Towards More Secure Systems: How to Combine Expert Evaluations

Marco Benini
Dipartimento di Informatica e Comunicazione
Università degli Studi dell'Insubria
via Mazzini 5, IT-21100 Varese, Italy
marco.benini@uninsubria.it

Sabrina Sicari
Dipartimento di Informatica e Comunicazione
Università degli Studi dell'Insubria
via Mazzini 5, IT-21100 Varese, Italy
sabrina.sicari@uninsubria.it

ABSTRACT

In previous works [2, 4] we have introduced a formal risk assessment method and we have shown its mathematical properties. The method allows to model a system as a structured set of vulnerabilities, each one potentially depending on the others: the goal of the method is to consider the influence of the dependencies and, thus, to provide a global risk assessment. A crucial point is the use of order-based metrics to measure the exploitability of a threat: order-based metrics reduce the subjective aspects in the risk evaluation process. This work extends the previous ones by showing how to combine the risk evaluations performed by different experts whose degree of expertise may vary.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations; C.4 [**Performance of Systems**]: Reliability, availability, and serviceability; D.2.8 [**Software Engineering**]: Metrics

General Terms

Security, Measurement

Keywords

Risk assessment, Algebraic metrics, Composition of metrics

1. INTRODUCTION

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

Following Howard and Le Blanc [13] who said "You cannot build a secure system until you understand your threats", to improve the security of a system, a preliminary investigation of its vulnerabilities is performed, the related threats are identified and then, the associated risks are evaluated. In this respect, it helps to understand what level of risk is acceptable [14] and to find a trade-off among risks.

The difficulty to evaluate the risks in a real system lies in the fact that most methods are based on the so-called exploitability values [13]: an exploitability value is a quantitative measure of the easiness to use a vulnerability to damage the system in some way. Usually, 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 connected risks on a personal metric, derived mainly from his experience. Thus, the process is exposed to criticisms since it is ultimately based on personal yet authoritative judgements [17].

Hence, to mitigate personal influences, when it is necessary to evaluate the risk of a complex system, many experts are called to analyse it from their points of view. Each expert adopts a personal metric and he applies a different knowledge depending on his own experience. Therefore, the distrust of the scientific community towards the validity of risk assessment could be traced back to two main problems:

1. The difficulty to objectively compare the risk analyses produced by different security experts;
2. The difficulty to combine the risk analyses produced by experts with different levels of knowledge.

In order to cope with these problems, we have formalised and extended the risk assessment method introduced in [2, 4]. In this paper, we show how to combine the results of the application of the method as obtained by different experts, each one adopting his own metric. Specifically, we show that it is possible the define a common metric that extends the personal metric of each expert in such a way that (1) either the common metric privileges no expert, or (2) it includes the metrics of all experts privileging the most trustable ones. In this way, the results from many experts can be combined in a single, common framework that respects the trust we have in each expert. As a consequence, one can compare the results from different experts since their outcomes can be interpreted in a common framework.

Therefore, the extension of the method in [2,4] is justified since it is based on an evaluation of the exploitability of the vulnerabilities in a system and on their dependencies; the method is mathematically formalised, hence it is possible to prove the properties of our combinations of metrics.

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 and extend the risk assessment method introduced in [2]; here, we illustrate the method, briefly discussing its foundations.

The starting point is to consider a distributed system as a composition of black-box elements communicating by means of 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 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 the design, implementation or management of a system or component that could be used to violate the security policy, see [22].

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* [9, 19], 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 it as children. In turn, each internal node in the tree represents 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 ways to achieve the father goal; `and` nodes represent the steps toward the achievement of the father goal; the leaves of the tree represent the system vulnerabilities.

Although the attack tree 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: 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, our method also considers 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 the 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 measures the loss an attack may cause, e.g, the loss of money, and the level of exploitability is a measure of the difficulty to make an attack, as defined in the STRIDE/DREAD theory [13].

