Risk Assessment via Partial Orders

Marco Benini and Sabrina Sicari*
Dipartimento di Informatica e Comunicazione
Università degli Studi dell'Insubria
via Mazzini 5, IT-21100, Italy
{marco.benini,sabrina.sicari}@uninsubria.it

Abstract

Although risk assessment is a well-established engineering practice to evaluate the security of a system, the significance of the obtained results is often debated since it depends on the estimates of one or more experts. The core of the debate lies in the metrics the experts use to quantify the importance and the impacts of the system vulnerabilities. This work directly addresses this problem on experts' metrics by showing a risk assessment method that is invariant with respect to compatible metrics. This result is obtained by abstracting over the individual values and, thus, by developing a method based only on the inner content of the experts' evaluations.

Keywords: Risk assessment, network security, network vulnerability

*corresponding author: tel. +39 0332 218924 fax. +39 0332 218909

1 Introduction

Security is an engineering process, characterised by distinct phases that have to be implemented in an appropriate order: an important phase is the risk assessment task that forms the basis to evaluate the success of the whole process. In fact, it allows to evaluate on a quantitative basis the security posture of a system, and, later, to measure the effectiveness of countermeasures.

Following Howard and Le Blanc [16] who said "You cannot build a secure system until you understand your threats", in order to improve the security of a system, a preliminary investigation of the system vulnerabilities is performed, the related threats are identified and, then, the associated risks are evaluated. Thus, risk assessment in general and risk analysis in particular, are a focal point in the definition of a security solution. In this respect, it is helpful to understand what level of risk is acceptable [17] and to find a trade-off among risks.

Despite the need of risk assessment, it is difficult to evaluate the risks in a real system, since most methodologies are based on the so-called exploitability values [16]: an exploitability value is a quantitative measure of the easiness to use a vulnerability to damage the system.

Usually, the exploitability values are assigned by experts in a subjective way: an expert uses his in-depth knowledge of a system security problem to evaluate the risks with respect to a personal metric, derived mainly from experience. Thus, the risk assessment process is exposed to criticisms, since it is ultimately based on a personal, yet authoritative, judgement [21]. Therefore, the distrust of the scientific community towards the validity of risk assessment could be traced back to two main problems:

1. the intrinsic difficulty to find a total order among the exploitabilities of very different vulnerabilities; this point can be equivalently formulated as the difficulty to compare unrelated or distant security problems by means of the same metric;

2. the difficulty to compare the risk analyses produced by different security experts; equivalently, the problem is described as how to devise a sound method to combine the risk assessments of different experts analysing the same system.

In order to cope with these problems, we have formalised and extended the risk assessment method introduced in [7, 27, 4] to show when different metrics produce equivalent results. As a matter of fact, the resulting risk assessment process clearly evidences that

1. the method does not require a metric to form a total order, thus eliminating from the root the need to compare totally unrelated vulnerabilities;

2. as far as metrics are *compatible*, the risk assessment method produces equivalent results.

The choice to extend the method in [7, 27, 4] is justified since it is based on the evaluation of the exploitability of the vulnerabilities and the dependencies among vulnerabilities in a system. Hence, the method suffers from the previously exposed problems, but, at the same time, it allows a direct mathematical formalisation that permits to overcome them.

The central notion of this formalisation and, thus, of this work, is the concept of *compatible* metric: as we will prove, when the experts use mutually compatible metrics, i.e., metrics allowing their results to be expressed one in the terms of the other, it is possible to construct a common metric such that each individual metric is embedded into the common one in a way that the evaluation each expert performs can be identically carried on in the common metric.

Therefore, after presenting in Section 2 the risk assessment method, we prove its main properties in Section 3, where the notion of *compatible* metrics is defined and discussed. The work finishes with a comparison with other approaches, in Section 4, and, in Section 5, a summary of the main results.

2 The risk assessment method

The goal of risk assessment is to determine the likelihood that the identifiable threats of a system will harm, weighting their occurrence with the damage they may cause. Thus, a risk assessment method is a procedure to define the risk of the occurrence of one or more threats; the risk evaluation, as inferred by the procedure, is *justified* by the method, whose aim is to explain the provided evaluation.

As already said in the Introduction, we adopt the risk assessment method introduced in [27]; in this Section, we illustrate the method, briefly discussing its foundations. In Section 3, we will prove its main formal properties and we will use them to derive some interesting facts about the quality of the risk evaluations the method allows to derive.

The starting point is to consider a distributed system as a composition of black-box elements communicating through directed links, where a link (c_1, c_2) means that c_1 may directly send input to c_2. Hence, the *architecture* of the system is modelled by the directed graph $\mathcal{A} = \langle C, L \rangle$ where C is the set of black-box components and L is the set of links.

Moreover, each element $x \in (C \cup L)$ is assumed to be *vulnerable*: a vulnerability is a flaw or weakness in a system's or component's design, implementation or management that could be exploited to violate the system's security policy. The definition is contained in [26] where we also find:

> "Most systems have vulnerabilities of some sort, but this does not mean that the systems are too flawed to use. Not every threat results in an attack, and not every attack succeeds. Success depends on the degree of vulnerability, the strength of attacks, and the effectiveness of any countermeasures in use. If the attacks needed to exploit a vulnerability are very difficult to carry out, then the vulnerability may be tolerable. If the perceived benefit to an attacker is small, then even an easily exploited vulnerability may be tolerable.

> However, if the attacks are well understood and easily made, and if the vulnerable system is employed by a wide range of users, then it is likely that there will be enough benefit for someone to make an attack."

Therefore, the vulnerabilities are organised in a structure showing how they can be used to perform an attack. The adopted formalism is the one of *attack trees* [18, 23], a well-known method to describe the attacks as goals to threaten a system: the attacks are represented in a tree structure, with the main goal as the root node and the different ways of achieving as its children. In turn, each internal node in the tree represent an intermediate goal to attain the root goal. There are `and` nodes and `or` nodes, each one representing an immediate sub-goal of the father node: `or` nodes are alternative sub-goals, i.e., different ways to satisfy the father goal; `and` nodes represent a set of steps toward achieving the father goal; the leaves of the tree represent the system vulnerabilities.

Although the attack tree representation is a satisfactory model of an attack plan and, thus, it can be thought as the description of the security status of a system with respect to an attack vector, it does not contain the whole wealth of information that can be used to evaluate the risk associated to its root goal.

In fact, the vulnerabilities may depend one on another, but this information is partially lost in the attack tree representation. In fact, only the structural dependencies are made explicit, i.e., when the attack requires the exploitation of one or more sets of vulnerabilities, while indirect dependencies, i.e., when a vulnerability might ease an attack, even if the attack is possible without its exploitation, are neglected.

Therefore, the method takes into account also indirect dependencies among vulnerabilities and it adopts an analytical approach to combine the risk assessment of the single vulnerabilities with the attack trees where they appears in, considering also their mutual dependencies.

2.1 Measuring risk

In general, the risk is measured by a function r of two variables: the damage potential of the hazard and its level of exploitability. The damage potential is defined as the average loss an attack may cause, where the loss may be any suitable quantity.

In addition, more specifically, the term *exploitability* refers to a value that measures both the easiness, the feasibility and the reproducibility of an attack, as defined in the STRIDE/DREAD theory [16].

The method evaluates the total risk of a possible threat following the subsequent steps:

1. The threat to the system under examination is modelled by using an attack tree: the attack goal is the root node and the children nodes represent different ways of achieving it. Recursively, children can be alternative sub-goal, each one satisfying the goal (`or` sub-trees) or partial sub-goals, whose composition satisfies the goal (`and` sub-trees).

