2019 IEEE 43rd Annual Computer Software and Applications Conference (COMPSAC)

Designing Good Security Metrics

George O. M. Yee
Computer Research Lab, Aptusinnova Inc.
Dept. of Systems and Computer Engineering, Carleton University
Ottawa, Canada
e-mail: gmyee@sce.carleton.ca

***Abstract*—This paper begins with an introduction to security metrics, describing the need for security metrics, followed by a discussion of the nature of security metrics, including the challenges found with some security metrics used in the past. The paper then discusses what makes a good security metric and proposes a rigorous step-by-step method that can be applied to design good security metrics, and to test existing security metrics to see if they are good metrics. Application examples are included to illustrate the method.**

Keywords—security metrics, weaknesses, designing, testing, good security metrics

I. INTRODUCTION

We live in a precarious world, in which barely a day goes by without headlines appearing about the latest data breaches, the most recent webs sites brought down by DDoS (Distributed Denial of Service) attacks, or the latest victim held hostage by a ransomware attack. In response, organizations have poured large sums of money into various security countermeasures (e.g. firewalls, biometrics, encryption, security training) and re-structured systems and workflows in attempts to resist these attacks. However, the return on these efforts or the resultant increase in the level of security has been largely unknown. The organization may then have to deal with the following dilemmas:

- Have we invested enough funds in securing our systems to be "safe" from attack?
- Will our software changes to improve security be effective?
- Are our work flows and processes sufficiently secure?
- How will our security be impacted by the addition of that third-party software component?
- How can we be accountable for keeping our systems at the required level of security if we are unable to determine our level of security?

These dilemmas can be resolved by having a way to assess the level of security of a computer system or an organization. In other words, properly defined, effective *security metrics* can help. Security metrics have been defined and are being used. However, many of them are poorly defined for their intended purposes. They are far from giving the results needed to assess security and are therefore ineffective. For example, a traditional security metric is the number of viruses detected and eliminated at a firewall. This metric has been used to gauge the effectiveness of a firewall at filtering out viruses, which impacts the organization's level of security. Unfortunately, this metric fails its mission because it says nothing about the viruses that were not detected and got through. If 50 viruses were detected and eliminated but 100 got through, basing the firewall's effectiveness on the 50 viruses that were detected would falsely inflate the effectiveness and the level of security.

The security metric mentioned in the previous paragraph is neither meaningful nor effective. This leads to the following questions: Can we define security metrics that are meaningful and effective? What are the conditions that security metrics must satisfy in order to be considered good metrics? We do not claim that all existing security metrics are bad. Rather, we wish to shed some light into how to define good security metrics.

The objectives of this paper are to a) introduce the reader to the problems and challenges exemplified by several typical traditional security metrics, b) propose a method that can be used to design a good security metric or to test whether a particular security metric passes in terms of the characteristics that a good security metric must have, and c) illustrate the method by applying it to design a good security metric, and to test existing security metrics.

The rest of this paper is organized as follows. Section II provides a brief introduction to security metrics and describes the challenges of some traditional security metrics. Section III discusses the requirements of a good security metric, and proposes a method that can not only be used to design a good security metric, but can also be used to test existing security metrics to see if they are good metrics. Section IV illustrates the method with application examples. Section V discusses related work, and Section VI presents conclusions and future research.

II. SECURITY METRICS

A. Definition and Purpose of Security Metrics

Computer systems must deliver their services in a manner that is secure and respects user privacy. Researchers and practitioners have made many efforts to develop security

978-1-7281-2607-4/19/$31.00 ©2019 IEEE
DOI 10.1109/COMPSAC.2019.10270

measures that aim to help the systems' owners to make informed decisions about a system's design, the selection of security controls, and the efficiency of security operations. However, the development of standardized security metrics has been a difficult challenge, and efforts have only met with partial success (more details below).

A *component* of the security level of an organization can be anything that plays a part in determining the security level. Example components are software vulnerabilities, security policies, security controls in place, staff awareness of social engineering, and so on. In reality, it is very difficult if not impossible to identify all the components of the security level for a typical organization (more details below). Components motivate the following definition of a security metric for this work:

DEFINITION: A *security metric* is a numerical value or set of numerical values that measures some component or components of the security level of "something" at a particular point in time, where that "something" could be a computer system, an organization, a software product, an access control system, and so on. A security metric may also be a formula that evaluates to the numerical value or values.

