A Survey on Systems Security Metrics

MARCUS PENDLETON and RICHARD GARCIA-LEBRON, The University of Texas at San Antonio
JIN-HEE CHO, US Army Research Laboratory
SHOUHUAI XU, The University of Texas at San Antonio

Security metrics have received significant attention. However, they have not been systematically explored based on the understanding of attack-defense interactions, which are affected by various factors, including the degree of system vulnerabilities, the power of system defense mechanisms, attack (or threat) severity, and situations a system at risk faces. This survey particularly focuses on how a system security state can evolve as an outcome of cyber attack-defense interactions. This survey concerns how to measure system-level security by proposing a security metrics framework based on the following four sub-metrics: (1) metrics of *system vulnerabilities*, (2) metrics of *defense power*, (3) metrics of *attack or threat severity*, and (4) metrics of *situations*. To investigate the relationships among these four sub-metrics, we propose a hierarchical ontology with four sub-ontologies corresponding to the four sub-metrics and discuss how they are related to each other. Using the four sub-metrics, we discuss the state-of-art existing security metrics and their advantages and disadvantages (or limitations) to obtain lessons and insight in order to achieve an ideal goal in developing security metrics. Finally, we discuss open research questions in the security metrics research domain and we suggest key factors to enhance security metrics from a system security perspective.

CCS Concepts: • **General and reference** → **Surveys and overviews**; *Metrics*; • **Security and privacy** → *Formal methods and theory of security*

Additional Key Words and Phrases: Security metrics, security measurement, security foundation, quantitative security

ACM Reference Format:
Marcus Pendleton, Richard Garcia-Lebron, Jin-Hee Cho, and Shouhuai Xu. 2016. A survey on systems security metrics. ACM Comput. Surv. 49, 4, Article 62 (December 2016), 35 pages.
DOI: http://dx.doi.org/10.1145/3005714

1. INTRODUCTION

How to develop security metrics has been identified as one of the hard problems by many key organizations including the US INFOSEC Research Council [Council 2007], the US National Science and Technology Council [Science and Council 2011], and the Science of Security Lablets [Nicol et al. 2015]. As one of the efforts to address this problem, the US National Institute of Standards and Technology (NIST) proposed security metrics in *implementation*, *effectiveness*, and *impact* [Chew et al. 2008]. The Center for Internet Security (CIS) defined 28 security metrics in *management*, *operational*, and *technical* aspects of a system [CIS 2010]. However, these efforts are exclusively geared towards

This research was supported in part by ARO Grant #W911NF-13-1-0141, NSF Grant #1111925, and the Assistant Secretary of Defense for Research and Engineering (ASD (R&E)).
Authors' addresses: M. Pendleton, R. Garcia-Lebron, and S. Xu, One UTSA Circle, San Antonio, TX 78249 USA; emails: marcusp46@yahoo.com, richard.garcialebron@gmail.com, shxu@cs.utsa.edu; J.-H. Cho, Computational and Information Sciences Directorate, U.S. Army Research Laboratory, Adelphi, MD 20783 USA; email: jinheechogwb@gmail.com.

DOI: http://dx.doi.org/10.1145/3005714

cyber defense administrations and operations. To the best of our knowledge, there has been little discussion on how security metrics may be used as parameters in security modeling. Further, the gaps or limitations between the state-of-the-art security metrics and the desirable ultimate goals, and how to fill these gaps have not been discussed in the literature. In this article, we make contributions to address these issues by proposing a security metric framework based on a system-level security perspective. We hope this effort can provide insightful guidelines and concrete directions for developing security metrics in this research domain.

1.1. Contributions

We propose a perspective for structuring systems security metrics, which is centered on measuring the dynamic systems security state (e.g., situation awareness for security decision-making) with three components: system vulnerabilities, attacks (or threats) severity, and power of defense mechanisms. This perspective allows us to easily tackle how to quantify security by seeking the mathematical abstractions or functions that can map multiple metrics associated with security into security state metrics. In this sense, our article makes the following contributions:

—We describe the following four sub-metrics based on key characteristics of attack-defense interactions: (1) metrics of *system vulnerabilities*, (2) metrics of *defense strength*, (3) metrics of *attack (or threat) severity*, and (4) metrics of *situation understanding*. These metrics reflect that attackers attempt to exploit system vulnerabilities despite the presence of defense mechanisms, and the strengths of defense mechanisms and attacks significantly impact a system's security state.
—We are particularly interested in the dynamic evolution of systems security over time, where a system needs to deal with attackers and employs various types of defense mechanisms [LeMay et al. 2011; Xu 2014a]. Investigating key factors affecting the dynamic evolution of systems security state provides the potential to enhance decision-making ability for cyber defense operations in highly dynamic, real-time situations.
—By categorizing many existing metrics into the four sub-metrics, we discuss their advantage and limitations and suggest future research directions. By investigating existing systems security metrics under these four metric areas, we propose a set of ideal metrics that can provide an insightful, forward-looking perspective to advance security metrics research.
—We discuss several gaps between the state of the art and the ultimate goals of security metrics. We leverage ontology-based methodologies to embrace comprehensive dimensions of systems security metrics, which enables systematically building a hierarchical structure of existing systems security metrics. This effort will substantially help define core systems security metrics where flooding terms of similar security metrics have been used in the field without any agreement or justification.

