RZ 3758 (# 99768) 11/12/2009
Computer Science 19 pages

Research Report

A Unifying Theory of Security Metrics with Applications

Klaus Julisch

IBM Research – Zurich
8803 Rüschlikon
Switzerland

Email: kju@zurich.ibm.com

LIMITED DISTRIBUTION NOTICE

This report has been submitted for publication outside of IBM and will probably be copyrighted if accepted for publication. It has been issued as a Research Report for early dissemination of its contents. In view of the transfer of copyright to the outside publisher, its distribution outside of IBM prior to publication should be limited to peer communications and specific requests. After outside publication, requests should be filled only by reprints or legally obtained copies (e.g., payment of royalties). Some reports are available at http://domino.watson.ibm.com/library/Cyberdig.nsf/home.

Research
Almaden • Austin • Beijing • Delhi • Haifa • T.J. Watson • Tokyo • Zurich

A Unifying Theory of Security Metrics with Applications

Klaus Julisch
IBM Research
Zurich Research Laboratory
Switzerland

November 12, 2009

Abstract

A large number of measures have been proposed under the umbrella of "security metrics". Some of these measures are percentages, others are frequencies, numbers, monetary amounts or other units. This absence of a basic unit of measurement is aggravated by a general lack of theory and consistency in the field of security metrics. This paper tries to fill the void by proposing a unifying theory of security metrics. Towards this end, we define security metrics by the properties (validity, accuracy, and precision) they have to fulfill. We clearly differentiate security metrics from the related concepts of *risk metrics*, *compliance metrics*, and *threat metrics*. We further introduce a new classification scheme for security metrics, which helps us review the prior work and identify pitfalls that metrics authors should be aware of. Finally, we show how the theory developed in this paper can be applied to help managers make IT security decisions. Most importantly, the presented theory implies two novel rules for deciding how much money to spend on security and how to allocate this money among multiple systems.

1 Introduction

The term *security metric* is commonly used to denote a measure that quantifies some aspect of the security of an IT system. Over the years, an abundance of quantities have been proposed as "security metrics", for example [1, 2, 3, 4, 5]:

- Time required to break a password;
- Number of security incidents;
- Number of software vulnerabilities;
- IT security spending as percentage of IT budget;
- Percentage of individuals screened before being granted access to organizational information and information systems;
- Extent (on 0, ..., 10 scale) to which security roles and responsibilities have been defined;
- Mean time from patch availability to patch approval to patch installation;
- Percentage of systems patched to the latest patch level;

Several things are striking about these examples: First, there is no agreed scale on which security is measured. Some of the above examples are percentages, others are times, numbers, frequencies, and so on. This is in stark contrast to all physical phenomena such as distance, weight, brightness, or power, which have well-defined units of measurement (meters, grams, lux, and watt, respectively). This lack of a basic unit of measurement renders the comparison or combination of security metrics difficult or impossible. It also illustrates the immaturity of the field because even a commonly accepted unit of measurement is missing.

The second observation to be made is that many of today's security metrics are vastly under-specified. For example, the "time required to break a password" cannot be measured until one specifies what software and hardware to use. Similarly, the "percentage of individuals screened" is impossible to calculate until we are told what constitutes a "screening".

A third observation is that many of these measures are difficult to interpret. For example, is it good or bad if an organization spends 5% of its IT budget on security? Or, is it good or bad if on average our passwords take 30 minutes to break (assuming the procedure for measuring times has been fully specified)? As no reference point has been defined, these metrics can only be interpreted in a relative sense where, for example, 40 minutes to break a password is better than 30 minutes.

Last but not least, we are concerned that the validity of many of these metrics is unclear. For example, is it more important to improve the "average time to apply a patch, measured across all systems" or the "percentage of systems patched within one week"? There is evidence that the latter is more important from a security point of view [6], but the authors of these metrics do not point this out. Or what good is a metric that measures the "IT security spending" given that there is no evidence that businesses that spend more on security products will necessarily experience a corresponding reduction in security incidents [7]. The general problem is that too many security metrics have not been tested for their validity and relevance to IT security.

To summarize, today's state of the art in security metrics is marked by ad-hoc metrics that lack a common scale, tend to be ambiguously defined, difficult to interpret, and of limited validity. This paper proposes a unifying theory for security metrics, which mitigates these shortcomings. Further novel contributions include the application of this theory to determine the optimal amount of security spending and the optimal way of splitting this spending among multiple systems. We further propose a novel classification of security metrics and identify frequent pitfalls in the definition of security metrics.

The remainder of this paper is structured as follows: Section 2 defines security metrics and explains why we chose this definition. Section 3 develops a classification scheme for security metrics. Section 4 contains two applications of our theory: Firstly, we review and classify the prior art in security metrics; this leads us to identify pitfalls that metrics authors must address in their work. Secondly, we show how the unifying theory can be used to make information security investment decisions. Section 5 summarizes and concludes the paper.

2 Security Metrics Defined

It has been deplored [5, 8] that security metrics tend to become ends in themselves with no clear objective or purpose. Before defining what security metrics are, it is therefore important to ask what they are to be used for, i.e. what objective they serve [9] . The answer to this question can be gleaned from the popular phrase that *"you cannot manage what you cannot measure"*: Metrics in general are tools for management, and security metrics in particular, are a means to support and improve the management of information security.

In particular, there are three management task that information security managers seek help with: First, managers want to measure and track their security over multiple time periods and relative to competitors or industry averages. Second, managers want help with the investment decision of how much to spend on security in general and how to allocate the security budget among various systems that need protection and technologies that promise to deliver this protection. Third, managers want to know if the security mechanisms deployed are adequate and effective, given their business needs [10]. The purpose of security metrics is to help managers

perform these (and other) management tasks.

Next, we have to become more specific about what "security" – this abstract thing we seek to measure – really is. Most people find it difficult to define security let alone to formalize it. We therefore choose an indirect approach and define security via its most concrete *manifestation*, namely the existence of losses when security is absent. More formally, for a given IT system $\mathbb{S}$, we calculate the *expected loss (EL)* from security incidents. In a first approximation, $EL = \sum_{loss} P(loss) \times size(loss)$. Section 2.1 refines this equation and proofs the equation:

$$EL = value \times E(T_m) \times vulnerability(\mathbb{S}) \tag{1}$$

In this equation, *value* is the monetary value (in dollars or Euros) of the IT system $\mathbb{S}$. The factor $E(T_m)$ is the expected malice of the threat environment in which $\mathbb{S}$ exists. It is important that this factor is entirely determined by the threat environment of $\mathbb{S}$; in particular, the expected malice $E(T_m)$ has nothing to do with the technical characteristics of system $\mathbb{S}$ itself. The last factor is a measure of how vulnerable the IT system $\mathbb{S}$ is. This factor is a weighted sum of the frequency of vulnerabilities, multiplied by their severities, multiplied by how easily they can be exploited. Specifically, system $\mathbb{S}$ is all the more vulnerable the more frequent, severe, and easily exploitable its vulnerabilities are. Section 2.1 formalizes these matters.