Since a security metric only measures some components of the security level, it only represents the security level, and does not measure the security level, unless of course it measures all components of the security level, which is practically next to impossible. For instance, consider that it is next to impossible to count all the vulnerabilities of a software system, where these vulnerabilities would be components of the system's security level. Also, it is important to note that a security metric represents the security level at a particular point in time, since the security level can change over time. Examples of security metrics include the frequency by which a user changes his or her banking password, the number of times a computer operating system is patched for security vulnerabilities in a year, and the amount of funds spent by an organization in a year to purchase and deploy security countermeasures.

Security metrics have the following objectives: a) provide a quantitative and objective basis for security operations, b) support decision making, e.g., are more security controls needed?, c) support software quality since quality includes security, d) support the reliable maintenance of security operations, e.g. how often do users need to change their passwords?, e) support the incremental improvement of the ability of software to resist attacks. This list of objectives is only partial but it serves to illustrate the usefulness of security metrics. In addition, the following goals of security metrics, quoted from [2], describe similar objectives but from new perspectives:

- "Strategic support – Assessments of security properties can be used to aid different kinds of decision making, such as program planning, resource allocation, and product and service selection.
- Quality assurance – Security metrics can be used during the software development lifecycle to eliminate vulnerabilities, particularly during code production, by performing functions such as measuring adherence to secure coding standards, identifying likely vulnearbilities that may exist, and tracking and analyzing security flaws that are eventually discovered.
- Tactical oversight – Monitoring and reporting of the security status or posture of an IT system can be carried out to determine compliance with security requirements (e.g., policy, procedures, and regulations), gauge the effectiveness of security controls and manage risk, provide a basis for trend analysis, and identify specific needs for improvement".

Clearly, security metrics play a very important role in today's computing systems.

B. Challenges with Definition and Application

By definition, a security metric has to measure some components of the security level. But which components? Consider the following example security metrics (shown in italics) [1] that are common but problematic. The metric *number of computer viruses or malware detected* is similar to the above firewall example, and is intended to measure the effectiveness of anti-malware controls. However, it does not include the malware that was undetected and got through, which would definitely change the perceived effectiveness! The purpose of the *number of security incidents* metric is to measure the effectiveness of security events monitoring in order to gauge the security level of the organization. However, it does not incorporate the thresholds at which the incidents are triggered (which is needed to understand those events that were not considered security events), nor does it include their causes, e.g., security incidents may be triggered due to flaws in work processes. Another often-used security metric is *time spent on a security-related task*, e.g., software patching, security incident investigation. This may be useful for project management, in order to make sure that there is sufficient time to complete the project, but it is practically useless as an indicator of security. To see this, consider that spending more time does not necessarily result in better security. For example, the extra time may have been due to inefficient procedures or work processes. The *business cost of a security incident* is yet another unreliable security metric. The purpose of this metric is to gauge the quality of the organization's security practices by looking at financial losses to the organization. However, the security incident may have been due to something other than poor security practice, such as the loss of valuable data due to an accident. Alternatively, the incident management was so good that the costs were kept to a minimum. This metric may measure the effectiveness of incident response in terms of minimizing business cost, but it is not a good measure of the quality of the organization's security practices, since it cannot distinguish if the costs were due to poor security practices.

III. Designing Good Security Metrics

How can we design a good security metric? What characteristics should good security metrics possess? Some examples of these characteristics are:

- Measure organizationally *meaningful* things, *reproducible*, *objective* and *unbiased*, measure over time some type of *progression* toward a goal [1].
- Accurate, precise, valid, and correct [3].
- Consistently measured, cheap to gather, expressed as a cardinal number or percentage and using at least one unit of measure, contextually specific [6].

Some traditional security metrics fail to have one or more of these characteristics, and were selected haphazardly or opportunistically, in the sense that whatever measures were readily available were chosen and applied. Many of the traditional metrics discussed above are misleading in that they fail to incorporate essential elements that are logically needed in order to fulfill their intended purposes.

A. Method for Designing Good Security Metrics (MDGSM)

Based on the above characteristics and the challenges presented in Sections II-B, we propose that security metrics be derived using the following steps:

1. **Definition:** Define the quantity to be measured. Check that this quantity is meaningful, objective, and unbiased as a measure of the component or components of the security level of "something", where that "something" could be the organization, the organization's computer system, or even a software product. Check also that this quantity can be obtained with undue hardship or costs. If the quantity passes all these checks, proceed to Step 2. Otherwise, repeat this step to obtain a new quantity. Example quantities are: *percentage of secured vulnerabilities over all known vulnerabilities*, *number of software security patches issued in a month*.