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

1. The threat to the system under examination is modelled using an attack tree and to each vulnerability v is associated an index $E_0(v)$, called its *initial exploitability*, which measures how probable is that v will be exploited 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 v depends on a vulnerability w if and only if when w is already exploited, then v becomes easier to exploit. As already said, we do not limit the analysis to structural dependencies, thus contextual, architectural and topological information is also considered. The dependencies among vulnerabilities are depicted in the *dependency graph* $\mathcal{D} = \langle V, D \rangle$, whose nodes are the vulnerabilities and the edge (w, v) is in D if and only if it is easier to compromise an element suffering the v vulnerability when one has already compromised an element affected by the w vulnerability. Moreover, an exploitability value weights each dependency (u, v) 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 v, assuming that the source vulnerability u has been exploited. This value is called *conditional exploitability* and is denoted as $E(v|u)$.

3. The exploitability of each vulnerability v is calculated by means of its initial value $E_0(v)$ and its dependencies. The algorithm is described in the following.

4. The risk associated to the threat under examination is computed by recursively aggregating the exploitabilities along the attack tree: 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 *lattice*, i.e., a partial order with unique lower and upper bounds: 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. Formally,

Definition 1. A *lattice* is a partial order $\langle \mathcal{O}, \leq \rangle$ such that every pair of elements $x, y \in \mathcal{O}$ has a least upper bound denoted as $x \vee y$ and a greatest lower bound denoted as $x \wedge y$. A lattice is *finite* if $\mathcal{O}$ is a finite set. A lattice is *bounded* if there are two distinct elements 0 and 1 such that 0 is minimal in the lattice, i.e., every element is greater than 0, and 1 is maximal, i.e., every element is less than 1. A lattice is *complete* if any subset $A \subseteq \mathcal{O}$ has a least upper bound denoted as $\bigvee A$ and a greatest lower bound denoted as $\bigwedge A$.

Evidently, it is safe to assume that the metric contains a finite number of elements, since the system vulnerabilities are always finite, and it is safe to assume that the metric is bounded 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. Hence, in mathematical terms, we assume that a metric is a finite and bounded lattice. It holds that every metric is a complete lattice:

LEMMA 1. *Every finite and bounded lattice is complete.*

PROOF. By induction on the cardinality of subsets: a subset composed by one element e has e as its greatest lower bound and as its least upper bound; a subset $A \cup \{e\}$ has as greatest lower bound $e \wedge \bigwedge A$ and as least upper bound $e \vee \bigvee A$. $\square$

Hence, it is possible to define the algorithm to capture the influence of dependencies: the exploitability value is updated according to the rule

$$E_{i+1}(v) = \bigvee(\{E_i(v)\} \cup \{E(v|w) \wedge E_i(w) \colon (w, v) \in D\}) \quad (1)$$

whose rationale is to update a value when it is convenient to use the depencency instead of the direct attack pattern.

In [4] it has been proved that the so-calculated exploitability values converge in finite and bounded time and that the result depends only on the ordering structure of the metric.

3. DERIVING A BALANCED METRIC

Let $\mathcal{O}_a$ and $\mathcal{O}_b$ be two metrics; without loss of generality we can assume that $\mathcal{O}_a \cap \mathcal{O}_b = \emptyset$. Moreover, let $\mathcal{O}_a$ be *compatible* with $\mathcal{O}_b$, that is, there is a function $f\colon \mathcal{O}_a \to \mathcal{O}_b$ such that, for every[1] $x, y \in \mathcal{O}_a$ with $x \leq_a y$, $f(x) \leq_b f(y)$.

We want to construct a metric $\mathcal{O}_c$, i.e. a finite bounded lattice, such that

1. $\mathcal{O}_a$ and $\mathcal{O}_b$ are contained in $\mathcal{O}_c$;
2. the update algorithm (formula (1)) operating in the $\mathcal{O}_c$ metric, when its initial values are all in $\mathcal{O}_a$ ($\mathcal{O}_b$, respectively), yields the same results as if operating in $\mathcal{O}_a$ ($\mathcal{O}_b$, respectively);
3. $x \vee_c y$ and $x \wedge_c y$ are in $\mathcal{O}_a$ ($\mathcal{O}_b$ respectively) if and only if $x, y \in \mathcal{O}_a$ ($\mathcal{O}_b$, respectively).