Let us now return to our initial question: What is security? To answer this question, let us assume that without changing the value of the IT system $\mathbb{S}$, we make some changes to its architecture, which results in a lower expected loss EL. This obviously means that we *improved security* – after all, reduced losses are the manifestation of better security. However, this security improvement came about without changing the value of the system $\mathbb{S}$ and without changing the expected malice $E(T_m)$ of its threat environment. Therefore, the drop in EL must be the result of the $vulnerability(\mathbb{S})$ factor dropping. So, given our premise that the security of a system manifests itself in the expected losses, we can conclude that the security of a system is inversely related to its vulnerability. This gives rise to the following (still informal) definition:

Definition 2.1 (Security Metric, Informal) *A security metric is a function that measures a system's vulnerability and maps it to a real nonnegative number that is inversely related to the system's vulnerability. The purpose of a security metric is to help management make better information security decisions.*

It is important to understand that this definition is *not* a law of nature, nor is it the only possible definition. Rather, it is a sound definition that follows from Equation (1). Definition 2.1 is further consistent with the field of software quality metrics. As explained by Kan [11], the number of "bugs" is an important metric of software quality. In operational terms, "bugs" may be measured by the number of defects relative to the software size, the mean time to failure, or by other means. Ultimately, however, many software quality metrics are tantamount with "bugginess metrics". As vulnerabilities are a special kind of bugs (namely bugs, which undermine the confidentiality, integrity, or availability of a system) we conclude that security metrics are a special case of software quality metrics because they measure a form of bugginess (namely vulnerability). This result provides further justification for Definition 2.1.

An important difference between bugs and vulnerabilities is that vulnerabilities are more multi-faceted. Specifically, counting the number of vulnerabilities of a system says little about how vulnerable or secure it really is. Counting bugs, by contrast, is an acceptable measure of software quality. This difference arises because vulnerabilities differ in their *severity* (how much power an attacker gains by exploiting them) and their *ease of exploitation* (how skilled an attacker has to be in order to exploit a vulnerability). Adding up vulnerabilities of different severity or ease of exploitation is therefore a meaningless exercise. Rather, a measure of vulnerability must weight vulnerabilities by their severity and ease of exploitation, as is done by the formula derived in the following section.

2.1 Derivation of the Expected Loss EL

This section derives Equation (1), according to which the expected loss EL in a system $\mathbb{S}$ abides by the equation $EL = value \times E(T_m) \times vulnerability(\mathbb{S})$. Two points are worth mentioning: First, the derivation is done in a model, which simplifies some aspects of real IT systems, while maintaining their main characteristics. Second, the derivation uses probabilities and other quantities that are difficult or impossible to obtain in practice. This is not a problem as we do not intend to calculate the expected loss. Rather, we seek to establish the formal relationship between the expected loss in an IT system and the system's monetary value, its vulnerability, and its threat environment. For this, it only matters that all quantities are well-defined and existent, which they are.

Our derivation of Equation (1) proceeds similar to [12], [13], and [14]: Let us consider an IT system $\mathbb{S}$, which has a set of vulnerabilities $\mathbb{V}$ and exists in a threat environment $\mathbb{T}$. The threat environment is the set of all actors (insiders and outsiders) who have access to system $\mathbb{S}$ and may damage it (either intentionally or accidentally). Finally, let ξ be a time duration such as "twelve hours", "one day", or "one month" . Our objective is to calculate the expected loss EL from security incidents during a time period of duration ξ.

IT systems as well as threat environments are dynamic in the sense that new vulnerabilities and threats emerge, while old ones get mitigated or disappear. To model this dynamism, we divide time into consecutive intervals of duration ξ. In our model, a vulnerability V, $V \in \mathbb{V}$, and a threat T, $T \in \mathbb{T}$, are either present during an entire ξ-interval or they are not present at all. Further, $P(V)$ is the probability that vulnerability V is present during a given ξ-interval, and $P(T)$ is the probability of a threat being active during a given ξ-interval.

To damage system $\mathbb{S}$, a threat has to target a vulnerability that the threat can exploit with its skill level. Successful exploitation gives the threat certain privileges on system $\mathbb{S}$, which it then uses to inflict damage on $\mathbb{S}$. The extent of damage inflicted depends on how malicious the threat is. For example, a "hobby hacker" might inflict no damage and merely enjoy the "thrill of victory". Economically or politically motivated threats, by contrast, will be more malicious and may inflict noticeable damage to system $\mathbb{S}$. To formalize these concepts, we define:

System value *value*: The variable *value* denotes the worst case monetary loss that the owners of system $\mathbb{S}$ can incur if $\mathbb{S}$ is compromised. This loss value includes direct repair costs, legal costs, reputation costs, and all other potential costs. It is measure in dollars, Euros, or some other currency.

Malice $T_m \in [0,1]$ of a threat $T \in \mathbb{T}$: Some threats exploit vulnerabilities just for the thrill of victory, while others do so in order to inflict damage on the target system $\mathbb{S}$. Threats therefore differ with respect to their *malice*, which we measure on a scale from 0 to 1. The interpretation is that a threat of malice x would cause x percent of the maximum loss it can theoretically inflict.

Skill level $T_s \in [0,1]$ of a threat $T \in \mathbb{T}$: Every threat T has a different skill level reflecting its knowledge, experience, access to hacking tools, and other factors. To define the skill level of a threat, first note that both the sets $\mathbb{V}$ and $\mathbb{T}$ are finite sets because IT systems have a finite number of states and actors. Let $s(T) := |\{V \in \mathbb{V} : \text{T can exploit V within } \xi \text{ time units}\}|/|\mathbb{V}|$ be the fraction of vulnerabilities in $\mathbb{V}$ that T can exploit within ξ time units. The intuition is that the more skilled a threat T is, the larger its $s(T)$ and vice versa. We now define the skill level $T_s := |\{t \in \mathbb{T} : s(t) < s(T)\}|/|\mathbb{T}|$. Note that T_s maintains the same order of threats defined by $s()$, but it has the added property that $1 - T_s$ equals the probability of a threat having skill level T_s or larger. We will need this property below.

Privilege level $V_p \in [0,1]$ attained by exploiting vulnerability $V \in \mathbb{V}$: Vulnerabilities differ in the privilege level an attacker can attain by exploiting them. For example, some vulnerabilities give the attacker root privileges on the target system, while others only enable the attacker to read some data or slow down some system function. We define the privilege level V_p to be the fraction of the system's value that is exposed when the vulnerability V is successfully exploited. The privilege level of a vulnerability is also known as its *severity*.

Difficulty $V_d \in [0,1]$ of exploiting vulnerability $V \in \mathbb{V}$: Vulnerabilities also differ in how easy or difficult they are to exploit. Some vulnerabilities (e.g. race conditions) require considerable skill to exploit, while other's such as default passwords are much easier to exploit. Formally, we define the difficulty V_d of exploitation as the skill level of the least skilled threat that can exploit vulnerability V, i.e. $V_d = \min($ $T_s | T \in \mathbb{T}$ and T can exploit V within ξ time units).