2. **Sufficiency:** Check that the quantity is a sufficient measure of the component or components of the security level (as in necessary and sufficient conditions for something to be true, see [4]). It is enough to check sufficiency since there are usually many ways to measure a component, so necessity will not apply in most cases. It is helpful to carry out this check by asking and answering the questions in Table I. For the quantity to be sufficient, the answers to questions 1, 2, and 3 must be "yes", "yes", and "no" respectively. If the quantity is found to be sufficient, proceed to Step 3. Otherwise, repeat from Step 1 to obtain a new quantity. For example, the quantity *time spent on a security-related task* is not a sufficient estimator since spending more time does not mean that the security level will be consistently higher (or lower) – the extra time could be simply due to a bad tool or an inefficient work process. Thus, the answer is "no" to question 1. Since this answer must be "yes" for sufficiency, this quantity is not sufficient.

3. **Divisibility:** If the quantity is not divisible into other constituent quantities, or is not expressible mathematically in terms of other constituent quantities, proceed to step 4. Otherwise, formulate a mathematical expression that equates the quantity to the constituent quantities, and proceed to Step 4. For example, the quantity *number of software security patches issued in a month* is not further divisible, whereas the quantity *outstanding vulnerabilities after threat analysis each month* may be divided into and equated to *the number of non-secured vulnerabilities from last month* plus *the number of new vulnerabilities found during threat analysis*.

TABLE I. Questions for Determining Sufficiency

No.	Question
1	If the quantity goes up, do you believe that the security level consistently goes up (or down)?
2	Does the quantity have a direct impact on the security level?
3	Are there any aspects missing from the definition of the quantity that are needed for it be effective as a measure of the component or components of the security level?

4. **Progression:** Check that the quantity has the "progression property", that when evaluated over a sufficiently large time period, from past to future, the quantity progresses to an acceptable pre-defined level that corresponds to an acceptable or maximal security level. If the quantity has this property, proceed to Step 5. Otherwise, repeat from Step 1 to obtain a new quantity. For example, in the case of *number of software security patches issued in a month,* suppose that this metric is evaluated at the first of the month for the last month. The value can only be a positive whole number or zero. Thus, over a sufficiently large number of months in which there is at least one nonzero number of patches issued, there is a corresponding increase in the security level of the software toward some level. The security level of the software increases with each patch issued until at some point, no new patches are issued, at which point the security level of the software is maximal (but not necessarily maximized since there may still be undiscovered security bugs).

5. **Reproducibility:** Check that the quantity is reproducible or verifiable by third-party verifiers. This means that the latter may evaluate the quantity or arrive at its value using the same inputs or procedure and obtain the same result. If the quantity is reproducible or verifiable, stop. The quantity is now considered a good security metric. Otherwise, repeat from Step 1 to obtain a new quantity. For example, if the quantity is *number of software security patches issued in a month*, a third-party verifier would add up the software security patches issued for a particular month, and find the same number as the organization that is using the metric. If the quantity is *outstanding vulnerabilities after threat analysis each month*, which we know is equated to *the number of non-secured vulnerabilities from last month* plus *the number of new vulnerabilities found during threat analysis*, the third-party verifier would do the latter addition and verify

that the total is the same as obtained by the organization using the metric.

MDGSM may also be used to test existing security metrics to see if they are good security metrics (see examples in Section IV).

B. MDGSM Application Notes

In Step 1, we mentioned that the quantity should be an objective and unbiased measure of the component(s) of the security level. Note that more than one quantity may be used to measure the component(s). So, after defining one quantity to measure the component(s), we may use MDGSM again and define another quantity that also measures the component(s). An example of a quantity that is a biased measure is the *number of viruses detected and eliminated at a firewall* that we mentioned in Section I. Correspondingly, this quantity results in biasing upward the security level of the computer system in which the firewall is installed, because it does not incorporate the number of viruses that were not detected by the firewall and got through.

In Step 2, it may be difficult to answer question 3. The answer may not be obvious. It may be helpful to have the "big picture" in mind. So, for example, in the firewall case, the big picture reminds us about the viruses that were not detected. It may also help to ask the opinion of others when answering the question. Two or three heads are usually better than one.

Step 3 is about breaking up the quantity into its constituents, if applicable, for ease of evaluation. Constituents are usually simpler and easier to deal with than the original whole. For example, the quantity *cost of a security incident* may be broken down into the *cost for investigating the security incident* plus the *cost of remediating the security incident*, which allows the original quantity to be more easily evaluated.