Henceforth, the metric $\mathcal{O}_c$ extends both $\mathcal{O}_a$ and $\mathcal{O}_b$, preserving the risk evaluations and combining values from different metrics does not yield a result in those metrics, thus preventing one metric to prevail on the other.

Definition 2. Let $\mathcal{O}$ be a complete lattice, a function C from $\wp(\mathcal{O})$, the powerset[2] of $\mathcal{O}$, to itself is called a *closure* operator if, for $A, B \subseteq \mathcal{O}$ it satisfies (1) $A \subseteq C(A)$, (2) $C(C(A)) = C(A)$ and (3) if $A \subseteq B$ then $C(A) \subseteq C(B)$.

LEMMA 2. *The functions C_a and C_b defined as $C_a(A) = \{x \in \mathcal{O}_a \colon x \leq_a \bigvee_a A\}$ and $C_b(B) = \{x \in \mathcal{O}_b \colon x \leq_b \bigvee_b B\}$ are closure operators.*

PROOF. Conditions (1) and (3) are evident; a simple calculation yields condition (2). □

Let $S = \{s_x \colon x \in \mathcal{O}_a\}$ and let us assume without loss of generality that $\mathcal{O}_a \cap S = \emptyset$ and $\mathcal{O}_b \cap S = \emptyset$; moreover, let $K = \mathcal{O}_a \cup \mathcal{O}_b \cup S$. We define C as the function $C(A \cup B \cup T) = C_a(A) \cup C_b(B) \cup T \cup \{s_x \in S \colon x \in C_a(A) \text{ and } \exists y \in C_b(B). y \leq_b f(x)\}$, where $A \subseteq \mathcal{O}_a$, $B \subseteq \mathcal{O}_b$ and $T \subseteq S$. It is immediate to show that:

LEMMA 3. *The function C is a closure operator.*

THEOREM 1. *$\mathcal{O}_c = \langle \{U \subseteq K \colon U = C(U)\}, \subseteq \rangle$ is a finite, bounded and complete lattice.*

PROOF. The proof that the structure is a complete lattice is standard, see [7]; since K is finite, its powerset is finite, thus the structure is finite too; finally, by construction, $\emptyset = C(\emptyset)$ is contained in every subset, so it acts as the 0 of the structure, and $K = C(K)$ contains every subsets thus it acts as the 1 of the structure, thus the lattice is bounded. □

Therefore, $\mathcal{O}_c$ is a finite bounded lattice ordered by subset inclusion, that is, a metric. Moreover, $\mathcal{O}_a$ and $\mathcal{O}_b$ are embedded into $\mathcal{O}_c$: in fact, the image of the function $i_a \colon \mathcal{O}_a \to \mathcal{O}_c$ defined as $i_a(x) = \{y \in \mathcal{O}_a \colon y \leq_a x\}$ is a sublattice of $\mathcal{O}_c$ and it is isomorphic to $\mathcal{O}_a$ being i_a injective; similarly $i_b \colon \mathcal{O}_b \to \mathcal{O}_c$ defined as $i_b(x) = \{y \in \mathcal{O}_b \colon y \leq_b x\}$ is injective, thus $\mathcal{O}_b$ can be embedded into $\mathcal{O}_c$. Hence, $\mathcal{O}_c$ contains both $\mathcal{O}_a$ and $\mathcal{O}_b$ and the images under the injections i_a and i_b are sublattices, i.e., they are closed under least upper bounds and greatest lower bounds. Finally, the risk assessment procedure behaves as expected and required since its operations are just $\bigvee$ and $\bigwedge$ and thus their results are in the image of $\mathcal{O}_a$ ($\mathcal{O}_b$ respectively) when their arguments are.