As has been mentioned earlier, it does not matter that we cannot calculate many of the above quantities. All that matters is that they are well-defined and existent. Also note that the above definitions implicitly assume a one-step attack model, i.e. we only consider the damage a threat can cause in one step by exploiting one vulnerability. In reality, threats can propagate throughout systems by exploiting multiple vulnerabilities. Our model does not capture this added complexity. We use the notation $P(breach|T \wedge V)$ to designate the conditional probability that a security breach occurs when the threat T and vulnerability V coincide. The expected loss EL now follows as:

$$
\begin{aligned}
EL &= \sum_{T\in\mathbb{T}} \sum_{V\in\mathbb{V}} [P(T) \times P(V) \times P(breach|T \wedge V)] \times [value \times V_p \times T_m] && (2)\\
&= value \times \sum_{T\in\mathbb{T}} \left\{ P(T) \times T_m \times \sum_{V\in\mathbb{V}} [P(V) \times V_p \times P(breach|T \wedge V)] \right\} && (3)\\
&= value \times \sum_{T\in\mathbb{T}} \left\{ P(T) \times T_m \times \sum_{V\in\mathbb{V}\wedge T_s \geq V_d} [P(V) \times V_p] \right\} && (4)\\
&= value \times \sum_{T_m} \sum_{T_s} \left\{ P(T_m \wedge T_s) \times T_m \times \sum_{V\in\mathbb{V}\wedge T_s \geq V_d} P(V) \times V_p \right\} && (5)\\
&= value \times \left\{ \sum_{T_m} P(T_m) \times T_m \right\} \times \left\{ \sum_{T_s} P(T_s) \times \sum_{V\in\mathbb{V}\wedge T_s \geq V_d} P(V) \times V_p \right\} && (6)\\
&= value \times \left\{ \sum_{T_m} P(T_m) \times T_m \right\} \times \left\{ \sum_{V\in\mathbb{V}} P(V) \times V_p \times \sum_{T_s \geq V_d} P(T_s) \right\} && (7)\\
&= value \times E(T_m) \times \sum_{V\in\mathbb{V}} [P(V) \times V_p \times (1 - V_d)] && (8)\\
&= value \times E(T_m) \times vulnerability(\mathbb{S}) && (9)
\end{aligned}
$$

Let us consider these equations and transformations in detail: Equation (2) expresses the expected loss EL as a sum over all threats T and all vulnerabilities V. Each summand is the product of two factors: The first factor (i.e. $P(T) \times P(V) \times P(breach|T \wedge V)$) is the probability that threat T and vulnerability V occur in the same ξ-interval and that they result in a security breach. The second factor is the ensuing loss, which is the value of system $\mathbb{S}$ multiplied by the percentage V_p of its value that is exposed when vulnerability V is exploited, multiplied by the malice factor T_m of the threat. Equation (3) merely rearranges the terms.

Equation (4) follows from (3) if we assume that $P(breach|T \wedge V) = 1$ if the threat's skill level exceeds the vulnerability's difficulty (i.e. $T_s \geq V_d$) and $P(breach|T \wedge V) = 0$, otherwise. This modelling assumption simplifies reality, but can be justified as follows: The condition $T_s \geq V_d$ means that threat T is capable of exploiting at least as many vulnerabilities as another threat T^*, which can exploit V. This means that in a sense, threat T has superior skill to T^* and consequently should be capable of exploiting vulnerability V. This is particularly so if the time interval ξ is large enough to allow T to learn new attack techniques that are needed for vulnerability V. This argument falls short when V is a very peculiar vulnerability that only highly specialized threats are capable of exploiting. In this case, model and reality diverge.

Equation (5) replaces the sum $\sum_{T \in \mathbb{T}}$ by two separate sums, a first sum over all malice levels T_m and a second sum over all skill levels T_s. Equation (6) then follows if we make the modelling simplification that $P(T_m \wedge T_s) = P(T_m) \times P(T_s)$. In words, this simplification means that the probability of a threat having malice T_m and skill T_s is equal to the probability that a threat has malice T_m multiplied by the probability that a threat has skill T_s. In our model, the malice and skill levels of a threat are therefore independent of each other.

Equation (7) rearranges the second factor of the Equation (6). Equation (8) makes two transformations: First, it uses the fact that $\sum_{T_m} P(T_m) \times T_m$ is the "expected malice" $E(T_m)$ of the threat environment $\mathbb{T}$. Second, it uses the fact that the sum $\sum_{T_s \geq V_d} P(T_s)$ equals the probability of a threat having a skill level of V_d or larger. This probability was shown to equal $1 - V_d$ (see page 4). Equation (9) replaces the last factor of Equation (8) by the function $vulnerability(\mathbb{S})$. This reflects the fact that the function

$$vulnerability(\mathbb{S}) = \sum_{V \in \mathbb{V}} P(V) \times V_p \times (1 - V_d) \tag{10}$$

measures the degree of vulnerability of system $\mathbb{S}$. As has been emphasized earlier, this measure weights the three key factors that determine the degree of vulnerability of a system. These factors are first, the probability of vulnerabilities, second, the privilege level V_p that a vulnerability confers, and third, the ease $1 - V_d$ of exploiting it.

In our model world, we could hence proof Equation (1). Using this equation, Section 2 had derived Definition 2.1, which defines a security metric as a function whose return value is inversely related to the vulnerability of the measured system. Section 2 had further shown that this definition is consistent with the field of software quality metrics. All this serves to justify Definition 2.1, which is why we will use it to further develop our unified theory of security metrics.

2.2 Operational Properties of Metrics

So far, we have given a semantic characterization of what security metrics are. However, not every measure of a system's vulnerability is a security metric. Rather, *measurement theory* defines three operational properties that *any* metric has to fulfill [5, 15]. These properties are:

Validity: Validity is the extent to which a metric adequately reflects the meaning of the concept it seeks to measure. In other words, valid metrics are highly correlated to the concepts they try to measure. For intangible concepts such as "security", "trustworthiness", or "intelligence", it is generally difficult to define valid metrics because the underlying concepts are difficult to grasp. In the case of security, a formal notion of validity follows from Definition 2.1 and Equation (10), as will be shown in Definition 2.2.

Accuracy: Metrics frequently measure things that are not readily observable. The *accuracy* of a metric is the difference between the value determined by the metric and the "true" or "real" value of the object measured. Measurement errors are the principal source of inaccuracies. For example, a metric that uses a vulnerability scanner to measure the number of vulnerabilities in an IT system is accurate if its return values match or are close to the actual number of vulnerabilities.

Precision: Precision assesses reproducibility of measurements. More precisely, it represents the amount of variability among repeated measurements of the same object under similar conditions. For example, the "number of virus alerts" is an imprecise metric of system security because consecutive measurements can vary significantly depending on the threat environment. Specifically, when there are more or fewer virus attacks, the "number of virus alerts" will be higher or lower even though nothing in the system or its security has changed.

The requirement for precision demonstrates another important point: A metric is defined by its *measurement process*, which is a detailed and unambiguous operational description of how the metric is calculated [5]. All relevant conditions and constraints of how the metric is to be derived are defined in the measurement process. The "number of virus alerts" metric mentioned above fails to properly specify the measurement process. For example, it does not specify which virus scanner to use or which threat environment to perform the measurement in. While it is easy to specify a virus scanner for the measurement, the unspecified nature of the threat environment poses a dilemma:

- If the threat environment is left unspecified, then the "number of virus alerts" is a metric of poor precision because it depends on random factors such as the mood, skill, and productivity of virus programmers. (One can postulate that fluctuations in the threat environment average out over long enough measurement intervals, and the "number of virus alerts" should consequently be a precise metric when measured over a sufficiently long time interval. This hypothesis, however, still has to be proven.)
- On the other hand side, trying to define the threat environment would mean that one conducts laboratory experiments, which threatens to destroy the validity of the metric. This is because even a very insecure system can score well in a laboratory experiments where the virus threat is low. But if an insecure system scores high on a security metric, then this obviously means that the metric is not valid.

The conclusion is that "number of virus alerts" is a poor security metric. It may be an interesting statistic that managers are keen to monitor, but it is *not* a useful security metric. More generally, security metrics that depend on empirical runtime measurements frequently suffer from the same dilemma that random factors (most notably the presence of attackers) destroy precision if they are left uncontrolled; and they destroy validity if one attempts to control them. This discussion is illustrative of the kind of analysis that is needed to differentiate genuine security metrics from mere security statistics.