The progression property in Step 4 is required in order to be able to answer questions such as when does the organization know that it is "safe" and how can the cost of new security controls be justified. Each such question can be answered with the help of a security metric that measures a progression over time that represents increasing security levels toward some goal or reaching a maximal level. Using such a metric, it is "safe" if the corresponding security level reaches an agreed, predefined threshold. The cost of new security controls will be justified by referring to the resulting increased levels of security.

The reproducibility requirement in Step 5 is necessary so that the value of the security metric can be verified by an independent third-party. It is a fundamental requirement of any "fact" or "evidence" of significant importance, that it be verifiable by others. Otherwise, how do we know that a mistake has not occurred? Verifiers can be independent consultants or government agencies, e.g. the Department of Homeland Security in the United States. Having reproducibility doesn't mean that a security metric has always to be verified – actual verification may be needed only in extreme and dangerous situations.

C. MDGSM Strengths and Weaknesses

Some strengths of MDGSM are a) it provides a rigorous step-by-step checklist for designing a new security metric that has all the characteristics, namely meaningful, objective, unbiased, sufficient, progressive, and reproducible, that are required of a good security metric (characteristics based on the literature and ability to meet the challenges mentioned in Section II), b) it can be used to test existing security metrics to see if they possess the characteristics mentioned in a), c) it designs metrics that provide answers to what management needs to know [1], namely an assessment of the security level of the organization, when is the organization "safe"?, how can the cost of more security controls be justified?, and so on.

A potential weakness of MDGSM is that steps 1 and 2 may be challenging to carry out since they require some security and computer systems expertise as well as logical thinking. For example, it may be hard to come up with a new security metric fulfilling Step 1. As well, it may be difficult to answer question 3 in Table I since it requires one to think through whether or not the metric will be effective in how it will be applied. To partially alleviate this, we recommend that a team consisting of security and computer systems knowledgeable people carry out MDGSM. The team can then use brainstorming to tackle steps 1 and 2, where more heads are better than one. The team doesn't need to be large – one consisting of 3 or 4 people should suffice.

IV. Applications

In this section, we apply MDGSM to design a new security metric and use it to test the metrics mentioned in Section II-B.

A. Designing a New Security Metric

Peter has been hired as a security consultant for Company A's computer system. Company A has spent tens of thousands of dollars on securing computer system vulnerabilities and now wants to know if it needs to secure even more vulnerabilities in order to be "safe". Peter will be working as part of a team that includes the computer system's operations manager. Peter is knowledgeable in both computer systems and security. The team applies MDGSM as follows:

STEP 1: Definition. The team believes that the security of the computer system is directly related to the number of secured vulnerabilities in the system: the higher this number, the higher the security, and the lower this number, the lower the security. Consequently, the quantity or component chosen to represent the system's security is the *percentage of secured vulnerabilities over all known vulnerabilities* (both secured and unsecured) or PSV (for Percentage of Secured Vulnerabilities). PSV is clearly meaningful for assessing whether or not more security measures are needed. It is objective since secured vulnerabilities relate directly to the security of the system. It is unbiased since there is nothing that could cause the value of PSV to be overstated or understated. Finally, one can

evaluate the quantity by doing a vulnerability or threat analysis [5], in order to discover vulnerabilities, and then noting how many of these vulnerabilities have been secured, without undue hardship or cost. The team considers the quantity as having passed all the checks in Step 1 and proceeds to Step 2.

STEP 2: Sufficiency. The team answers the questions in Table I. The first question asks if the security would consistently go up (or down) if the quantity goes up. Clearly if PSV goes up, the security consistently goes up, so the answer is "yes". The second question asks if the quantity has a direct impact on the security. The answer is again "yes" since securing more vulnerabilities means that there are less targets for attackers to attack, directly impacting the security. Finally, the third question asks if the quantity is missing any components or aspects that are needed for it to be effective. The answer here is "no" since as far as the team could tell, the quantity has all it needs for assessing if more security controls are needed. The answers to the three questions conform to the answers required for sufficiency. The team declares the quantity sufficient and proceeds to Step 3.