Moreover, each element A in $\mathcal{O}_c$ is a set and it has a least upper bound[3] that uniquely identifies A.

4. DERIVING A COMMON PRIVILEGING METRIC

As before, let $\mathcal{O}_a$ and $\mathcal{O}_b$ be two metrics; without loss of generality we can assume that $\mathcal{O}_a \cap \mathcal{O}_b = \emptyset$; moreover, let $\mathcal{O}_a$ be compatible with $\mathcal{O}_b$ via the function f from $\mathcal{O}_a$ to $\mathcal{O}_b$.

Following the schema illustrated in the previous section, we want to construct two metrics, $\mathcal{O}_c^a$ and $\mathcal{O}_c^b$ privileging $\mathcal{O}_a$ and $\mathcal{O}_b$, respectively, such that

1. $\mathcal{O}_a$ is contained in $\mathcal{O}_c^b$ as a sublattice, and $\mathcal{O}_b$ is contained in $\mathcal{O}_c^b$ as a suborder[4];
2. $\mathcal{O}_b$ is contained in $\mathcal{O}_c^a$ as a sublattice and $\mathcal{O}_a$ is contained in $\mathcal{O}_c^a$ as a suborder;
3. the update algorithm (formula (1)) operating in the $\mathcal{O}_c^a$ metric, when its initial values are in $\mathcal{O}_b$, yields the same result as in $\mathcal{O}_b$; in the same situation, the update algorithm whose initial values are in $\mathcal{O}_a$, yields a result which is, as most, the result as if operating in $\mathcal{O}_a$;
4. the update algorithm (formula (1)) operating in the $\mathcal{O}_c^b$ metric, when its initial values are in $\mathcal{O}_a$, yields the same result as in $\mathcal{O}_a$; in the same situation, the update algorithm whose initial values are in $\mathcal{O}_b$, yields a result which is, as most, the result as if operating in $\mathcal{O}_b$;
5. in $\mathcal{O}_c^b$ it holds that $x \in \mathcal{O}_a$ is less than $f(x)$, thus privileging the values in $\mathcal{O}_b$;
6. oppositely, in $\mathcal{O}_c^a$ it holds that $x \in \mathcal{O}_a$ is greater than $f(x)$, thus privileging the values in $\mathcal{O}_a$.

Let $H = \mathcal{O}_a \cup \mathcal{O}_b$: we define $C^a(A \cup B) = C_b(B) \cup C_a(A \cup \{x \in \mathcal{O}_a \colon f(x) \in C_b(B)\})$ and $C^b(A \cup B) = C_a(A) \cup C_b(B \cup \{f(x) \colon x \in C_a(A)\})$ where $A \subseteq \mathcal{O}_a$ and $B \subseteq \mathcal{O}_b$. It is almost immediate to prove that C^a and C^b are closure operators.

Therefore, $\mathcal{O}_c^a = \langle \{U \subseteq H \colon U = C^a(U)\}, \subseteq \rangle$ and $\mathcal{O}_c^b = \langle \{U \subseteq H \colon U = C^b(U)\}, \subseteq \rangle$ are metrics, following the guidelines of Theorem 1.

As in the previous section, it is immediate to show that $\mathcal{O}_a$ can be embedded into $\mathcal{O}_c^b$, and that $\mathcal{O}_c^a$ contains a copy of $\mathcal{O}_b$. Being these copies sub-lattices of the combined metric, the update algorithm operates on the copy as on the original metric. Also, by construction, the values in the combined lattices are ordered as requested.

Moreover, as before, each element U in the combined metrics is uniquely identified by the least upper bound of U.

[1]We use subscripts to indicate the metric, e.g., $\leq_a$ denotes the less-than relation in $\mathcal{O}_a$.

[2]The powerset of a set is the set of its subsets.

[3]In more algebraic terms, the elements of $\mathcal{O}_c$ are *ideals* thus they are closed under the $\bigvee$ and the $\bigwedge$ operations.

[4]Requiring $\mathcal{O}_b$ to be a sublattice would imply that 0_b equals 0_c^b thus making 0_a and 0_b equivalent.