The rest of this article is organized as follows. Section 2 discusses existing survey papers on security metrics and makes clear the additional contributions of our article. Section 3 clarifies the scope and survey methodology in our article. For Sections 4–7, we discuss existing metrics categorized by the aforementioned four sub-metrics. Section 8 identifies the gaps between existing security metrics and ideal security metrics. Last, Section 9 concludes the article and suggests future research directions.

2. EXISTING SURVEY ON METRICS

Verendel [2009] conducted a survey on the literature published between 1981 and 2008 on the topic of quantitative security. The surveyed papers were categorized based on their (i) perspective (e.g., security goals, economic and risk, reliability), (ii) targets (e.g., economic incentives, framework for selecting quantification method, threats),

(iii) assumptions (e.g., independence, rationality, stationarity), and (iv) validation methods (e.g., hypothetical, empirical, simulation-based, theoretical). Verendel argued that "*quantified security is a weak hypothesis*" because the validity of results was not clear due to the lack of statistically sound evidence to either corroborate or contradict them. The main cause of this issue is the lack of security data to validate the methods.

Villarrubia et al. [2004] classified security metrics based on security goals (i.e., confidentiality, integrity, availability, and authentication), control areas (i.e., management, operations, and technical aspects), temporal dimension (i.e., prevention, detection, response, and recovery), or targeted users (e.g., technical personnel, decision-maker, external authority). Unlike Verendel [2009] and Villarrubia et al. [2004], our work uses a holistic perspective to develop system-wide security metrics based on the four key dimensions of attack-defense interactions.

To the best of our knowledge, this is the first work to systematically survey systems security metrics. Nevertheless, other related efforts have been made in the literature. For example, Landwehr et al. [1994] discussed a taxonomy of program flaws, Nicol et al. [2004] discussed model-based evaluation of dependability, Chandola et al. [2009] surveyed anomaly detection, Milenkoski et al. [2015] surveyed the evaluation of Intrusion Detection Systems (IDS), Roundy and Miller [2013] reviewed the problem of obfuscation, and Ugarte-Pedrero et al. [2015] surveyed packing tools. However, we still find that our survey article makes a unique contribution by using the proposed dynamic security metric framework dealing with key factors impacting the overall security of a system in multiple dimensions and discussing the existing metrics and their merit/demerit.

3. SURVEY APPROACH

3.1. Scope

To achieve an appropriate scope, we conduct our survey based on the following criteria:

—The articles used in this survey article were selected based on the types of security metrics because we focus on the definitions of security metrics used by each article. We treat analysis approaches as an orthogonal issue because a security metric may be analyzed via multiple approaches.
—We emphasized the latest security metrics. For example, we review the Common Vulnerability Scoring System (CVSS) v3.0 [FIRST 2015], which supersedes CVSS v2.0.
—We focused on investigating systems security metrics, excluding building-blocks security metrics (e.g., security metrics of cryptographic primitives). Our goal is to propose a metric framework applicable to many contexts, providing a generic framework of security metrics that can be the basis of a security metrics standard.

3.2. Terminology

This work uses the following terminologies:

—**Security metrics and measurements:** Although there is no standard definition of security metrics, security metrics have been considered to reflect quantitative security attributes based on certain scales (e.g., nominal, ordinal, interval) [Jansen 2009; Böhme and Freiling 2008]. Note that a *metric* refers to assigning a value to an object while *measurement* is the process of estimating attributes of an object.
—**Systems:** This work focuses on systems security metrics. In this context, we consider the two types of systems as follows:
(1) ***Enterprise systems*** refer to networked systems of multiple computers/devices, clouds, or even the entire cyberspace; and

(2) ***Computer systems*** represent individual computers/devices. We interchangeably use the terms node, device, or computer to refer to a single entity. Since an enterprise system consists of multiple entities, measuring security of individual entities can lead to measuring security of the enterprise system.

—**Attackers:** These are *attacking entities* representing computers or IP addresses from which cyber attacks are launched against other normal entities.

—**Incident:** It represents a successful attack (e.g., malware infection or data breach).