2.3 Formal Definition and Discussion

Definition 2.2 (Security Metric, Formal) *A security metric is a function $m()$ from the set of all IT systems to the set of nonnegative real numbers. It further is valid, accurate, and precise in the following sense:*

- Validity: *For any two IT systems $\mathbb{S}_1$ and $\mathbb{S}_2$, the relationship $m(\mathbb{S}_1) \leq m(\mathbb{S}_2) \Leftrightarrow vulnerability(\mathbb{S}_1) \geq vulnerability(\mathbb{S}_2)$ holds with $vulnerability()$ being the function defined in Equation (10). In other words, validity requires that more vulnerable systems score lower values on the security metric.*
- Accuracy: *If $\mathscr{S}(m)$ is the specification of security metric $m()$, then $m()$ is accurate if the difference between its return values and the true values according to $\mathscr{S}(m)$ are small on average. Inaccuracies arise when metrics use approximative procedures, either because the exact values cannot be calculated or they are too expensive to calculate.*
- Precision: *For any given IT system $\mathbb{S}$, repeated measurements under the conditions and constraints specified by the metric $m()$ always return the same value $m(\mathbb{S})$. (A weaker form of this requirement is that the variance of repeat measurements has to be small.)*

This is a good place to contrast our work to prevalent positions found in the literature on security and security metrics.

- "Security risk" [12, 16, 17, 18] is *not* a security metric (by our definition) because it mixes loss amounts and loss probabilities into one measure. The loss probability, in turn, depends on a system's intrinsic

security as well as the threats it faces. This mixing of concepts conflicts with the theory developed in this paper, which clearly distinguishes between *peripheral factors* such as threats or monetary system values and the *intrinsic security* of a system. Security metrics are measures of the latter. By the same argument, "Return on Security Investment (ROSI)" [5], is *not* a security metric because it is a "mixed measure" that combines a system's intrinsic security, its threat environment, and its monetary value.

- We suggest introducing the notion of *compliance metrics* so they can be clearly distinguished from security metrics. For example, "percentage of organizational units that use symmetric cryptography" [5] is not a security metric as it is not valid: 100% adoption of symmetric cryptography may be insufficient if keys cannot be protected properly, and 0% adoption of symmetric cryptography does not necessarily imply poor security as asymmetric cryptography may have been used instead. The metric therefore has virtually no security relevance. On the other hand side, for organizations that have the compliance objective [19] to "use no symmetric cryptography" or to "use only symmetric cryptography", the above metric is clearly a useful way of tracking and measuring their performance against this objective. We therefore (informally) define: A *compliance metric* is an accurate and precise function that measures the extent to which an IT system meets its compliance objectives. Note that compliance is easier to measure than security because it assesses the extent to which checklists of specific methods and procedures are implemented. Security metrics, by contrast, assess the *result* of such methods and procedures, which is an abstract system property.

- We further propose introducing the concept of *threat metrics*. Equation (1) clearly shows that *security* and *threat* are two different and independent concepts. Given that there has been some excellent research on quantifying the threat environment [20, 21], we feel that the term *threat metric* is needed to reflect the importance and independence of this concept. Informally, a threat metric is an accurate and precise measure of the maliciousness of the environment.

3 Classification of Security Metrics

Section 2 pointed out that a metric is defined by its measurement process, which is a detailed and unambiguous operational description of how a metric is calculated. In particular, the measurement process has to define the *inputs* of a metric and the *algorithm* used to process those inputs. According to Definition 2.2, the input of a security metric always is the IT system $\mathbb{S}$ for which the metric is calculated. While this is formally correct, actual security metrics only observe *aspects* of real IT systems. This is a matter of practicality because real IT systems are too large and complex to be analyzed in their entirety. Thus, when referring to the *input* of a security metric we mean those aspects of IT systems that a security metric actually observes and analyzes.

When classifying security metrics, their inputs are of particular importance, even more so than the algorithms used to process these inputs. This is because "garbage in, garbage out" also holds here: When unsuitable inputs are chosen, no matter how sophisticated an algorithm is used to process them, the outcomes will be flawed metrics of low validity, accuracy, and/or precision. It is for this reason that we chose the inputs of metrics as the basis of our classification scheme.

To structure our analysis of the inputs used in security metrics, we will develop a simple model of IT systems and the processes used to build and operate them. In this model, three *operational processes* are involved: First, the *Design & Build* process builds IT systems; second, the *Operate* process runs them; and third, the *Maintain & Update* process changes IT systems during their run-time. Each of the three operational processes produces an *output*: The "Design & Build" process produces "IT Systems" as output. The "Operate" process, which executes the "IT System", produces "Services" as output (which are consumed by "Consume" processes). Finally, the "Maintain & Update" process produces a "Changed System", which the "Operate" process then executes.

Figure 1 summarizes our model and shows the three operational processes (in grey) and their interactions. For the sake of completeness, the figure also shows two exogenous processes on the right-hand side. The functional and control logic that govern the operational processes is represented by the ”Logic” and ”Control” boxes.

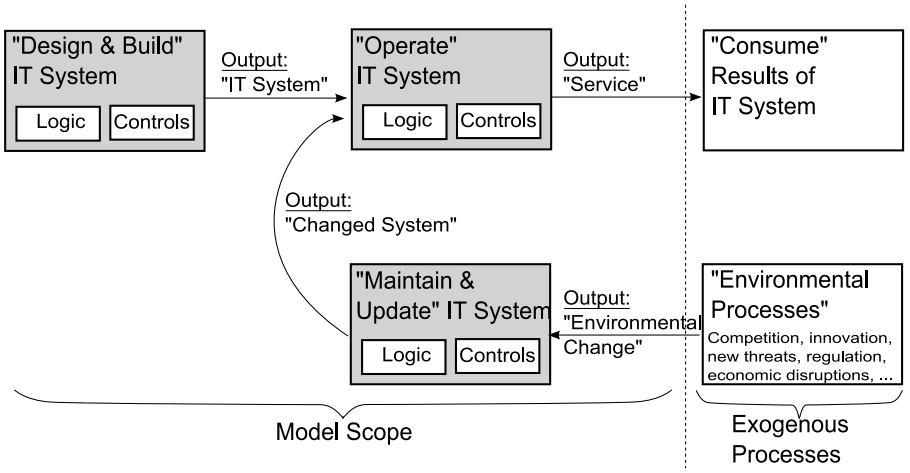

Figure 1: Simple Life-Cycle Model of IT Systems

Each of the three operational processes of Figure 1 can serve as input to security metrics, and we will consider each of the three processes in turn. To begin with, the ”Operate” process, which executes the IT system that the ”Design & Build” process built, gives rise to three types of input:

- The first type of input is the static definition of the IT system, i.e. the binaries, source code, and configuration files that are executed by the ”Operate” process.
- The second type of input is the execution trace of the running IT system, consisting of all system-internal run-time events.
- The third type of input are the results that the running IT system delivers as output to its consumers.

The difference between the second and the third input type is that the second type includes all system-internal run-time events, while the third type only ”sees” the IT system from the outside – just like a consumer would. Also note that the first input type is used by metrics that treat IT systems as static objects while the second or third input types are used by metrics that observe and analyze run-time behavior. For example, configuration checkers use the first type of input while vulnerability scanners use the second input type.

We now turn to identifying four additional input types that have been used by security metrics. Note that a good ”Design & Build” process results in more secure IT systems, and the same is true for the ”Maintain & Update” process. Accordingly, security metrics can be defined to analyze the ”Design & Build” and ”Maintain & Update” processes. Each of these processes, gives rise to two input types: First, the static process definition of either process can serve as input to security metrics; and second, the run-time event trace of executing either process can serve as input to security metrics. The output produced by either process is the IT system executed by the ”Operate” process, which gives rise to the three input types discussed above.