STEP 3: Divisibility. The team observes that the quantity *percentage of secured vulnerabilities over all known vulnerabilities* has a mathematical expression in terms of the number of secured vulnerabilities p, and the number of unsecured vulnerabilities q, where $p + q$ is the number of all known vulnerabilities (both secured and unsecured). The team therefore assigns PSV as follows:

$$\text{PSV} = (100 \times p) / (p + q) \quad \text{if } p + q > 0 \qquad (1)$$
$$= 100 \quad \text{if } p + q = 0 \qquad (2)$$

The team finds that this expression has some nice properties. If $q = 0$ in (1), all vulnerabilities would be secured and PSV = 100, as expected. If $p = 0$ in (1), no vulnerabilities would be secured and PSV = 0, as expected. If $p + q = 0$ in (2), there are no vulnerabilities, and PSV = 100, again as expected. The team proceeds to Step 4.

STEP 4: Progression. Suppose that vulnerabilities are determined (through a threat analysis) and PSV is re-calculated at regular time intervals, e.g., monthly. Suppose also that Company A's management has agreed on a PSV goal of 95%, at which the computer system is considered "safe", i.e. management is willing to live with the risks arising from remaining non-secured vulnerabilities. With this goal in mind, management will want to secure vulnerabilities at each opportunity until PSV = 95. This doesn't mean that PSV will increase monotonically, since it is possible that a particular threat analysis identifies so many new vulnerabilities that PSV is actually lower than when it was last calculated. However, PSV will eventually reach 95%, given that management wants to secure new vulnerabilities until this goal is reached, which is all we mean by having the progression property. The team now considers the quantity as having passed Step 4 and proceeds to Step 5.

STEP 5: Reproducibility. Given the expression for PSV in Step 3, anyone will calculate the same value for PSV given the same values for p and q. Thus, the team considers PSV as being reproducible.

The team has successfully completed MDGSM and concludes that their newly created security metric, PSV, is a good security metric that can answer the question of whether or not more vulnerabilities need to be secured (as outlined in Step 4).

B. Testing Existing Security Metrics

We apply MDGSM to the metrics mentioned in Section II-B, except for *time spent on a security-related task*, which was already shown to have failed Step 2 in Section III-A. Since we already know that these are bad metrics, showing that they fail MDGSM will also be a check on MDGSM itself.

METRIC: *number of computer viruses or malware detected*

MDGSM: STEP 1: Definition. This metric is biased toward overstating the effectiveness of anti-malware controls since it does not account for the viruses or malware that were not detected. Thus, this metric fails Step 1 and is not a good security metric.

METRIC: *number of security incidents*

MDGSM: STEP 1: Definition. The consideration here would be that the higher the number of security incidents reported, the more effective the security events monitoring, and the lower the security level. It appears to be meaningful, objective, and unbiased. This metric also would not cause undue hardship to collect. So, let's say that this metric passes Step 1.

MDGSM: STEP 2: Sufficiency. We consider the questions in Table I. For question 1, we do believe that if this metric goes up, the security level will consistently go down. For question 2, the metric does have a direct impact on the security level. So, for the first 2 questions, the answers are "yes" and "yes". For question 3, whether or not there are aspects missing from the definition of the metric that are needed for it to be effective, we have to say "yes". For example, how do we know that a security incident occurred due to a lack of security? The incident may have been caused by an accident, and have nothing to do with security. Since the answer to question 3 is "yes", the metric fails Step 2, meaning that it is not a good security metric.

METRIC: *business cost of a security incident*

MDGSM: STEP 1: Definition. This metric is used to gauge the security level in terms of the quality of the organization's security practice – the higher the cost of a security incident, the lower the quality of security practice, and the lower the security level. However, the cost of a security incident may be due to factors that have nothing to do with security practice, e.g. the high value of the asset lost. Thus, this metric is not meaningful and is biased. This metric fails Step 1 and is not a good security metric.

V. Related Work

To our knowledge, MDGSM is unique and there is no other work that presents a similar method. Other works such as [1], [3], and [6] identify some of the characteristics that good security metrics should have, but do not present a comprehensive method for designing good security metrics.