To each vulnerability v is associated an index $E_0(v)$, called its *initial exploitability*, which measures how easy is to exploit v to perform a successful attack, supposing to have the total control of the link or the component in the given architecture.

2. The dependencies among identified vulnerabilities are introduced: a vulnerability A depends on a vulnerability B if and only if when B is already exploited, then A becomes easier to exploit. As already said, we do not limit the analysis to structural dependencies, so dependencies should be analysed by taking into account contextual, architectural and topological information.

   Moreover, each dependency is weighted by an exploitability value, using the same metric as E_0; the meaning of this value is to measure how easy is to use the identified dependency to violate the target vulnerability, assuming that the source vulnerability has been exploited. This exploitability is called *conditional exploitability*.

3. The exploitability of each single vulnerability v is calculated taking into account its initial value $E_0(v)$ and its dependencies, according to the algorithm described in Section 2.2.

4. The risk associated to the threat under examination is finally computed by recursively aggregating exploitabilities along the attack tree.

   Specifically, the exploitability of an `or` sub-tree is the easiest (maximal) value of its children, and the exploitability of an `and` sub-tree is the most difficult (minimal) value of its children. The aggregated exploitability measures the level of feasibility of the attack and is combined with the damage potential to assess the risk of the threat.

The metric employed in the evaluation of exploitabilities and their dependencies is the set of possible values for $E_0(v)$. We require this set to be a partial order[1]: this choice reflects the difficulty to compare an arbitrary pair of vulnerabilities in order to decide their relative difficulty; usually, similar vulnerabilities are easily compared, while different vulnerabilities may be compared only to some extent, e.g., saying that both are easier or more difficult to exploit than a third one.

Evidently, it is safe to assume that the partial order contains a finite number of elements, since the system vulnerabilities are always finite, and, moreover, we assume that the partial order contains a global maximum, denoted as 1, and a global minimum, denoted as 0. This assumption is justified since every actual vulnerability is easier to exploit than the ideal perfectly secure component, while each vulnerability is harder to violate than the ideal perfectly insecure component.

[1] In this sense, our definition of metric differs from the standard one which requires the values to be numbers of some sort. Nevertheless, our definition strictly adheres to the intended meaning: a metric is mathematical structure used to measure something. In fact, outside the domain of risk assessment, there are many examples of metrics which cannot be reduced to a set of numbers but, still, they possess a structure that enables calculation and comparison.

2.2 Exploitability of dependent vulnerabilities

As already said, the system is described as a graph $\mathcal{A} = \langle C, L \rangle$ where C is the set of *components* and L is the set of *links* between components.

The components and links are exposed to the set of vulnerabilities V where an element $(u, v) \in V_C$ means respectively that the component, or link, u is susceptible to be subverted thanks to the flaw v.

Initially, during Step 1 of the method application, an expert assesses how easy and repeatable is to exploit every single vulnerability to gain the control of a component or a link in the given architecture. We call this value the *initial exploitability* $E_0(v)$ of the vulnerability v in the system $\mathcal{A}$.

The functions $E_i \colon V \mapsto \mathcal{O}$ map vulnerabilities to $\mathcal{O}$, a partial ordered set of degrees of exploitability. The functions are indexed by a step number (details later), thus the E_0 function generates the exploitability values $E_0(v)$, as expected.

The $\mathcal{O}$ set, modelling the expert's metric, is a finite, partially ordered set containing two distinct elements, 0, its minimum, and 1, its maximum.

Two comparable elements $a < b$ in the order represent two exploitability values potentially associated with two vulnerabilities v_a and v_b, modelling the fact that it is easier to exploit the v_b vulnerability than v_a. As a matter of fact, not every pair of vulnerabilities can be directly compared to each other, thus the partial ordering relation among exploitability values.

As already remarked, the architecture of the system imposes dependencies among vulnerabilities. For example, we need to understand if it is easier to exploit a vulnerability of a component given that an input link attached to it has already been compromised or a component attached to any of its input links has already been compromised. Hence, we denote with $E(v|w)$ the (conditional) exploitability of v given that the vulnerability w has already been violated. The value of $E(v|w)$ is assessed during phase 2 of the risk assessment procedure[2].

The dependencies among vulnerabilities are represented in the *dependency graph* $\mathcal{D} = \langle V, D \rangle$, whose nodes are the vulnerabilities and the edge (w, v) is in D iff $E(v|w) \geq E_0(v)$, i.e., an edge (w, v) means that it is easier to compromise an element suffering the v vulnerability when one has already compromised an element affected by the w vulnerability.

The formalisation shows that the number of exploitability evaluations is bounded since the number of edges in the $\mathcal{D}$ graph is, at most, $|V|(|V| - 1)$. However, in practice, most of the vulnerabilities are independent, and the evaluations the expert has to guess is typically closer to $|V|$ than to $|V|^2$.

Initially, each node v in the dependency graph $\mathcal{D}$ is labelled with the value $E_0(v)$, that is, its initial measure of how difficult is to exploit the vulnerability. Similarly, the conditional exploitabilities are used to label the edges they belong to. The initial assessment depicted in the graph $\mathcal{D}$ does not take into account

[2] The initial assessments of $E_0(v)$ and $E(v|w)$ are performed by an expert according to his experience. Since these assessments require both experience and ingenuity, it is legitimate to ask whether the expert is *trustable*. However, thus important question is outside the scope of this article, where an expert is always assumed to produce reliable assessments.

that each vulnerability could be exploited thanks to the previous exploitation of one of the vulnerabilities on which it depends.

Therefore, the labels of the nodes should be iteratively updated by considering the easiest way, i.e., the maximum value, to exploit an incoming vulnerability in the dependencies graph. In turn, each incoming vulnerability could be exploited by controlling the affected element or leveraging on the dependency itself: the most difficult, i.e., the minimum, constraints the value.

Furthermore, from a purely mathematical point of view, the notions of maximum and minimum are not available, since the metric is a partial order. The equivalent formal notions, well defined on partial orders, is the one of sup (supremum) and inf (infimum): in particular, $\sup(a, b)$ is the least element c in the order such that $a \leq c$ and $b \leq c$; dually, $\inf(a, b)$ is the greatest element c in the order such that $a \geq c$ and $b \geq c$. It is evident that, in the case of a total order, sup = max and inf = min. Moreover, the sup and inf operators are always defined because of the existence of a global maximum 1 and of a global minimum 0 in the order.

Therefore, the update rule for labels is defined by the following formula:

$$E_{i+1}(v) = \sup(\{E_i(v)\} \cup \{\inf\{E(v|w), E_i(w)\} \colon (w, v) \in D\}) \ . \tag{1}$$

Thus, the third step of the method consists in iteratively applying (1) for each vulnerability, until the system converges to stability after a suitable number n of steps.

Then, the values of $E_n(v)$ represent the final exploitability of each vulnerability v considering also its dependencies, allowing to perform the fourth step in the risk assessment procedure where the risk is finally calculated.

3 The role of orderings

As the reader may expect, the choice of the metric $\mathcal{O}$ influences the results obtained in the application of the method. On a more subtle level, the mathematical characters of the method depend on the *structure*[3] of the ordering $\mathcal{O}$, as we will prove in the following.

The first property of the method is *convergence*; we want to prove that the method reaches a fixed point in the computation of the exploitability values. A side effect of the proof will be that the algorithm terminates after a number of steps bounded by $|\mathcal{O}||V|$.