Figure 2 summarizes our discussion and shows the resulting classification of security metrics. As stated earlier, the reason for basing the classification on the types of inputs used by the security metrics is because the input types have a particularly large impact on the validity, accuracy, and precision of security metrics. This classification is further supported by the fact that it is useful in classifying prior work in the field of security metrics. Section 4.1 further explores these points.

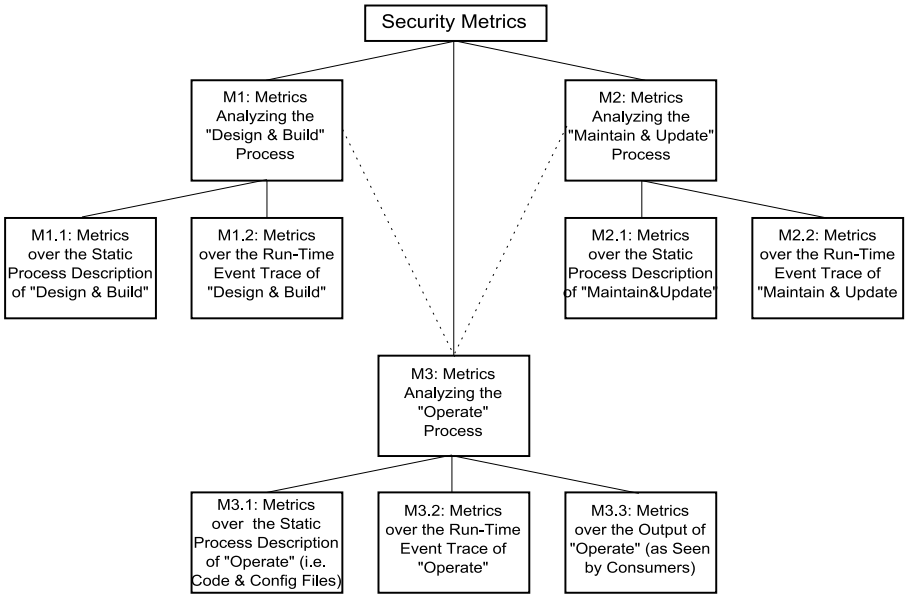

Figure 2: Classification of Security Metrics by their Input Types

Previous work on classification schemes [22, 23, 24] had a tendency to mix security metrics (as defined in this paper) with other security-related quantities such as risk, threat, or assurance. The above classification avoids such imprecisions. Also, our work uses a new classification criteria, namely the input type. Previous work, by contrast, has taken other classification criteria such as who the end user of a metric is or the purpose of the artifact measured .

4 Applications

This section presents two applications of the theory developed in this paper: First, Section 4.1 reviews prior work in security metrics and identifies pitfalls to avoid. Second, Section 4.2 shows how our theory can help managers make better information security management decisions.

4.1 Review and Discussion of Prior Work

In our review of prior art, we will interpret the term ”security metric” in the loosest sense possible and also include work that does not satisfy Definition 2.2. That way, we can offer a broad overview of the kind of work that has been done in the field. Our review follows the classification of Section 3 and proceeds by input type:

Category M1: In software engineering, metrics of category M1 are known as *maturity levels* [25]. In the present context, they measure the ”maturity” of the security engineering process. With a few exceptions (most notably [26, 27]), little research has been focused on this category of security metrics. However, given that the software development process has a major impact on the security of the resulting software products [26, 28], we consider this to be an opportunity for future work.

Category M2: Metrics of sub-category M2.1 analyze the the static description of the ”Maintain & Update” process, while metrics of sub-category M2.2 analyze its run-time behavior. Either way, metrics in category M2 include the ”frequency of audit record review”, the ”mean-time to mitigate vulnerabilities”, the ”mean-time to patch”, the ”percent of changes with security review”, the ”number of formally approved new accounts relative to all new accounts”, and the ”percentage of personnel that have receive security

training each year" [2, 4, 29], to name some representative examples. The ISO/IEC 27004 standard [10] also emphasizes the importance of measuring the effectiveness of information security management systems.

Category M3.1: Metrics of category M3.1 include: *(i)* Measures of source code vulnerabilities [3, 30, 31, 32]; *(ii)* the *attack surface* [33, 34] consisting of the weighted number of exposed (but not necessarily vulnerable) system resources such as open sockets; *(iii)* various vulnerability indices that weight current or historic vulnerabilities by their severity [35, 36]; or indices that weight the factors that affect the vulnerability of a system such as the patching frequency or password aging policy [37]; *(iv)* the probability of a given security goal being violated in a formal model (such as a Markov model), which is built to represent real IT systems [38, 39, 40, 41, 42, 43, 44, 45]; *(v)* metrics that seek to quantify erroneous application settings and configuration mistakes [46]; *(vi)* numeric measures of the extent to which a system implements security best practices such as "Percentage of systems that default to a fail-safe state in the event of errors" [5], "percentage of users with access to shared accounts" [4], or "strength of authentication mechanism" [47].

Category M3.2: Metrics of category M3.2 analyze run-time events pertaining to security incidents such as the "number of incidents", the "mean time between security incidents", or the "number of anomalous or unexpected events" [2, 47]. Metrics derived from vulnerability scanners or penetration testing [48] also fall into this category.

Category M3.3: Metrics in this category measure violations of Service Level Agreements between two parties [49, 50, 51].

Let us critically appraise some of this prior art. We have emphasized repeatedly that a security metric has to be valid, accurate, and precise as defined in Definition 2.2. Unfortunately, for many of the above "metrics", these properties were never shown to hold. In many cases, the validity, accuracy, and precision even seem unlikely. As such they are better characterized as "security statistics", rather than "security metrics". This criticism is founded in the following basic challenges that authors of metrics have to overcome:

- Security metrics calculated over run-time events (e.g. "number of incidents" [2]) are affected by changes in the threat environment. This makes them inherently imprecise. For example, two subsequent measurements of the "number of incidents" in the *same* IT system may result in very *different* results, just because the threat environment has changed between the measurements. As such, "number of incidents" is not a precise metric of an IT system's security. More generally, all metrics in the categories M1.2, M2.2, and M2.3 are affected by environmental influences that can destroy precision.

- For processes that are manual (as opposed to automated) metrics that analyze the static process description (categories M1.1 and M2.1) tend to be invalid. This is because manual processes are subject to execution errors and omissions. These errors and omissions lead to a situation where the *measured properties* of the "on-paper process" differ from the properties of the actual process that is being executed. This is the hallmark of an invalid metric.

- For some metrics, accuracy is a challenge. For example, if the specification of a metric is "number of incidents" then accuracy becomes a problem in practice because only an estimated 10% of incidents are detected [52]. Thus, whatever operational procedure we define to estimate the "number of incidents" it will be fairly inaccurate. One may be tempted to fix this problem by changing the metric to "number of incidents found by our security team", but this metric creates imprecisions (what if a key team member leaves?) and it raises questions about the validity (what do the team's findings really say about the system's security?).

- It is dangerous to take the validity of a metric for granted or to justify it merely with plausible arguments. For example, both the "average time to apply a patch, measured across all systems" and the "percentage of systems patched within one week" are both plausible metrics. However, the latter metric has stronger bearing on the security of a system [6]. Similarly, many proposed metrics operate on such a micro-level that it is unclear how relevant they are for the security of the *overall* IT system. In other words, their validity is questionable. The "percentage of users with access to shared accounts" is an example of such a micro-level metric. A different view of this point is that micro-level metrics are difficult to interpret and utilize because there are so many of them [29].

