for computing risk levels, without using distributions or BNs. As argued by Neil et al. [25] BNs provide a flexible and attractive solution to the problem of modeling (operational) risk. In particular, BNs enable an analyst to combine quantitative information (e.g. available historical data) with qualitative information (e.g. subjective judgments) regarding the loss-generating processes. In the context of cyber-risk, BNs have been used for a variety of purposes, such as to model attack graphs or loss event frequencies [30, 20].

The consequence assessment for the quantitative version of our approach is based on the Loss Distribution Approach (LDA), which is typically used to model operational risk and its insurability [22], provided that a sufficient amount of data is available. In the LDA, the temporal occurrence of the losses is frequently modeled by a Poisson process, while various families of distributions (Gamma, Generalized Pareto, Lognormal, etc.) might be used to model the severity of the losses. Biener et al. [3] study whether models which prove to be useful for operational risk can also be applied to an analysis of cyber-risk. They conclude that the LDA approach is suitable to model cyber-risk and that it provides useful insights regarding, e.g., the distinct characteristics of cyber-risk with respect to operational risk in general.

For the qualitative version of our method, we chose DEXi due to its simplicity and ease of integration in the WISER framework. DEXi is one of many approaches within the field of multi-criteria decision making (on which there is a huge literature [37]), and has been tried out in a wide range of domains, such as health care, finance, construction, cropping systems, waste treatment systems, medicine, tourism, banking, manufacturing of electric motors, and energy [9, 12]. To the best of our knowledge, DEXi has not been used for security risk assessment. However, it has been applied to assess safety risks within highway

traffic [27] and ski resorts [8]. Although they focus on safety risks, the approaches provided by Omerčević et al. [27] and Bohanec et al. [8] are similar to our approach in the sense that they use DEXi models as the underlying algorithm to compute an advice based on relevant indicators. Unlike our approach, they do not employ any dedicated risk modeling language to provide a basis for developing the DEXi models.

7 Discussion and Conclusion

The framework presented in Sect. 2 has been successfully demonstrated in three different pilot organizations, using quantitative and qualitative algorithms based on 10 different CORAS models, all developed following the method outlined in Sect. 3. Due to the structure of a CORAS model and the simple relationship to the algorithm structure, we believe that most cyber-risk practitioners who are familiar with CORAS and the chosen algorithm language (**R** or DEXi) will have little problems establishing the algorithm structure from a CORAS model.

Having established the structure of an algorithm, the remaining challenge is to fill in the details, in particular deciding the impact of the indicators. This amounts to defining the functions from indicators to the parentless nodes in the BN skeleton (in the quantitative approach) or defining the scales and utility functions for the attributes (in the qualitative approach). We provide further guidelines for this in [13] and [14], respectively.

An inherent limitation of the approach is that new and unforeseen threats, vulnerabilities and attack types can only be addressed by updating the relevant risk models and algorithms (or creating new ones) manually. The only dynamic changes automatically covered by the monitoring are those captured by changing indicator values. Periodic evaluations are therefore needed to decide whether new or updated models and algorithms are required. Of course, this limitation applies to all methods that rely on human experts for risk identification.

Although our own experiences from applying the method and framework are promising, further empirical studies are needed to evaluate them in a wider context. In particular, we hope to investigate to what degree cyber-risk practitioners outside the WISER consortium are able to establish algorithms that are sufficiently correct to provide useful decision support for those in charge of dealing with cyber-risk for an organization.

Acknowledgments. This work has been conducted as part of the WISER project (653321) funded by the European Commission within the Horizon 2020 research and innovation programme.