Theorem 3.1 *Given a dependency graph $\mathcal{D} = \langle V, D \rangle$, there is a number k such that, for every $v \in V$, $E_{k+1}(v) = E_k(v)$.*

proof: We notice that, for any number i and for any $v \in V$, $E_{i+1}(v) \geq E_i(v)$, since, by definition,

$$\begin{aligned} E_{i+1}(v) &= \sup(\{E_i(v)\} \cup \{\inf\{E(v|w), E_i(w)\} \colon (w, v) \in D\}) \\ &\geq \sup\{E_i(v)\} = E_i(v) \ . \end{aligned}$$

[3] The precise notion of structure we are referring to, will be clear after Theorem 3.2.

We know that the exploitability values form a finite partial order $\mathcal{O}$: let n be the number of elements in $\mathcal{O}$ and let $k = n|V|$.

By contradiction, let us suppose that, for every number i, there is a $v \in D$ such that $E_{i+1}(v) > E_i(v)$. Then, for every $v \in V$, $E_k(v) = 1$ by the pigeon hole principle. In fact, at every step we are forced to increment an element w; thus, after n steps, not necessarily consecutive, the element w reaches the maximum value. Since there are $|V|$ elements, after k steps every element reaches the maximum. But, for every $v \in V$, $E_{k+1}(v) = E_k(v)$, since no element can be incremented beyond the maximum. Therefore, we get a contradiction and the assertion is proved. □

It is important to remark that Theorem 3.1 provides an effective bound to the number of iterations: as obvious by the use of the pigeon hole principle, the given bound is unnecessarily large, and in practice[4], convergence can be obtained in a few steps, usually close to $|V|$.

Moreover, as proved in the following theorem, the method depends only on the *structure* of the order of exploitability values; intuitively, the structure of an order is the way its values are arranged by the $\leq$ relation. The precise mathematical formulation is as follows:

Theorem 3.2 *Given a dependency graph $\mathcal{D} = \langle V, D\rangle$ and two finite partial orders with maximum and minimum, $\mathcal{O}_a = \langle O_a, \leq_a, 0_a, 1_a\rangle$ and $\mathcal{O}_b = \langle O_b, \leq_b, 0_b, 1_b\rangle$, if $g\colon \mathcal{O}_a \to \mathcal{O}_b$ is a morphism[5] from $\mathcal{O}_a$ to $\mathcal{O}_b$ such that $g(E_0^a(v)) = E_0^b(v)$ for every $v \in V$ and $g(E^a(v|w)) = E^b(v|w)$ for every $(w, v) \in D$, then, for any $v \in V$ and for any i, $g(E_i^a(v)) = E_i^b(v)$, where E^a and E^b are the exploitability functions using, respectively, $\mathcal{O}_a$ and $\mathcal{O}_b$ as metrics.*

proof: A standard result, see, e.g., [10], in the theory of lattices is that to every order $\mathcal{O} = \langle O, \leq, 0, 1\rangle$ it is possible to associate a lattice $\mathcal{L} = \langle O, \sup, \inf, 0, 1\rangle$ such that $\mathcal{L}$ is a *presentation* of $\mathcal{O}$. The meaning of this result is that, for any morphism f between two orders, there is a corresponding morphism f' between the associated lattices, and, moreover, for every element x, $f(x) = f'(x)$, i.e., the order morphism and the lattice morphism are functionally equivalent.

In particular, calling $\mathcal{L}_a$ and $\mathcal{L}_b$ the lattices associated to the orders $\mathcal{O}_a$ and $\mathcal{O}_b$, respectively, there is lattice morphism $g'\colon \mathcal{L}_a \to \mathcal{L}_b$ such that $g'(x) = g(x)$ for every $x \in O_a$.

We prove by induction on i, the number of steps, that for any $v \in V$, $g'(E_i^a(v)) = E_i^b(v)$:

- base step: by hypothesis, for every $v \in V$, $g'(E_0^a(v)) = g(E_0^a(v)) = E_0^b(v)$;
- induction step: supposing that, for every $x \in V$, $g'(E_i^a(x)) = E_i^b(x)$, we prove that, for any $v \in V$, $g'(E_{i+1}^a(v)) = E_{i+1}^b(v)$.

[4] In fact, a tighter bound justifying the empirical convergence speed, can be derived considering the *diameter* of the dependency graph. Such a proof is more involved and not very significant, since the obtained bound does not improve in the worst cases what shown.

[5] A morphism is a structure-preserving function. In particular, a function f between $\mathcal{O}_a$ and $\mathcal{O}_b$ is a morphism iff for every $x, y \in O_a$ such that $x \leq_a y$, it holds that $f(x) \leq_b f(y)$.

Since $E^a_{i+1}(v)$ is defined using the lattice operators $\sup_a$ and $\inf_a$, it is convenient to consider the value of $g'(E^a_{i+1}(v))$, remembering that

$$g(E^a_{i+1}(v)) = g'(E^a_{i+1}(v)) \ .$$

By definition,

$$\begin{aligned} g'(E^a_{i+1}(v)) = g'(&\sup{}_a(\{E^a_i(v)\} \cup \\ &\cup \{\inf{}_a\{E^a(v|w), E^a_i(w)\} \colon (w,v) \in D\})) \ . \end{aligned}$$

But g' is a lattice morphism, thus it preserves the lattice operations, then

$$\begin{aligned} g'(E^a_{i+1}(v)) = \sup{}_b(&\{g'(E^a_i(v))\} \cup \\ &\cup \{\inf{}_b\{g'(E^a(v|w)), g'(E^a_i(w))\} \colon (w,v) \in D\}) \ . \end{aligned}$$

Applying the induction hypothesis and using the fact that the functions g and g' are functionally equivalent,

$$\begin{aligned} g'(E^a_{i+1}(v)) = \sup{}_b(&\{E^b_i(v)\} \cup \\ &\cup \{\inf{}_b\{g'(E^a(v|w)), E^b_i(w)\} \colon (w,v) \in D\}) \ . \end{aligned}$$

Finally, applying the hypothesis

$$g'(E^a(v|w)) = g(E^a(v|w)) = E^b(v|w) \ ,$$

we get

$$\begin{aligned} g'(E^a_{i+1}(v)) = \sup{}_b(&\{E^b_i(v)\} \cup \\ &\cup \{\inf{}_b\{E^b(v|w), E^b_i(w)\} \colon (w,v) \in D\}) \ , \end{aligned}$$

thus, by definition, $g'(E^a_{i+1}(v)) = E^b_{i+1}(v)$.

Therefore, by induction and by functional equality of g and g', for every $v \in V$ and for every i, $g(E^a_i(v)) = E^b_i(v)$. □

Some comments are due:

- The hypothesis "$g(E^a_0(v)) = E^b_0(v)$ for every $v \in V$ and $g(E^a(v|w)) = E^b(v|w)$ for every $(w,v) \in D$" is the formal way to code the fact that the initial exploitability values in both orders are used to label the dependency graph in the same way. Therefore, this hypothesis says that the initial situation to which the method is applied, is the same, modulo the g morphism.
- The g is a morphism, i.e., a function respecting the relation and the constants of the order. The meaning of this requirement is that the metrics are compatible, that is, any comparable pair of values in the first metric is associated with a pair of values in the same order in the second metric.