While these points identify important pitfalls that authors of metrics should avoid, there is ultimately only one way to create a valid, accurate, and precise security metric, and that is to experimentally proof these properties. In fact, it is our belief that the next big breakthrough in security metrics will be the development of empirical validation tools, frameworks, and test beds. Recent research has also recognized the importance of experimental validation [53, 29, 32].

4.2 Making Investment Decisions

Section 2 pointed out that the purpose of a security metric is to help managers make better information security decisions. In this section, we use the theory developed in this paper to address the following management problems:

1. How to minimize security-related losses;
2. How to split one's security budget among multiple IT systems, all of which require protection;
3. How much to spend on security.

This list is obviously incomplete, and there are other important information security decisions that we cannot consider here. In particular, the above decisions are from the perspective of the software consumer, rather than the software vendor. The consumer's and vendor's views are different because software vendors must decide how much security to build into their products, but in general, they do not bear the cost of security failures.

Management Problem #1 Minimizing security-related losses is a difficult problem because losses depend on how suitable an IT system's security is relative to its threat environment and its value. It is the interplay of those three factors (value, threat, and security) that determine the actual losses. Among these factors, the "value" of an IT system is the factor that security managers are most capable of controlling. We therefore recommend that companies keep the data they need for competitive advantage and purge the rest. This technique has been called *data minimization* [54] and it is the most predictable way to minimize losses.

On the surface, it may seem that security managers are equally capable of minimizing losses by controlling the security of their IT systems. This, however, is only partially true. The problem is that our ability to understand the security of a system is very limited and our ability to assess if said security is sufficient given the threat environment is even weaker. As a consequence, it is currently not possible to manage one's security in a manner that is proven to minimize losses.

As a practical work-around, some authors advocate observing where losses occur and improving security in these places [7]. For example, when today's security incidence data is categorized according to the number of records exposed, the majority of incidents can be seen to relate to lost or stolen equipment or media [7]. A good heuristic is to focus one's security improvement efforts on these areas where losses concentrate. The problem with this approach is that it assumes that security incidents are observable, which is generally not the case [52]. Moreover, it opens up the system to "Black Swans", i.e. rare but potentially catastrophic events, which nobody

prepares for because they are not in the historic loss data. In a variant of this heuristic, CVSS scores have been advocated as a means to prioritize security efforts [13].

Another best-practices approach for minimizing security-related losses is given by the ISO 27001 definition of an information security management system [55]. The general philosophy of this standard is that security follows if management teams implement the security management processes defined in the standard.

Beyond the above guidance, the sobering reality is that we do not understand how to manage IT security in a manner that minimizes losses.

Management Problem #2 Here we consider the problem of two IT systems $\mathbb{S}_1$ and $\mathbb{S}_2$, and a manager who has to allocate her security budget of B Euros between these two systems. We will propose a solution to this problem, based on the following assumptions:

(1) We assume that the security manager wants to minimize the total expected losses, $EL_{total} = EL(\mathbb{S}_1) + EL(\mathbb{S}_2) = value_1 \times E(T1_m) \times vulnerability(\mathbb{S}_1) + value_2 \times E(T2_m) \times vulnerability(\mathbb{S}_2)$.

(2) We assume that the threat faced by a system is proportional to its value, i.e. $E(T1_m) = \alpha \times value_1$ and $E(T2_m) = \alpha \times value_2$ for some positive real number α. This assumption models the fact that criminals are attracted by valuable targets, because "that is where the money is" [56].

(3) We assume that the productivity of security spending is equal for both system $\mathbb{S}_1$ and system $\mathbb{S}_2$. Moreover, we assume that spending b Euros on the security of either system will drive the system's vulnerability score to equal $vulnerability(\mathbb{S}_{1/2}) = 1/(\beta \times b)$, for a constant $\beta \in \mathbb{R}^+$. This function models the fact that vulnerability goes to infinity if no money is spent on security. As more money is spent on security, the vulnerability of a system decreases and asymptotically approaches zero. The vulnerability score approaches but never reaches zero because of diminishing returns on security spending. Further, the constant β models the productivity of security spending: The smaller β the more money one has to spend to achieve a desired level of vulnerability.

With these assumptions we can calculate the total expected losses when b Euros are spent on the security of system $\mathbb{S}_1$ and $B-b$ Euros are spent on the security of system $\mathbb{S}_2$:

$$
\begin{aligned}
EL_{total}(b) &= EL(\mathbb{S}_1) + EL(\mathbb{S}_2) && (11)\\
&= value_1 \times E(T1_m) \times vulnerability(\mathbb{S}_1) + \\
&\quad value_2 \times E(T2_m) \times vulnerability(\mathbb{S}_2) && (12)\\
&= \alpha \times value_1^2 \times 1/(\beta \times b) + \\
&\quad \alpha \times value_2^2 \times 1/(\beta \times (B-b)) && (13)\\
&= \frac{\alpha \times {value_1}^2}{\beta} \times \left(\frac{1}{b} + \left(\frac{value_2}{value_1} \right)^2 \times \frac{1}{B-b} \right) && (14)
\end{aligned}
$$

As stated in assumption (1), the goal is to minimize the total expected loss. The requirements $EL'_{total}(b) = 0$ and $EL''_{total}(b) > 0$ yield the following solution, in which we use $\varphi := value_2/value_1$ for the relative value of the two systems:

$$b_{opt} = \frac{B}{1+\varphi} \tag{15}$$

As expected, $b_{opt} = 0.5 \times B$ when both systems are equally valuable, i.e. when $\varphi = 1$. If system $\mathbb{S}_1$ is much more valuable than system $\mathbb{S}_2$, then $\varphi \to 0$ and $b_{opt} \to B$. This result is expected, as well. Finally, if system $\mathbb{S}_2$

is two times as valuable as system $\mathbb{S}_1$, then the optimum amount of money to be spent on the security of $\mathbb{S}_1$ is one third of the budget (i.e. $b_{opt} = B/3$).

Given the modelling assumptions that went into deriving Equation (15), it is obviously not the only possible answer to the question of budget allocation. It is, however, a well-reasoned guideline that managers can use to inform budget allocation decisions. Moreover, Equation (15) is important because *it uses two quantity (the budget B and the relative systems value φ) that can be estimated easily and accurately*. In particular, there is no need to make guesses about the threat environment, and it can be argues that the relative value φ of two systems is much easier to estimate than the absolute value of a system [53]. The simplicity and usability of Equation (15) is therefore an important result of our work. Future research will assess the impact of our modelling assumptions on this result.

Management Problem #3 The last management problem we address is how much money to spend on security. To solve this problem, we search the spending level that minimizes the *expected cost of security*. Please note that alternatively, we could have minimized the *worst-case cost of security*, which focuses on rare but potentially devastating security incidents. Thus, just as above, there is no single "right" answer to the question of how much to spend on security. However, by optimizing the expected cost of security, we will be able to derive a simple and practicable rule that managers can use to guide their investment decisions.