The rest of this section presents related work that have to do with the nature and development of security metrics. Chapin and Akridge [1] describe what is wrong with traditional security metrics, giving characteristics of good metrics. It also discusses security maturity models giving examples of the models. Jansen [2] gives an overview of security measurement and proposes some possible research areas, such as formal models of security measurement and artificial intelligence assessment techniques. Herrmann [3] discusses the definition of a security metric and what makes a good security metric. Swartz [4] provides a background on sufficient conditions which we use in this work. Salter et al. [5] provide a systematic procedure for threat analysis. Jaquith [6] describes security metrics for enterprise applications. It applies security metrics broadly, not only to computer systems but to all sorts of enterprise processes. Hayden [7] is similar to [6] in its focus on the enterprise. It covers security metrics in terms of effectiveness, implementation, operations, compliance, costs, people, organizations, and includes four case studies. Rodes et al. [8] propose the use of security arguments to facilitate meaningful and comprehensive security metrics, based on the observation that a security argument provides comprehensive documentation of all evidence and rationales for justifying the value of the metric for a computer system. Pereira and Santos [9] give some suggestions for the definition of meaningful metrics for security controls, in order to provide decision makers with actionable information for managing their organizational assets and ensuring their day-to-day operations. Flater [10] is the first part of a two-part series on the problems of security metrics and their solutions. Flater [11] is the second part that provides solutions for the problems in the first part. The first part uses measurement theory to quickly determine that many metrics are unfit for the purposes for which they are used. The second part presents solutions by way of six principles to help avoid the problems and move forward using sound measurement. Flater's goal is similar to ours in providing an approach for better security metrics but their method appears to be more theoretical. Sanders [12] gives an excellent account of security metrics in terms of origin, categorization, challenges, and the path forward. The latter includes a discussion on the appropriateness of security metrics, methods for estimating metrics, and the need for a comprehensive security argument methodology that can relate business and technical security metrics to one another. Finally, Yee [13] gives an introduction and literature review for security metrics, also touching on the scientific basis for security metrics.

VI. Conclusions and Future Research

Security metrics provide a quantitative basis for security operations, providing actionable information for security decision makers. They should be applicable to measuring security improvements over time, which is needed to show if a series of new investments in security controls results in better security. Unfortunately, some traditional security metrics have been selected haphazardly and have been problematic, missing aspects of security or the system that are relevant to the intended purpose of the metric. We have proposed MDGSM, a rigorous step-by-step method for designing good security metrics that tackles the problems of poorly defined metrics and fulfills the needs of organizations.

Future research includes making MDGSM easier to use, investigating possibly other characteristics of good security metrics and incorporating them into MDGSM, and looking into any issues identified by users of MDGSM. In addition, it would be very interesting to look into how security metrics may be obtained from Big Data. There are certainly large data sets of attack data available. How can these be used to reliably drive security metrics and reveal the security level of an organization, an industry, or a defense force?

References

[1] D. Chapin and S. Akridge, "How can security be measured?," Information Systems Control Journal, Vol. 2, 2005.

[2] W. Jansen, "Directions in Security Metrics Research," NIST NISTIR 7564, April 2009.

[3] D.S. Herrmann, "Complete Guide to Security and Privacy Metrics: Measuring Regulatory Compliance, Operational Resilience, and ROI," Auerbach Publications, Boca Raton, Fl., 2007.

[4] N. Swartz, "The Concepts of Necessary Conditions and Sufficient Conditions," Department of Philosophy, Simon Fraser University, 1997, retrieved January 25, 2019 from: https://www.sfu.ca/~swartz/conditions1.htm

[5] C. Salter, O. S. Saydjari, B. Schneier, and J. Wallner, "Toward A Secure System Engineering Methodology," Proceedings of the New Security Paradigms Workshop, pp. 2-10, 1998.

[6] A. Jaquith, "Security Metrics: Replacing Fear, Uncertainty, and Doubt," Addison-Wesley, 2007.

[7] L. Hayden, "IT Security Metrics: A Practical Framework for Measuring Security & Protecting Data," McGraw-Hill Osborne Media, June 2010.

[8] B. D. Rodes, J. C. Knight, and K. S. Wasson, "A security metric based on security arguments," Proceedings of the 5th International Workshop on Emerging Trends in Software Metrics (WETSoM 2014), pp. 66-72, June 2014.

[9] T. Pereira and H. Santos, "Security metrics to evaluate organizational IT security," Proceedings of the 8th International Conference on Theory and Practice of Electronic Governance (ICEGOV '14), pp. 500-501, October 2014.

[10] D. Flater, "Bad security metrics part 1: problems," IT Professional, pp. 64-68, January/February 2018.

[11] D. Flater, "Bad security metrics part 2: solutions," IT Professional, pp. 76-79, January/February 2018.

[12] W. H. Sanders, "Quantitative security metrics: unattainable holy grail or a vital breakthrough within our reach?," IEEE Security & Privacy, pp. 67-69, March/April 2014.

[13] G. Yee, "Security Metrics: An Introduction and Literature Review," in Computer and Information Security Handbook (Third Edition), Elsevier, pp. e57-e70, 2013.