- The proof does not require g to be an isomorphism, i.e., to be an invertible function: in the light of the previous remark, two compatible metrics may differ in the number of values, but they must agree in the relative order of related elements. For example, the order $0 \leq a \leq b \leq 1$ and the order $0 \leq 1/2 \leq 1$ are compatible but not isomorphic since the function $g(0) = 0$, $g(1) = 1$, $g(a) = g(b) = 1/2$ is a morphism but no invertible function can be defined on these structures.

An important side effect of Theorem 3.2 is the notion of *compatible* metrics: the metric $\mathcal{O}_a$ is compatible with the metric $\mathcal{O}_b$ if there is a morphism of the form $g\colon \mathcal{O}_a \to \mathcal{O}_b$.

The notion of compatibility reveals an hidden aspect in Theorem 3.2. In fact, the result can be rephrased as: given a metric $\mathcal{O}_b$ and a compatible metric $\mathcal{O}_a$, if the initial evaluation in the $\mathcal{O}_a$ metric of the dependency graph $\mathcal{D}_a$ is equivalent to the initial evaluation of the dependency graph $\mathcal{D}_b$ in the other metric, then the final results of the risk assessment procedure are equivalent.

Henceforth, the idea behind the result in Theorem 3.2 is that the evaluation performed on a dependency graph $\mathcal{D}_a$, can be replicated on any other graph $\mathcal{D}_b$ differing only on the metric, as far as the a metric is compatible with the b metric.

Consequently, a natural application of Theorem 3.2 is to consider the evaluations of two experts using compatible metrics: in fact, since the evaluation of one expert can be mapped in the same metric as the evaluation of the second expert, the two evaluations become comparable, being expressed in the same metric, i.e., in the same set of reference values.

3.1 Composing compatible metrics

Let us suppose to have two experts, Alice and Bob, using respectively the metrics $\mathcal{O}_a$ and $\mathcal{O}_b$. Moreover, let us suppose that $\mathcal{O}_a$ is compatible with $\mathcal{O}_b$ and vice versa. Theorem 3.2 establishes that the risk evaluation of Alice can be translated into Bob's metric allowing the comparison of the two risk evaluations. Of course, also Bob's evaluation can be expressed into Alice's metric.

Being mutually compatible, a natural question is if there is a metric $\mathcal{O}_c$, the *composition* of $\mathcal{O}_a$ and $\mathcal{O}_b$, such that $\mathcal{O}_c$ extends both $\mathcal{O}_a$ and $\mathcal{O}_b$ and that preserves the risk evaluations developed in these metrics. The aim of this Section is to provide a positive answer to such question.

Therefore, we will deduce that, as far as experts' metrics are mutually compatible, it is possible to develop a common metric where each expert can conduct his own risk analysis.

Definition 3.1 *Let $\mathcal{O}_a$ and $\mathcal{O}_b$ be two metrics and let $f\colon \mathcal{O}_a \to \mathcal{O}_b$ and $g\colon \mathcal{O}_b \to \mathcal{O}_a$ be two morphisms between them. The binary relation $\leq_c$ on $\mathcal{O}_a \sqcup \mathcal{O}_b$, the disjoint union of the two metrics, is defined as*

- *if $x, y \in \mathcal{O}_a$ and $x \leq_a y$ then $x \leq_c y$;*

- *if $x, y \in \mathcal{O}_b$ and $x \leq_b y$ then $x \leq_c y$;*
- *if $x \in \mathcal{O}_a$ and $y \in \mathcal{O}_b$ and $f(x) \leq_b y$ then $x \leq_c y$;*
- *if $x \in \mathcal{O}_a$ and $y \in \mathcal{O}_b$ and $x \leq_a g(y)$ then $x \leq_c y$;*
- *otherwise $x \not\leq_c y$.*

The $\leq_c$ relation is reflexive, since $x \in \mathcal{O}_a \sqcup \mathcal{O}_b$ means either $x \in \mathcal{O}_a$ or $x \in \mathcal{O}_b$ and, thus, $x \leq_a x$ or $x \leq_b x$, respectively, being $\leq_a$ and $\leq_b$ ordering relations. Hence, by definition, $x \leq_c x$ for every x.

Moreover, the $\leq_c$ relation is anti-symmetric, i.e., if $x \leq_c y$ and $y \leq_c x$ then $x = y$: in fact, if $x \in \mathcal{O}_a$ it follows that $y \in \mathcal{O}_a$, since, looking at $y \leq_c x$, the only clause in the definition of $\leq_c$ where the second argument of $\leq_c$ is in $\mathcal{O}_a$ is the first one, hence, $x \leq_a y$ and $y \leq_a x$ thus, $x = y$, being $\leq_a$ an ordering relation. Similarly, if $x \in \mathcal{O}_b$ then $y \in \mathcal{O}_b$ because only the second clause in the definition of $\leq_c$ applies to $x \leq_c y$, hence, $x \leq_b y$ and $y \leq_b x$, implying $x = y$ as before.

Finally, $\leq_c$ is transitive, i.e., if $x \leq_c y$ and $y \leq_c z$ then $x \leq_c z$: the proof goes by cases on the definition of $x \leq_c y$

- if $x \leq_a y$ then $x, y \in \mathcal{O}_a$, so we reason be cases on the definition of $y \leq_c z$
  - if $y \leq_a z$ then $x \leq_a z$ being $\leq_a$ an ordering relation, thus $x \leq_c z$;
  - if $f(y) \leq_b z$ and $z \in \mathcal{O}_b$ then $f(x) \leq_b f(y)$ being f a morphism, thus $f(x) \leq_b z$ and, by definition, $x \leq_c z$;
  - if $y \leq_a g(z)$ and $z \in \mathcal{O}_b$ then $x \leq_a g(z)$, thus $x \leq_c z$;
- if $x \leq_b y$ then $x, y \in \mathcal{O}_b$ hence $y \leq_c z$ iff $z \in \mathcal{O}_b$ and $y \leq_b z$, thus $x \leq_b z$, that is, $x \leq_c z$;
- if $f(x) \leq_b y$ then $x \in \mathcal{O}_a$ and $y \in \mathcal{O}_b$ hence $y \leq_c z$ iff $z \in \mathcal{O}_b$ and $y \leq_b z$, thus $f(x) \leq_b z$, that is $x \leq_c z$;
- if $x \leq_a g(y)$ then $x \in \mathcal{O}_a$ and $y \in \mathcal{O}_b$ hence $y \leq_c z$ iff $z \in \mathcal{O}_b$ and $y \leq_b z$, but $g(y) \leq_a g(z)$ being g a morphism, thus $x \leq_a g(z)$, that is $x \leq_c z$.

Therefore, $\leq_c$ is an ordering relation over $\mathcal{O}_a \sqcup \mathcal{O}_b$. The minimum of $\leq_c$ is 0_a while its maximum is 1_b: in fact, $0_a \leq_c x$ for every $x \in \mathcal{O}_a$ and $0_a \leq_c 0_b$ since $0_a \leq_a g(0_b) = 0_a$ being g a morphism, hence $0_a \leq_c x$ for every $x \in \mathcal{O}_b$ since $0_b \leq_c x$. The maximality of 1_b is established in an analogous way.

Definition 3.2 *We call $\mathcal{O}_c = \langle \mathcal{O}_a \sqcup \mathcal{O}_b; \leq_c \rangle$ the* composition metric *of $\mathcal{O}_a$ and $\mathcal{O}_b$, two mutually compatible metrics.*

It is evident that $\mathcal{O}_c$ is a proper metric. Moreover, if $x, y \in \mathcal{O}_a$ and $x \leq_c y$ then $x \leq_a y$; the same holds for $\mathcal{O}_b$, so the two metrics $\mathcal{O}_a$ and $\mathcal{O}_b$ are properly contained in $\mathcal{O}_c$ which extends both of them.