The expected cost of security is the sum of the expected losses EL and the security budget B that is spent on avoiding such losses. Using Equation (1) for expected losses and the above assumption that $vulnerability(\mathbb{S}) = 1/(\beta \times B)$, $\beta \in \mathbb{R}^+$ is the vulnerability score resulting from a security budget B, we obtain the expected cost of security $EC(B)$:

$$\begin{aligned} EC(B) &= EL(\mathbb{S}) + B & (16)\\ &= value \times E(T_m) \times \frac{1}{\beta \times B} + B & (17) \end{aligned}$$

This function has its minimum where the conditions $EC'(B) = 0$ and $EC''(B) > 0$ hold. Resolving for B yields the following optimum budget:

$$B_{opt} = \sqrt{\frac{value \times E(T_m)}{\beta}} \tag{18}$$

An IT security manager who wants to use this equation to calculate the optimum budget would proceed as follows: While operating the current IT system, the manager would determine the security budget B_{actual} and the expected loss EL_{actual} he observes. The expected loss EL_{actual} is the average loss experienced over a couple of successive time periods; it is an empiric value which can be determined *accurately* because every well-run IT organization tracks the losses it incurs from security incidents. As such, EL_{actual} can be calculated by adding and averaging historic losses. According to Equation (1) and $vulnerability(\mathbb{S}) = 1/(\beta \times B)$, $\beta \in \mathbb{R}^+$, the following holds:

$$EL_{actual} = value \times E(T_m) \times \frac{1}{\beta \times B_{actual}} \tag{19}$$

In this equation, $value$ and $E(T_m)$ are the same constants as in Equation (18). They represent the value of the IT system and the expected malice of the system's threat environment, respectively. By rearranging the terms in Equation (19), it follows that

$$\sqrt{\frac{value \times E(T_m)}{\beta}} = \sqrt{EL_{actual} \times B_{actual}} \tag{20}$$

Note that the left-hand side of Equation (20) equals the right-hand side of Equation (18). This leads us to conclude:

$$B_{opt} = \sqrt{EL_{actual} \times B_{actual}} \tag{21}$$

In other words, the optimum amount of security spending is the geometric mean of today's actual spending B_{actual} and the expected loss EL_{actual} that occurs at this spending level. This is a very simple rule, which requires only measurements that are reasonably easy to obtain in any real-world context. Moreover, Equation (21) implies that when security spending B_{actual} exceeds expected losses E_{actual} then too much money is spent on security. This result is consistent with the result obtained in the seminal paper by Gordon and Loeb on the economics of IT security spending [14]. Our future work will explore the robustness of Equation (21) with respect to the assumptions made in its derivation.

5 Summary and Conclusion

This paper has defined that security metrics are accurate and precise functions whose return values are inversely related to the vulnerability of the measured system. We justified this definition by showing that it is consistent with the field of software quality metrics. Moreover, the definition follows from the formula for expected losses, which we also derived in this paper. We further showed how security metrics are different from general security statistics, risk metrics, threat metrics, and compliance metrics. As such, our work makes an important contribution to simplifying and clarifying the terminology in the field of security.

Next, we introduced a new classification of security metrics. This classification – unlike previous ones – is based on the input data analyzed by security metrics. The decision to use the input data as the basis of a new classification was made because the input data has a particularly large influence on the validity, accuracy, and precision of security metrics. The paper elaborates on this point, reviews related work, and uncovers important pitfalls that authors of security metrics must be aware of.

Lastly, we turned to the use of metrics. The purpose of security metrics is to help management make better information security management decisions. As such, we considered three security management problems and approached them using the theory developed in this paper. Most importantly, we derive the following two results:

1. The optimum security budget is $B_{opt} = \sqrt{EL_{actual} \times B_{actual}}$ with B_{actual} being the current security budget and EL_{actual} being the empirically observed expected loss for this budget.

2. The optimum split of one's security budget B between two systems $\mathbb{S}_1$ and $\mathbb{S}_2$ that need protection is to spend $B/(1+\varphi)$ on the security of system $\mathbb{S}_1$ and the rest on system $\mathbb{S}_2$. The parameter φ in this equation is the value of system $\mathbb{S}_2$ divided by the value of system $\mathbb{S}_1$.

These are important results because they show that careful modelling can yield practical answers to very difficult problems. In our future work, we will explore the robustness of these results with respect to the modelling assumptions made.

Acknowledgement

The research leading to these results has received funding from the European Communitys Seventh Framework Programme (FP7/2007-2013) under grant agreement n FP7-216917.

References

[1] S. Berinato, "A few good information security metrics," http://www.csoonline.com/, July 2005.

[2] T. C. for Internet Security (CIS), "The cis security metrics," CIS, Tech. Rep., May 2009, http://www.cisecurity.org/securitymetrics.html.

[3] Andy Chou and Junfeng Yang and Benjamin Chelf and Seth Hallem and Dawson Engler, "An Empirical Study of Operating Systems Errors," in *ACM Symposium on Operating System Design and Implementation*, 2001.

[4] E. Cew, M. Swanson, K. Stine, N. Bartol, A. Brown, and W. Robinson, "Performance measurement guide for information security," National Institute for Standards and Technology, Tech. Rep. NIST SP 800-55, July 2008.

[5] D. S. Herrmann, *Complete Guide to Security and Privacy Metrics*. Auerbach Publications, 2007.

[6] Verizon, "2009 data breach investigations report," Verizon Business, Tech. Rep., 2009.

[7] A. Shostack and A. Stewart, *The New School of Information Security*. Addison-Wesley, 2008.

[8] in *Proceedings of the ACSA Workshop on Information Security System Scoring and Ranking*, Ronda Henning, Ed., 2001.

[9] V. Basili, G. Caldiera, H. Rombach, and R. van Solingen, "Goal question metric (gqm) approach," in *Encyclopedia of Software Engineering*. John Wiley & Sons, Hoboken, New Jersey, 2002.

[10] ISO/IEC 27004, *Information Technology — Security Techniques — Information Security Management — Measurement*. International Organization for Standardization, Geneva, Switzerland, 2009.

[11] S. H. Kan, *Metrics and Models in Software Quality Engineering*, 2nd ed. Pearson Education, 2003.

[12] T. O. Group, "Risk taxonomy," The Open Group, Technical Standard ISBN 1-931624-77-1, 2009.

[13] P. Mell, K. Scarfone, and S. Romanosky, "Cvss a complete guide to the common vulnerability scoring system," Forum of Incident Response and Security Teams (FIRST), Tech. Rep., June 2007, http://www.first.org/cvss/cvss-guide.html.

[14] Lawrence A. Gordon and Martin P. Loeb, "The economics of information security investment," *ACM Transactions on Information and System Security*, vol. 5, no. 4, pp. 438–457, 2002.

[15] L. M. Laird and M. C. Brennan, *Software Measurement and Estimation: A Practical Approach*. IEEE Press, 2006.

[16] T. R. Peltier, *Information Security Risk Analysis*, 2nd ed. Auerback, 2005.

[17] B. C. Soh and T. S. Dillon and P. County , "Quantitative Risk Assessment of Computer Virus Attacks on Computer Networks," *Computer Networks and ISDN Systems*, vol. 27, no. 10, pp. 1447–1456, 1995.

[18] Ashish Gehani and Gershon Kedem, "RheoStat: Real-Time Risk Management," in *Proceeding of the 7th International Symposium on Recent Advantages in Intrusion Detection (RAID)*, E. Jonsson, A. Valdes, and M. Almgren, Eds. Springer Verlag, 2004, pp. 296–314.

[19] Klaus Julisch, "Security Compliance: The Next Frontier in Security Research," in *Proceedings of 2008 the New Security Paradigms Workshop*, 2008.

[20] E. Jonsson and T. Olovsson, "A Quantitative Model of the Security Intrusion Process Based on Attacker Behavior," *IEEE Transactions on Software Engineering*, vol. 23, no. 4, pp. 1–1 :235, 1997.

[21] C. Kanich, C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage, "Spamanalytics: An empirical analysis of spam marketing conversion," *Communications of the ACM*, 2009.

[22] Reijo Savola, "Towards a Security Metric Taxonomy for the Information and Communication Technology Industry," in *Proceedings of 2007 International Conference on Software Engineering Advances (ICSEA)*, 2007.

[23] Rayford B. Vaughn and Ronda Henning and Ambareen Siraj, "Information Assurance Measures and Metrics - State of Practice and Proposed Taxonomy," in *Proceedings of 36th Hawaii International Conference on System Sciences (HICSS)*, 2002.