5. AN ILLUSTRATING EXAMPLE

This section tries to clarify our findings by means of an abstract example that evidences the core of our results without the complexities of a real case study.

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 both on 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, 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 status of a system.

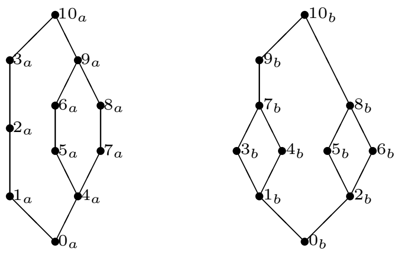

Figure 1: The metrics a and b

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. In our example, Alice adopts the metric a while Bob uses the metric b; both of them are represented in Fig. 1. 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.

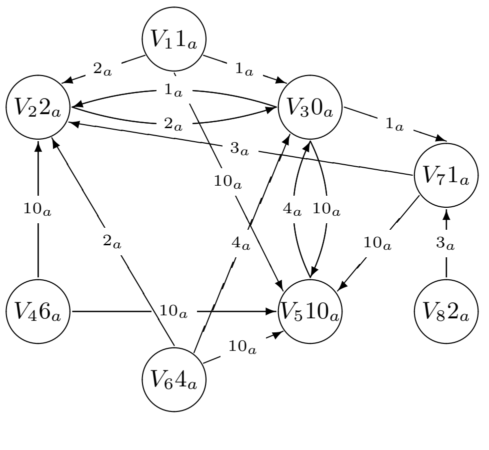

Figure 2: The initial evaluation of Alice

In the scenario, Alice develops an initial evaluation of the exploitability values, synthesised in Fig. 2; Bob does the same, as illustrated in Fig. 3. These evaluations are the result of the application of the experts' experience and judgement, thus, at least to some extent, the values are subjective.

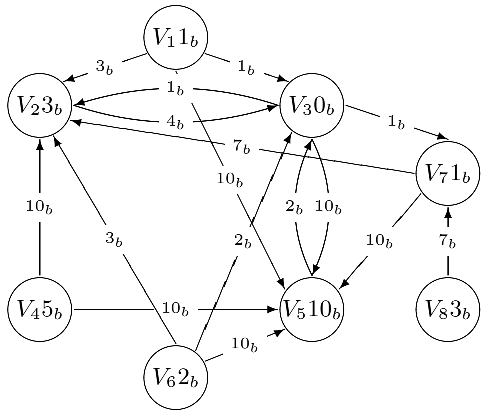

Figure 3: The initial evaluation of Bob

Applying our method, Alice and Bob can calculate their final risk assessments, considering also the role of dependencies: after a few iterations of the application of (1), Alice derives the following risk vector

V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8
1_a	10_a	10_a	6_a	10_a	4_a	2_a	2_a

while Bob obtains as his final result

V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8
1_b	10_b	10_b	5_b	10_b	2_b	3_b	3_b

It is evident that the derived evaluations are different, so we expect the measured risk to differ.

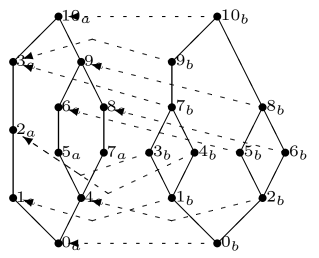

Figure 4: The function from the metric b to a

In order to compare their results, one tries to build an order-preserving mapping from one metric to the other. For example, the metric b used by Bob can be mapped in the metric a of Alice via the function f shown in Fig. 4: it is immediate to check that, if $x \leq y$ in the metric b, then $f(x) \leq f(y)$ in the metric a.

Following the construction in Section 3, one obtains a common balanced metric shown in Fig. 5 where the solid lines reveal the original metrics a and b, and the dashed lines show the newly introduced relations.