Finally, the infimum $\inf_c(S)$ and the supremum $\sup_c(S)$ in $\mathcal{O}_c$ of a set S of values in $\mathcal{O}_a$ coincides with $\inf_a(S)$ and $\sup_a(S)$, respectively. In fact, calling $m = \inf_c(S)$, m is such that, for every $x \in S$, $m \leq_c x$ and, for each y such that $y \leq_c x$ for every $x \in S$, $y \leq_c m$. But, by the definition of $\leq_c$, if $m \leq_c x$ for some $x \in \mathcal{O}_a$, then $m \in \mathcal{O}_a$ as well, so, by minimality of $\inf_a(S)$, m coincides with $\inf_a(S)$. An similar argument holds for the infimum in $\mathcal{O}_b$ and the supremum in $\mathcal{O}_a$ and $\mathcal{O}_b$.

Therefore, our experts, Alice and Bob may work in their metrics $\mathcal{O}_a$ and $\mathcal{O}_b$, but their evaluations can be immediately compared considering them as being developed in the $\mathcal{O}_c$ metric, since, in a proper mathematical language, $\mathcal{O}_a$ and $\mathcal{O}_b$ are sub-metrics of $\mathcal{O}_c$, which is closed under the risk evaluation procedure.

3.2 An illustrating example

Although the properties of our method have been proved beyond any doubt, some relevant aspects of their practical application may not be immediately clear from the purely mathematical presentation. Therefore, in this Section we want to illustrate an abstract example that may help to clarify the scope of our results as well as some consequences of their application. It should be remarked that the example has been designed to reveal the hidden aspects of our approach in an application: hence, the example is fictitious to meet the goal to have an understandable dimension and to clearly show the peculiarities of our approach.

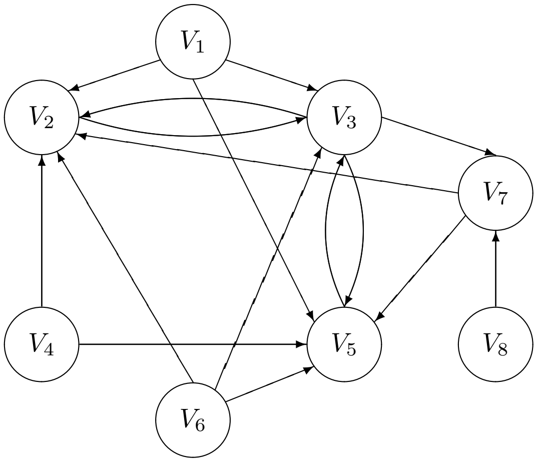

Figure 1: The dependency graph of the example system

The scenario is as follows: we have two security experts, Alice and Bob, working together to evaluate the risk of a network attack to a complex system. They developed a suitable attack tree (not shown) and they agree on both the set of vulnerabilities affecting the system, and on the way they depend one on each other. Hence, our experts produce the system dependency graph, shown in Fig. 1, whose nodes are the identified vulnerabilities and whose arcs are the dependencies.

In practice, the depicted scenario is common: the possible ways to conduct an attack, the identification of the vulnerabilities and, finally, the dependencies among the identified vulnerabilities are subjects on which experts can easily integrate their knowledge, thus producing a common, agreed picture of the security status of a system. From a different perspective, since the outcome of this phase of a security analysis is of a qualitative kind, the experts tend to accept a common view, where their contributions are fused in an integrated picture.

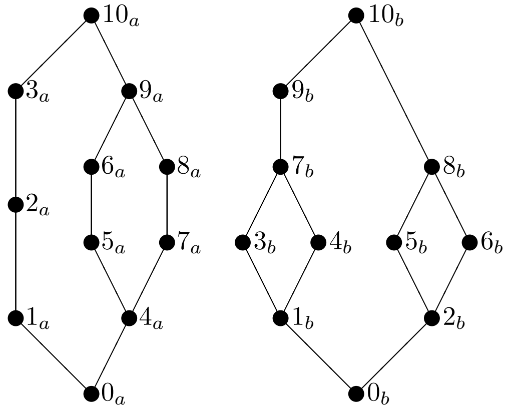

Figure 2: The metrics a and b

Differently, when the experts are asked to quantify the risks connected to the identified vulnerabilities, their evaluations may diverge because of the application of different metrics coming from their different training, attitude and experience. In our example, Alice adopts the metric a while Bob uses the metric b; both of them are represented in Fig. 2. The drawing shows the minima (0_a and 0_b) at the bottom, the maxima (10_a and 10_b) at the top, and a value x is less than y if x is below y and connected to. The supremum of two elements x and y is the minimal point above x and y, connected to both of them, and, dually, the infimum of x and y is the closest connected point below them.

In the scenario, Alice develops an initial evaluation of the exploitability values, synthesised in Fig. 3; Bob does the same, as illustrated in Fig. 4. These evaluations are the result of the application of the experts' experience and judgement, thus, at least to some extent, the values are subjective.

Applying our method, Alice and Bob can calculate the final risk assessment, considering also the role of dependencies: after a few iterations of the application

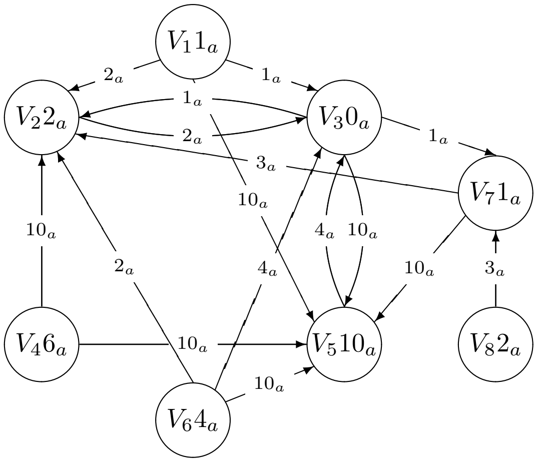

Figure 3: The initial evaluation of Alice

of (1), Alice derives the following risk vector

$E^a(V_1)$	$E^a(V_2)$	$E^a(V_3)$	$E^a(V_4)$
1_a	10_a	10_a	6_a

$E^a(V_5)$	$E^a(V_6)$	$E^a(V_7)$	$E^a(V_8)$
10_a	4_a	2_a	2_a

,

while Bob obtains as his final result

$E^b(V_1)$	$E^b(V_2)$	$E^b(V_3)$	$E^b(V_4)$
1_b	10_b	10_b	5_b

$E^b(V_5)$	$E^b(V_6)$	$E^b(V_7)$	$E^b(V_8)$
10_b	2_b	3_b	3_b

.

It is evident that the derived evaluations are different. Nevertheless, following Theorem 3.2, if the metrics are compatible and the initial assessments are equivalent, then the results should coincide, modulo a suitable *renaming* of the values in the metrics.

The *renaming* function is the morphism relating the metrics of our experts, and its existence is the criterion to say that the metrics are compatible. For example, the metric b used by Bob can be mapped in the metric a of Alice via the morphism g shown in Fig. 5. It should be immediate from the graphical representation that g is, indeed, a morphism, i.e., a function preserving orders: if $x < y$ in the metric b, then $g(x) < g(y)$ in the metric a.