[24] Nabil Seddigh and Peter Pieda and Ashraf Matrawy and Biswajit Nandy and Ioannis Lambadaris and Adam Hatfield, "Current Trends and Advances in Information Assurance Metrics," in *Proceedings of the Second Annual Conference on Privacy, Security and Trust*, 2004, pp. 197–205.

[25] Margaret K. Kulpa and Kent A. Johnson, *Interpreting the CMMI: A Process Improvement Approach*. Auerbach Publications, 2008.

[26] Michael Howard and Steve Lipner, *The Security Development Lifecycle, SDL: A Process for Developing Demonstrably More Secure Software*. Microsoft Press, 2006.

[27] ISSEA, "Systems Security Engineering Capability Maturity Model Project, Appraisal Method," International Systems Security Engineering Association (ISSEA), Tech. Rep., January 1999, http://www.sse-cmm.org/.

[28] Michael Howard and David LeBlanc, *Writing Secure Code*, 2nd ed. Microsoft Press, 2003.

[29] Yolanta Beres and Marco Casassa Mont and Jonathan Griffin and Simon Shiu, "Using Security Metrics Coupled with Predictive Modeling and Simulation to Assess Security Processes," Hewlett Packard, Tech. Rep. HPL-2009-142, 2009.

[30] Istehad Chowdhury and Brian Chan and Mohammad Zulkernine, "Security Metrics for Source Code Structures," in *Proceedings of the Fourth International Workshop on Software Engineering for Secure Systems*, 2008, pp. 57–64.

[31] Stephan Neuhaus and Thomas Zimmermann and Christian Holler and Andreas Zeller, "Predicting Vulnerable Software Components," in *Procceedings of the 14th ACM Conference on Computer and Communications Security (CCS)*, 2007, p. 529540.

[32] Stephan Neuhaus and Thomas Zimmermann, "The Beauty and the Beast: Vulnerabilities in Red Hats Packages," in *Proceedings of the 2009 USENIX Annual Technical Conference*, 2009.

[33] Michael Howard and Jon Pincus and Jeanette M. Wright, "Measuring Relative Attack Surfaces," in *Proceedings of Workshop on Advanced Developments in Software and Systems Security*, 2003.

[34] Pratyusa K. Manadhata and Yuecel Karabulut and Jeannette M. Wing, "Measuring the Attack Surfaces of SAP Business Applications," in *IEEE International Symposium on Software Reliability Engineering (ISSRE)*, 2008.

[35] Muhammad Abedin and Syeda Nessa and Ehab Al-Shaer and Latifur Khan, "Vulnerability Analysis For Evaluating Quality of Protection of Security Policies," in *Proceedings of the 2nd ACM workshop on Quality of protection)*, 2006.

[36] Mohammad Salim Ahmed and Ehab Al-Shaer and Latifur Khan, "A Novel Quantitative Approach for Measuring Network Security," in *Proceedings of IEEE Infocom Miniconference*, 2008.

[37] Jim Alves-Foss and Salvador Barbosa, "Assessing Computer Security Vulnerability," *ACM SIGOPS Operating Systems Review*, vol. 29, no. 3, pp. 3–13, 1995.

[38] David M. Nicol and William H. Sanders and Kishor S. Trivedi, "Model-Based Evaluation: From Dependability to Security," *IEEE Transactions on Dependable and Secure Computing*, vol. 1, no. 1, pp. 48–65, 2004.

[39] Oleg Sheyner and Joshua Haines and Somesh Jha, "Automated Generation and Analysis of Attack Graphs," in *IEEE Symposium on Security and Privacy*, 2002, pp. 273–284.

[40] Somesh Jha and Oleg Sheyner and Jeannette M. Wing, "Minimization and Reliability Analysis of Attack Graphs," Carnegie Mellon University (CMU), Tech. Rep. CMU-CS0-2-109, 2002, http://www-2.cs.cmu.edu/afs/cs.cmu.edu/project/calder/www/tr02-109.html.

[41] Bharat B. Madan and Katerina Goseva-Popstojanova and Kalyanaraman Vaidyanathan and Kishor S. Trivedi, "A Method for Modeling and Quantifying the Security Attributes of Intrusion Tolerant Systems," *Performance Evaluation*, vol. 56, no. 1-4, pp. 167–186, 2004.

[42] R. Ritchey and P. Ammann, "Using Model Checking to Analyze Network Vulnerabilities," in *Proceedings of the IEEE Symposium on Security and Privacy*, 2000, pp. 156–165.

[43] R. Ortalo and Y. Deswarte and M. Kaaniche, "Experiments with Quantitative Evaluation Tools for Monitoring Operational Security," *IEEE Transactions on Software Engineering*, vol. 25, no. 5, pp. 633–650, 1999.

[44] B. Littlewood and S. Brocklehurst and N. Fenton and P. Mellor and S. Page and D. Wright, "Towards Operational Measures of Computer Security," *Journal of Computer Security*, vol. 2, pp. 211–229, 1993.

[45] Laura P. Swiler and Cynthia Phillips and D. Ellis and S. Chakerian, "Computer-Attack Graph Generation Tool," in *Proceedings of the DARPA Information Survivability Conference and Exposition*, vol. 2, 2001, pp. 307–321.

[46] Hazem Hamed and Ehab Al-Shaer and Will Marrero, "Modelling and Verification of IPSec and VPN Security Policies," in *Proceedings of the 13th IEEE International Conference on Network Protocols*, 2005.

[47] Thomas Heyman and Riccardo Scandariato and Christophe Huygens and Wouter Joosen, "Using Security Patterns to Combine Security Metrics," in *Proceedings of Third International Conference on Availability, Reliability, and Security)*, 2008.

[48] Rayford B. Vaughn and Ronda Henning and Ambareen Siraj, "Information Assurance Measures and Metrics - State of Practice and Proposed Taxonomy," in *36th Annual Hawaii International Conference on System Sciences*, 2003.

[49] Guenter Karjoth and Birgit Pfitzmann and Matthias Schunter and Michael Waidner, "Service-oriented Assurance Comprehensive Security by Explicit Assurances," in *Quality of Protection Security Measurements and Metrics*, 2006.

[50] Ronda R. Henning, "Security Service Level Agreements: Quantifiable Security for the Enterprise?" in *Proceedings of the 1999 Workshop on New Security Paradigms (NSPW)*, 1999.

[51] Y. Karabulut, F. Kerschbaum, F. Massacci, P. Robinson, and A. Yautsiukhin, "Security and trust in it business outsourcing: a manifesto," *Electronic Notes in Theoretical Compututer Science*, vol. 179, pp. 47–58, 2007.

[52] S. Bosworth and M.E. Kabay and E. Whyne, *Computer Security Handbook*, 5th ed. Wiley, New York, 2009, ch. Understanding Studies and Surveys of Computer Crime.

[53] J. J. Ryan and D. J. Ryan, "Performance metrics for information security risk management," *IEEE Security & Privacy*, vol. 6, no. 5, pp. 38–44, 2008.

[54] G. Skinner and E. Chang, "A conceptual framework for information privacy and security in collaborative environments," *nternational Journal of Computer Science and Network Security*, vol. 6, no. 2B, 2006.

[55] ISO/IEC 27001, *Information Technology — Security Techniques — Information Security Management Systems – Requirements*. International Organization for Standardization, Geneva, Switzerland, 2005.

[56] Willie Sutton and Edward Linn, *Where the Money Was: The Memoirs of a Bank Robber*. Broadway, 2004.