References

1. The ACM Computing Classification System (CCS). `https://dl.acm.org/ccs/ccs.cfm`. Accessed November 3, 2017.

2. K.A. Atkinson. *An Introduction to Numerical Analysis*. Wiley, 1989.
3. C. Biener, M. Eling, and J.H. Wirfs. Insurability of Cyber Risk An Empirical Analysis. *The Geneva Papers on Risk and Insurance Issues and Practice*, 40(1):131–158, 2015.
4. B.W. Boehm. *Software Engineering Economics*. Prentice Hall, 1981.
5. B.W. Boehm, C. Abts, A.W. Brown, S. Chulani, B.K. Clark, E. Horowitz, R. Madachy, D.J. Reifer, and B. Steece. *Software Cost Estimation with COCOMO II*. Prentice Hall, 2000.
6. M. Bohanec. DEXi: Program for Multi-Attribute Decision Making. User's Manual v 5.00 IJS DP-11897, DEXi, 2015.
7. M. Bohanec, G. Aprile, M. Costante, M. Foti, and N. Trdin. A Hierarchical Multi-Attribute Model for Bank Reputational Risk Assessment. In *DSS 2.0 – Supporting Decision Making with New Technologies*, pages 92–103. IOS Press, 2014.
8. M. Bohanec and B. Delibašić. Data-Mining and Expert Models for Predicting Injury Risk in Ski Resorts. In *Proc. 1st International Conference on Decision Support System Technology (ICDSST'15)*, pages 46–60. Springer, 2015.
9. M. Bohanec, M. Žnidaršič, V. Rajkovič, I. Bratko, and B. Zupan. DEX Methodology: Three Decades of Qualitative Multi-Attribute Modeling. *Informatica (Slovenia)*, 37(1):49–54, 2013.
10. J.C.P. Bus. Convergence of Newton-like methods for solving systems of nonlinear equations. *Numerische Mathematik*, 27(3):271–281, 1976.
11. A.S. Chernobai, S.T. Rachev, and F.J. Fabozzi. *Operational Risk: A Guide to Basel II Capital Requirements, Models, and Analysis*. Wiley, 2007.
12. DEXi: A Program for Multi-Attribute Decision Making. `http://kt.ijs.si/MarkoBohanec/dexi.html`. Accessed October 19, 2017.
13. G. Erdogan, A. Gonzalez, A. Refsdal, and F. Seehusen. A Method for Developing Algorithms for Assessing Cyber-Risk Cost. In *Proc. The 2017 IEEE International Conference on Software Quality, Reliability, & Security (QRS'17)*, pages 192–199. IEEE, 2017.
14. G. Erdogan and A. Refsdal. A Method for Developing Qualitative Security Risk Assessment Algorithms. In *Proc. 12th International Conference on Risks and Security of Internet and Systems (CRiSIS'17)*. Springer, 2017. To appear.
15. International Organization for Standardization. *ISO/IEC 27001 – Information technology – Security techniques – Information security management systems – Requirements*, 2005.
16. International Organization for Standardization. *ISO/IEC 27032 – Information technology – Security techniques – Guidelines for cybersecurity*, 2005.
17. International Organization for Standardization. *ISO 31000:2009(E), Risk management – Principles and guidelines*, 2009.
18. International Organization for Standardization. *ISO/IEC 27005:2011(E), Information technology – Security techniques – Information security risk management*, 2011.
19. S.A. Klugman, H.H. Panjer, and G.E. Willmot. *Loss Models: From Data to Decisions*. Wiley, 2012.
20. A. Le, Y. Chen, K.K. Chai, A. Vasenev, and L. Montoya. Assessing Loss Event Frequencies of Smart Grid Cyber Threats: Encoding Flexibility into FAIR Using Bayesian Network Approach. In *Proc. 1st EAI International Conference on Smart Grid Inspired Future Technologies (SmartGIFT'16)*, pages 43–51. Springer, 2017.
21. M. S. Lund, B. Solhaug, and K. Stølen. *Model-Driven Risk Analysis: The CORAS Approach*. Springer, 2011.

22. A.J. McNeil, R. Frey, and P. Embrechts. *Quantitative Risk Management: Concepts, Techniques and Tools*. Princeton University Press, 2015.
23. Common Attack Pattern Enumeration and Classification (CAPEC). `https://capec.mitre.org/`. Accessed October 18, 2017.
24. S. Mittnik and I. Starobinskaya. Modeling Dependencies in Operational Risk with Hybrid Bayesian Networks. *Methodology and Computing in Applied Probability*, 12(3):379–390, 2010.
25. M. Neil, N. Fenton, and M. Tailor. Using Bayesian Networks to Model Expected and Unexpected Operational Losses. *Risk Analysis*, 25(4):963–972, 2005.
26. Solve Systems of Nonlinear Equations. `https://cran.r-project.org/web/packages/nleqslv/nleqslv.pdf`. Accessed October 19, 2017.
27. D. Omerčević, M. Zupančič, M. Bohanec, and T. Kastelic. Intelligent response to highway traffic situations and road incidents. In *Proc. Transport Research Arena Europe 2008 (TRA'08)*, pages 21–24. TRA, 2008.
28. The Open Web Application Security Project. `www.owasp.org`. Accessed October 18, 2017.
29. OWASP Zed Attack Proxy Project. `https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project`. Accessed November 2, 2017.
30. N. Poolsappasit, R. Dewri, and I. Ray. Dynamic Security Risk Management Using Bayesian Attack Graphs. *IEEE Transactions on Dependable and Secure Computing*, 9(1):61–74, 2012.
31. A. Refsdal, G. Erdogan, G. Aprile, S. Poidomani, R. Colgiago, A. Gonzalez, A. Alvarez, S. González, C.H. Arce, P. Lombardi, and R. Mannella. D3.4 – Cyber risk modelling language and guidelines, final version. Technical Report D3.4, WISER, 2017.
32. A. Refsdal, B. Solhaug, and K. Stølen. *Cyber-Risk Management*. Springer, 2015.
33. A. Refsdal and K. Stølen. Employing Key Indicators to Provide a Dynamic Risk Picture with a Notion of Confidence. In *Proc. 3rd IFIP International Conference on Trust Management (TM'09)*, pages 215–233. Springer, 2009.
34. The R Project for Statistical Computing. `https://www.r-project.org`. Accessed October 19, 2017.
35. B. Solhaug and K. Stølen. The CORAS Language - Why it is designed the way it is. In *Proc. 11th International Conference on Structural Safety & Reliability (ICOSSAR'13)*, pages 3155–3162. Taylor and Francis, 2013.
36. A. Černivec, A. Alvarez, S. González, C. H. Arce, A. Žitnik, R. Plestenjak, and A. L. Biasibetti. D4.2 – WISER Monitoring Infrastructure. Technical Report D4.2, WISER, 2016.
37. M. Velasquez and P.T. Hester. An analysis of multi-criteria decision making methods. *International Journal of Operations Research*, 10(2):56–66, 2013.
38. Web Application Attack and Audit Framework. `http://w3af.org/`. Accessed November 2, 2017.
39. Wide-Impact cyber SEcurity Risk framework (WISER). `https://www.cyberwiser.eu/`. Accessed October 16, 2017.