In fact, transforming the resulting risk vector of Bob by means of the g

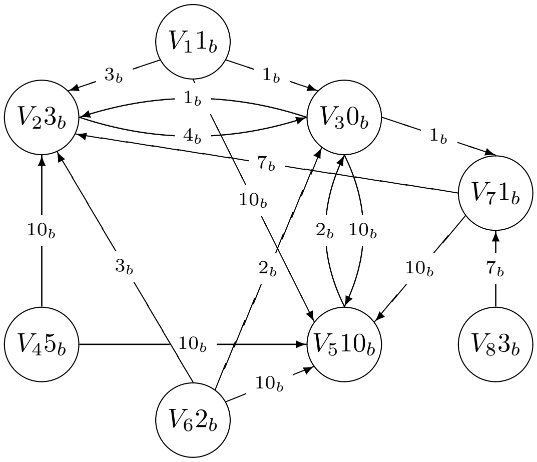

Figure 4: The initial evaluation of Bob

morphism, we get

$g(E^b(V_1))$	$g(E^b(V_2))$	$g(E^b(V_3))$	$g(E^b(V_4))$	$g(E^b(V_5))$	$g(E^b(V_6))$	$g(E^b(V_7))$	$g(E^b(V_8))$
1_a	10_a	10_a	6_a	10_a	4_a	2_a	2_a

,

that is exactly the result of Alice.

If the initial evaluations differ, the result of Theorem 3.2 allows us to compare the final risk analyses: in fact, if the initial assessment of Alice is the one depicted in Fig. 6, then, after the propagation of dependencies, her final assessment is:

$E^a(V_1)$	$E^a(V_2)$	$E^a(V_3)$	$E^a(V_4)$	$E^a(V_5)$	$E^a(V_6)$	$E^a(V_7)$	$E^a(V_8)$
5_a	10_a	10_a	6_a	10_a	4_a	2_a	2_a

.

Therefore, Alice and Bob's results can be compared by considering Bob's assessment expressed in Alice's metric:

$g(E^b(V_1)) \mapsto E^a(V_1)$	$g(E^b(V_2)) \mapsto E^a(V_2)$	$g(E^b(V_3)) \mapsto E^a(V_3)$	$g(E^b(V_4)) \mapsto E^a(V_4)$	$g(E^b(V_5)) \mapsto E^a(V_5)$	$g(E^b(V_6)) \mapsto E^a(V_6)$	$g(E^b(V_7)) \mapsto E^a(V_7)$	$g(E^b(V_8)) \mapsto E^a(V_8)$
$1_a \mapsto 5_a$	$10_a \mapsto 10_a$	$10_a \mapsto 10_a$	$6_a \mapsto 6_a$	$10_a \mapsto 10_a$	$4_a \mapsto 4_a$	$2_a \mapsto 2_a$	$2_a \mapsto 2_a$

.

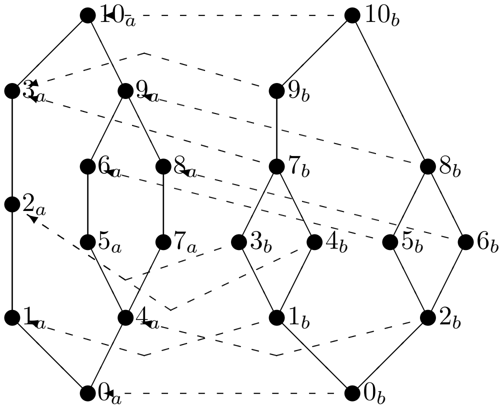

Figure 5: The morphism from the metric b to a

It is easy to show that Alice's metric is compatible with Bob's by means of a suitable morphism, as shown in Fig 7.

Hence, it is possible to construct the composition metric c as described in Section 3.1: the reader is invited to check that the resulting metric c is the one shown in Fig 8.

The embedding of the metrics a and b is made evident in Fig 9 to clarify how the composition metric is constructed starting from the two compatible metrics.

4 Related works

Even though the application of risk evaluation methodologies has been widely discussed and analysed, see, e.g., [11, 1, 17, 28], among information security experts there appears to be no agreement regarding the best or the most appropriate method to assess the probability of computer incidents [24].

In literature there are many attempts to face the risk assessment problem; some of them define systematic approaches while others provide more ad-hoc methods to evaluate the likelihood of (a class of) violations.

In particular, we have found of interest Baskerville's description [3] of the evolution of various ad-hoc methods to measure risk that sometimes could be combined to improve the accuracy of the security evaluation.

On the side of systematic approaches, S. Evans et al. [13] present a system security engineering method to discover system vulnerabilities, and to determine what countermeasures are best suited to deal with them: the paradigm of this work is *analysing information systems through an adversary's eyes*.

Differently, [22] provides a probabilistic quantitative model that measures security risk. It is also possible to calculate risk starting from hybrid values both quantitative and qualitative ones.

With respect to the previous works, our approach, starting from its initial definition in [27], has been based on the structured evaluation of single vul-

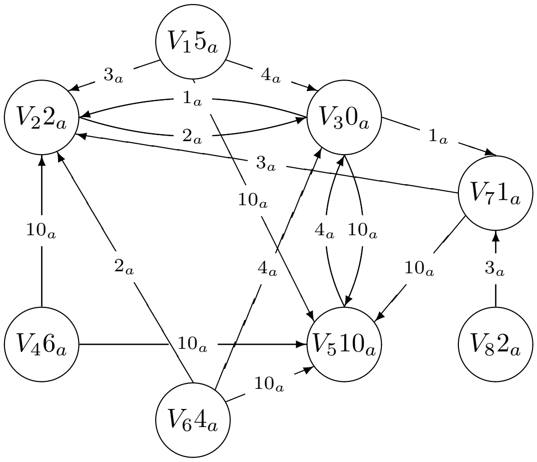

Figure 6: A different initial evaluation of Alice

nerabilities along with their mutual dependencies. In this respect, the results in [13] are similar to ours, although they do not propose a formal method based on mathematical arguments. In fact, the distinctive aspect of our work with respect to the discussed ones is the mathematical formalisation of the risk assessment method in order to derive its characterising properties.

Moreover, there are more formalised approaches, employing a graph-based representation of systems and their vulnerabilities, that provide methodologies whose properties are, at least partially, mathematically analysed. Among those approaches, of prominent interest are those based on attack graphs [20, 25], where state-transition diagrams are used to model complex attack patterns. In particular, [20] proposes the use of attack graphs to automate the step of hardening a network against a multi-step intrusions. The proposed security solution is expressed as an adjustable network configuration rather than a set of countermeasures to possible exploits.

Specifically, [19] divides a system into sub-domains and each sub-domain could be characterised by vulnerabilities. Applying probability theory and graph transformations [19] evaluates the possibility that a insecurity flow exploits some vulnerability to penetrate into the system.

The extreme consequence of this family of approaches is to use model-checking techniques to simulate attacks, like in [25].

In this respect, our approach is simpler both in the method and in its formalisation. Despite its simplicity, our results are stronger on the mathematical side and some experimentation [8, 6, 5] make evident the practical value of the method in real-world situations.