Hence, it is possible to combine the results of Alice and Bob by assigning to each vulnerability a value which is the greatest upper bound of the values calculated by our experts. The resulting common evaluation is:

V_1	V_2	V_3	V_4	V_5	V_6	V_7	V_8
s_1	s_{10}	s_{10}	s_5	s_{10}	s_2	s_3	s_3

Finally, the privileging metrics are shown in Fig. 6: their pictorial representation can be misleading since the nodes should be thought as sets, but it conveys the intuition behind their construction. We

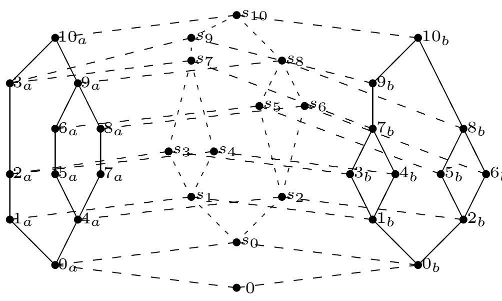

Figure 5: The balanced combination of a and b

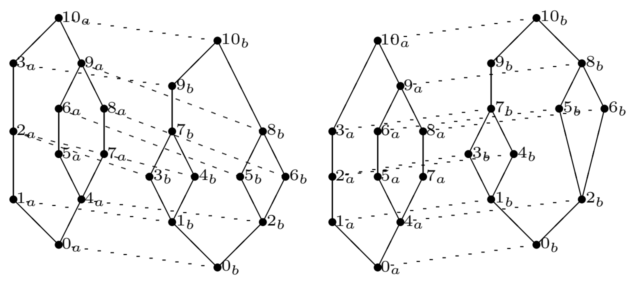

Figure 6: The metrics privileging a (left) and b (right)

leave to the reader to see how the evaluations of Alice and Bob get interpreted in these metrics.

6. RELATED WORKS

Even though the application of risk evaluation methods has been widely discussed and analysed, see, e.g., [1, 6, 14], among information security experts there appears to be no agreement regarding the most appropriate method to assess the probability of computer incidents [20]. 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 that sometimes could be combined to improve the accuracy of the security evaluation.

On the side of systematic approaches, [10] present a system security engineering methodology 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, [18] provides a probabilistic quantitative model that measures security risk where it is possible to calculate risk starting from semi-numerical values.

With respect to the previous works, our approach, starting from its initial definition in [2], has been based on the structured evaluation of the vulnerabilities along with their mutual dependencies. In this respect, the results in [10] are similar to ours, although they do not propose a formal method based on mathematical arguments.

Moreover, there are formalised approaches, employing a graph-based representation of systems and their vulnerabilities, that provide methods whose properties are, at least partially, mathematically analysed. Among those approaches, of prominent interest are those based on attack graphs [16, 21], where state-transition diagrams are used to model complex attack patterns. In particular, [16] 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, [15] divides a system into sub-domains and each sub-domain could be characterised by vulnerabilities. Applying probability theory and graph transformations [15] 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 [21].

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 experimentations make evident the practical value of the method in real-world situations. In fact, we use the attack tree model [9, 19] 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 can derive 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 [5], leading to incorrect or imprecise models. To avoid this problem, complete models should be specified taking into account system interconnections. In agreement with this point of view [5, 8, 10, 18, 20] present models for assessing security risks taking into account interdependence between components.

Particularly, [5] uses 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, [8] 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 taking into account context information.

With respect to this family of risk assessment methods, whose goal is to evaluate the likelihood of a failure in the design of a complex software system, our method appears to be ad-hoc. 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, when compared with methods in this area, its origin is quite evident.

As a matter of fact, independently from their application areas, the risk assessment methods have a core weakness: the use of subjective metrics. In fact, in the scientific community the main criticism to these methods 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.

Our treatment of metrics follows the work of N. Fenton, in particular [11]. 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 various approaches by which the risk evaluation becomes more systematic.

In particular, Sharp et al. [20] develop a scheme for probabilistic evaluation of the impact of the security threats and propose a risk management system with the goal of assessing the expected damages due to attacks in terms of their economical costs. Also, Z. Dwaikat et al. [8] define security requirements for transactions and provide mechanisms to measure likelihood of violation of these requirements.

Looking towards risk assessment as a decision support tool, Fenton [12] propose the use of Bayesian networks as a way to deal with uncertain criteria.

Differently, our approach towards objective risk assessment is based on the abstraction over values. Hence, objectivity is gained by considering values in the metric not as *absolute measures of risks*, but, instead, as *relative evaluations of risks*. Therefore, as [5, 10, 12, 18], the information computed by our model can be used as a decision support.

7. CONCLUSIONS

In this work, we have addressed the problem of combining the metrics from different experts who have analysed a system to assess the possible risks. The combination should be transparent to the various experts, not influencing their analyses and it should be sound, not distorting the meaning of the experts' results: employing a formal risk assessment method and using its mathematical formulation, we have shown how to derive combinations of metrics with the requested properties, both in the case when an expert is more reliable than the others, and in the case there is no reason to prefer an expert over the others. By a clever usage of the presented techniques, any order of trust/preference among the experts leads to a suitable common metric where the outcomes of experts' evaluations can be compared and combined in a sound way.

Therefore, the composition of knowledge from different experts becomes significant and our contribution has been to provide the conceptual instruments to perform mathematically sound combinations in a systematic way.

8. REFERENCES

[1] C. Alberts et al.: Introduction to the Octave approach (2003). `http://www.cert.org/octave`

[2] D. Balzarotti, M. Monga, S. Sicari: Assessing the risk of using vulnerable components. In: D. Gollmann, F. Massacci, A. Yautsiukhin (eds.) Quality of Protection. Security Measurements and Metrics, pp. 65–78. Springer-Verlag (2006).

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

[4] M. Benini, S. Sicari: A mathematical framework for risk assessment. In H. Labiod, M. Badra (eds.) New Technologies, Mobility and Security, Signals and Communication, pp. 459–469. Springer-Verlag (2007)

[5] G. Biswas, K.A. Debelak, K. Kawamura: Application of qualitative modelling to knowledge-based risk assessment studies. In M. Ali (ed.) Proc. of the 2nd Intl. Conf. on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems, pp. 92–101. ACM Press (1989).

[6] F. den Braber et al.: The CORAS methodology: Model-based risk management using UML and UP. In L. Favre (ed.) UML and the Unified Process, pp. 332–357. IRM Press (2003).

[7] S. Burris, H.P. Sankappanavar: A Course in Universal Algebra. Graduate Texts in Mathematics, Springer-Verlag (1982).

[8] Z. Dwaikat, F. Parisi-Presicce: Risky trust: Risk-based analysis of software system. In Proc. of the Software Engineering for Secure Systems — Building Trustworthy Applications, pp. 1–7. ACM Press (2005).

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

[10] S. Evans et al.: Risk-based system security engineering: Stopping attacks with intention. IEEE Security & Privacy Magazine **2(6)**, 59–62 (2004).

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

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

[13] M. Howard, D. Leblanc: Writing Secure Code. Microsoft Press (2003).

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

[15] I.S. Moskowitz, M.H. Kang: An insecurity flow model. In Proc. of New Security Paradigms, pp. 61–74. ACM Press (1997).

[16] S. Noel et al.: Efficient minimum-cost network hardening via exploit dependency graphs. In Proc. of 19th Annual Computer Security Applications Conf., pp. 86–95. IEEE Computer Society (2003).

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

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

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

[20] G.P. Sharp et al.: Managing vulnerabilities of information system to security incidents. In Proc. of the 5th Intl. Conf. on Electronic Commerce, pp. 348–354. ACM Press (2003).

[21] O. Sheyner et al.: Automated generation and analysis of attack graphs. In Proc. of the IEEE Symposium on Security and Privacy, pp. 273–284. IEEE Computer Society (2002).

[22] R. Shirey: RFC 2828: Internet security glossary (2000). `http://www.ietf.org/rfc/rfc2828.txt`