In fact, we use the attack tree model [18, 23] to evaluate the security threats combining them with the dependency graph, a formalisation of a piece of experts' knowledge. This combination is the subject of our mathematical analysis, and being a richer structure than the simple attack trees, we are able to derive

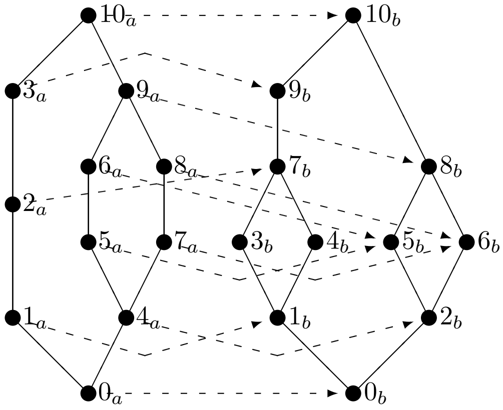

Figure 7: The morphism from the metric a to b

stronger properties for our method.

On a rather different comparison line, the software component paradigm in software engineering has received a great deal of interest from both industries and academia since it allows the reusability of components and a natural approach to distributed programming. A software component is independently developed and delivered as an autonomous unit that can be combined to become part of a lager application.

Despite its evident benefits, the component interdependence is often ignored or overlooked [9], leading to incorrect or imprecise models. In order to avoid this problem, complete models should be specified taking into account system interconnections. In agreement with this point of view [9, 12, 13, 22, 24] present models for assessing security risks taking into account interdependence between components.

Particularly, [9] use techniques for automating and enhancing risk assessment studies of technological processes using qualitative models. A set of fundamental parameters and primitive functions are defined for the domain from which the system behaviour is derived, detecting a number of interesting interdependencies among components.

Similarly, [12] defines a model based on security policy and individual risks. The model gives the possibility to evaluate if the risk associated to each transaction is acceptable. The evaluation of risk also takes into account context information.

With respect to this family of risk assessment methodologies, whose goal is to evaluate the likelihood of a failure in the design of a complex software system, rather than to assess the risk of a malicious intrusion into a telecommunication network, our method appears to be an ad-hoc method. In fact, it has been conceived to analyse the security of a computer network, and, although it can be used in the analysis of information system designs, and, therefore, it may be compared with methodologies in this area, its origin is quite evident.

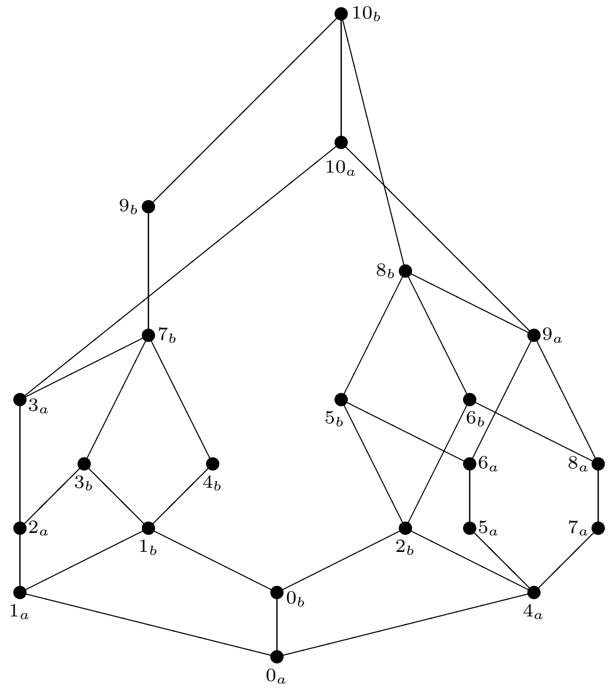

Figure 8: The combined metric c

As a matter of fact, independently from their application areas, the risk assessment methodologies have a core weakness: the use of subjective metrics. In fact, in the scientific community the main criticism to these methodologies is about the fact that values assigned on the basis of a personal knowledge and experience are regarded as *random* values, making the total risk evaluation process to be considered as a *guess*.

It is a fact that the evaluation metric behind exploitability deeply influences the risk evaluation. But, at least in our treatment, what matters is the *structure* of the metric rather than its absolute value.

Generalising, in many field of ICT there is the need to define an objective metric. In the abstract, a metric is defined as [2] the instrument to compare and to measure a quantity or a quality of an observable. The importance of metrics lies in this quotation of Lord Kelvin: "When you can measure what you are speaking about and express it in numbers, you know something about it; but when you cannot measure it, when you cannot express it in numbers, your knowledge is of a meager and of unsatisfactory kind". Our treatment of metrics follows the work of N. Fenton, in particular [14].

In agreement with him, we consider measurement as the process by which numbers or symbols are assigned to attributes of entities, in our case to the exploitability of a vulnerability. Therefore, even though there is no widely recognised way to assess risks and to evaluate the induced damages, there are

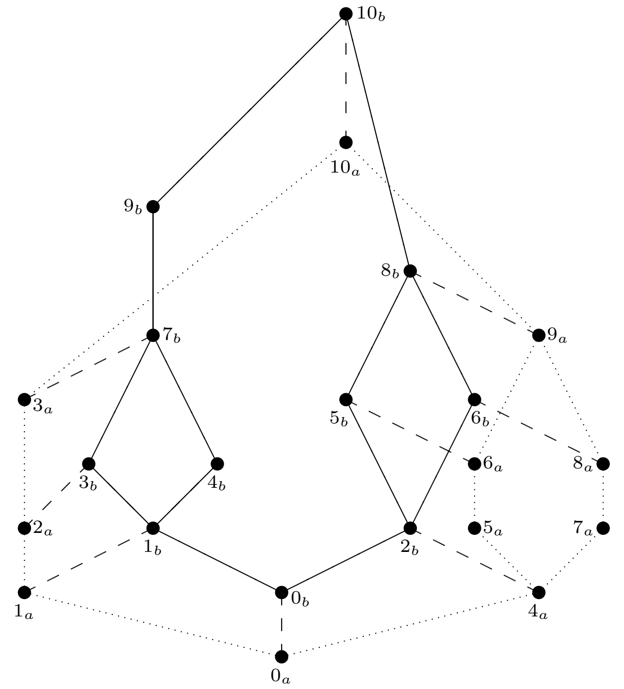

Figure 9: The embedding of a (dotted line) and b (full line) into the metric c

various approaches that provide methodologies by which the risk evaluation becomes more systematic.

In particular, Sharp et al. [24] developed a scheme for probabilistic evaluation of the impact of the security threats and proposed a risk management system with the goal of assessing the expected damages due to attacks in terms of their economical costs.

Z. Dwaikat et al. [12] defined security requirements for transactions and provided mechanisms to measure likelihood of violation of these requirements.

Looking towards risk assessment as a decision support tool, Fenton [15] proposed the use of Bayesian networks. He distinguishes between certain and uncertain criteria and points out the power of Bayesian networks to reason about uncertainty.

Differently, our approach towards objective risk assessment is based on the abstraction over values, thus what matters in our treatment is the *structure* of the metrics. Hence, objectivity is gained by considering values in the metric not as *absolute measures of risk*, but, instead, as *relative evaluations of risks*. Therefore, in agreement with [9, 13, 15, 22], the information computed by our model can be used as a decision support.

5 Conclusions

The presented work addresses the problem of providing a reliable base to risk assessment; this problem arises since a fundamental phase in the risk assessment process is given by the quantitative evaluation of the exploitabilities of the system vulnerabilities. Because the evaluation is performed by human experts, their values, although authoritative, are subjective and, thus, debatable.

The illustrated results show that it is possible to design a risk assessment method which relies only on the *relative* values w.r.t. a metric. Moreover, if two metrics are compatible, we have mathematically proved that the presented method is, in fact, independent from the specific values, giving the same final result in both metrics, modulo a suitable renaming of the values. Moreover, the renaming process encodes the notion of compatibility of metrics in a straightforward way.

Apart the formal results, this work introduces the idea of considering a metric as a partial order of values, modelling the fact that two vulnerabilities cannot always be directly compared. In this respect, we have shown that a risk assessment procedure can work with partial orders as metrics, which is an extension of the standard procedures; moreover, we have proved that the fundamental character of the method, namely its convergence, is guaranteed to hold even in presence of partial metrics. Of course, the given proof covers as a special case also the metrics which are based on a totally ordered set of values.

The central result of this work is contained in Theorem 3.2, that defines the concept of compatible metrics and shows how a risk evaluation is preserved when changing the initial metric with a compatible one. We discussed how this result can be used to compare the evaluations of different experts, when their metrics can be shown to be compatible. In particular, we have shown how to derive a metric that *composes* the various experts' views when their own metrics are mutually compatible.

In this respect, Theorem 3.2 is an initial result of a wider study, whose aim is to identify classes of metrics that are invariant under a risk assessment procedure, thus providing the mathematical model to enable the formal analysis of experts' quantitative evaluations. Therefore, the ultimate goal is to individuate the *essence* of a quantitative evaluation and to use its content instead of the apparent form of an evaluation, which, as shown in the related works, is often misleading. In the present work, we suggested that the essence of the quantitative evaluation resides in the ordering of values inside a metric, while the apparent form is how an evaluation appears with the absolute values of the associated metric. The deep nature of the evaluation is its essence and thus, as we have proved, two evaluations are essentially equivalent if they are the same modulo a suitable renaming of values, where a suitable renaming is a map that preserves the relative order of values.

References

[1] C. Alberts, A. Dorofee, J. Stevens, and C. Woody, Introduction to the Octave approach, October 2003. `http://www.cert.org/octave/approach_intro.pdf`

[2] S. Arshad, M. Shoaib, and A. Shah, Web metrics: The way of improvement of quality of non web-based systems, in H. R. Arabnia and H. Reza, eds., *SERP '06: Proceedings of the International Conference on Software Engineering Research and Practice*, vol. 2, CSREA Press, Las Vegas, NV, USA, 2006, pp. 489–495.

[3] R. Baskerville, Information system security design methods: Implications for information systems development, *ACM Computing Survey*, 25(4)(1993), 375–412.

[4] M. Benini and S. Sicari,Towards More Secure Systems: How to Combine Expert Evaluations, in *SecureComm'08: Proceedings of the 4th International Conference onSecurity and Privacy in Communication Networks-ACM Digital library*, Istanbul, Turkey, September 2008

[5] M. Benini and S. Sicari, Risk Assessment in Practice: A Real Case Study,*Elsevier Computer Communication*, 31(15): 3691-3699, July 2008

[6] M. Benini and S. Sicari, Assessing the risk to intercept VoIP calls, *Elsevier Computer Network* 52(12):2432-2446, August 2008

[7] M. Benini and S. Sicari, A mathematical framework for risk assessment, in *NTMS '07: Proceedings of the First International Conference on New Technologies, Mobility and Security*, Paris, France, March 2007.

[8] M. Benini and S. Sicari, Risk assessment: Intercepting VoIP calls, in *Proceedings of the VIPSI 2007 Venice Conference*, March 2007.

[9] G. Biswas, K.A. Debelak, and K. Kawamura, Application of qualitative modelling to knowledge-based risk assessment studies, in M. Ali, ed., *IEA/AIE '89: Proceedings of the Second International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems*, vol. 1, ACM Press, New York, NY, USA, 1989, pp. 92–101.

[10] S. Burris and H.P. Sankappanavar,A Course in Universal Algebra, Graduate Texts in Mathematics, Springer-Verlag, Heidelberg, DE, 1982.

[11] F. den Braber, T. Dimitrakos, B.A. Gran, M.S. Lund, K. Stølen, and J.Ø. Aagedal, The CORAS methodology: Model-based risk management using UML and UP, in L. Favre, ed., *UML and the Unified Process*, IRM Press, Hershey, PA, USA, 2003, pp. 332–357.

[12] Z. Dwaikat and F. Parisi-Presicce, Risky trust: Risk-based analysis of software system, in *SESS '05: Proceedings of the 2005 Workshop on Software Engineering for Secure Systems — Building Trustworthy Applications*, ACM Press, New York, NY, USA, 2005, pp. 1–7.

[13] S. Evans, D. Heinbuch, E. Kyle, J. Piorkowski, and J. Wallener, Risk-based system security engineering: Stopping attacks with intention, *IEEE Security & Privacy Magazine*, 2(6)(2004), 59–62.

[14] N. Fenton, Software measurement: A necessary scientific basis, *IEEE Transactions on Software Engineering*, 20(3)(1994), 199–206.

[15] N. Fenton and M. Neil, Making decisions: Bayesian nets and MCDA, *Knowledge-Based Systems*, 14(7)(2001), 307–325.

[16] M. Howard and D. Leblanc, Writing Secure Code, Microsoft Press, Redmond, WA, USA, 2003.

[17] B. Jenkins, Risk analysis helps establish a good security posture; risk management keeps it that way, 1998. `http://www.nr.no/{\~{}}abie/RiskAnalysis.htm`.

[18] A.P. Moore and R.J. Ellison, Survivability through intrusion-aware design, Technical Report 2001-TN-001, CERT Coordination Center, 2001.

[19] I.S. Moskowitz and M.H. Kang, An insecurity flow model, in *NSPW '97: Proceedings of the 1997 Workshop on New Security Paradigms*, ACM Press, New York, NY, USA, 1997, pp. 61–74.

[20] S. Noel, S. Jajoidia, B. O'Berry, and M. Jacobs, Efficient minimum-cost network hardening via exploit dependency graphs, in *ACSAC '03: Proceedings of 19th Annual Computer Security Applications Conference*, IEEE Computer Society, Los Alamitos, CA, USA, 2003, pp. 86–95.

[21] F. Redmill, Risk analysis: A subjective process, *Engineering Management Journal*, 12(2)(2002), 91–96.

[22] M. Sahinoglu, security meter: A practical decision-tree model to quantify risk, *IEEE Security & Privacy*, 3(3)(2005), 18–24.

[23] B. Schneier, Attack trees, *Dr. Dobb's Journal*, 12(24)(1999), 21–29.

[24] G.P. Sharp, P.H. Enslow, S.B. Navathe, and F. Farahmand, Managing vulnerabilities of information system to security incidents, in *ICEC '03: Proceedings of the 5th International Conference on Electronic Commerce*, ACM Press, New York, NY, USA, 2003, pp. 348–354.

[25] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J.M. Wing, Automated generation and analysis of attack graphs, in *SP '02: Proceedings of the 2002 IEEE Symposium on Security and Privacy*, IEEE Computer Society, Washington, DC, USA, 2002, pp. 273–284.

[26] R. Shirey, RFC 2828: Internet security glossary, May 2000. http://www.ietf.org/rfc/rfc2828.txt.

[27] S. Sicari, D. Balzarotti, and M. Monga, Assessing the risk of using vulnerable components, in D. Gollmann, F. Massacci, and A. Yautsiukhin, eds., *Quality of Protection. Security Measurements and Metrics*, Springer-Verlag, New York, NY, USA, 2006, pp. 65–78.

[28] T. Siu, Risk-eye for the IT security guy, February 2004. http://www.giac.org/certified_professionals/practicals/gsec/3752.php