3.3. Scales of Metrics

Attributes of an object should be measured by certain scales. We discuss the five types of scales [Stevens 1946]. The five types of the scales formulate a hierarchy in terms of invariance of transformation where each scale has its associated permissible procedures for statistical inference and data analysis [Roberts 1979; Stevens 1946; Böhme and Freiling 2008]. The hierarchy may not be comprehensive for security metrics [Velleman and Wilkinson 1993] and the discussion of its validity is beyond the scope of this work. The six types of scales are as follows:

—**Nominal:** This scale is also known as a *categorical scale*, represented by a discrete unordered set. For example, an attacker's attack vector can be denoted as a set of exploits with respect to a set of software vulnerabilities. Measurement in a nominal scale can be compared using standard set operations. Nominal scales can be transformed between each other via some methods for one-to-one mapping.

—**Ordinal:** This scale is *a discrete ordered set*, in which we can use operators such as "greater than" to compare two measurements on the same ordinal scale. This scale can measure severity of vulnerabilities or attacks. Ordinal scales can be transformed between each other based on monotonic (i.e., order-preserving) one-to-one mapping methods. Ordinal scales are one step higher (i.e., more accurate in the measurement) than nominal scales in the hierarchy of scales because it can downgrade to nominal scale at the cost of information loss.

—**Interval:** This scale extends the ordinal scale such that the distance between adjacent points on the scale is constant. This *equal-distance property* allows the difference operator by which one can compare the difference between two points and the difference between another two points. One can define linear transformations between two interval scales, which preserve constant distance between two points on the same scale. An example is the measurement of temperature at the Fahrenheit or Celsius scale, between which there is a well-known transformation. Interval scales can be analyzed via statistical analysis, such as mean, variance, and even higher moments. Interval scales can be downgraded to ordinal (and therefore nominal) scales by relaxing the equal-distance condition on the scale.

—**Ratio:** This scale extends the interval scale by defining the origin (i.e., value of 0) in a natural fashion. Examples are length, mass, pressure, time, temperature on the Kelvin scale, or amount of money. Ratio scales can be operated with multiplication, logarithm, and finding roots. Transformations can be defined between ratio scales.

—**Absolute:** This measures things measurable only in a single method. Counting is a simple example of absolute scale. Clearly, any kind of statistical analysis can be applied to measurements in the absolute scale.

—**Distribution:** This scale is used to measure an attribute with *inherent uncertainty*. For instance, projected vulnerabilities should be measured as random variables, which can be compared based on their distributions [Da et al. 2014].

Discussion. Using a different scale gives a different level of accuracy, leading to a different degree of understanding in a given value as a metric. A higher level of scale

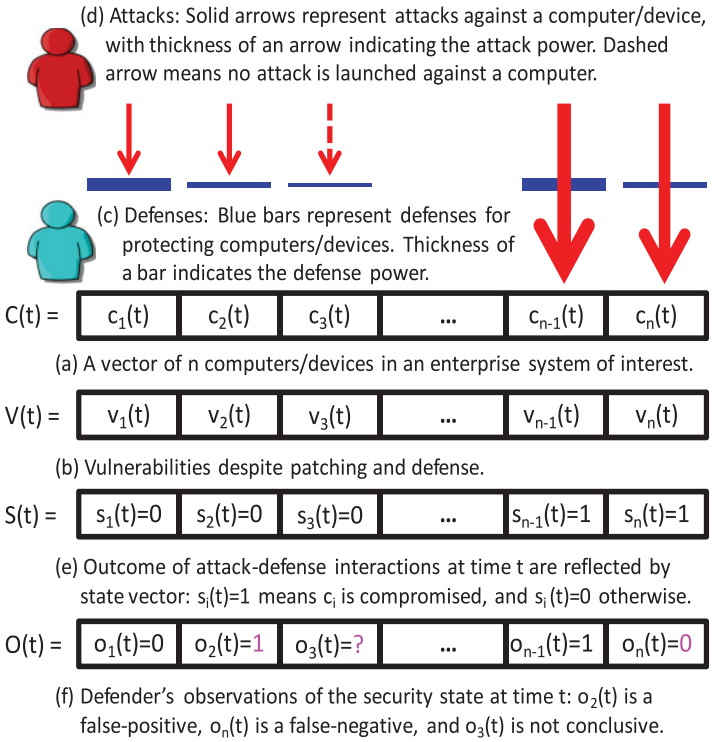

Fig. 1. Attack-defense interactions in an enterprise system at time t.

in the hierarchy gives higher accuracy that can be easily transformed to a low level of scale. Measurements using a higher level of scale permits richer statistical inference.

3.4. Methodology

The perspective of cyber attack-defense interactions is equally applicable to both *enterprise systems* and *computer systems*.

3.4.1. Enterprise Systems. Figure 1 illustrates a snapshot of an *enterprise system* under attack-defense interactions. At time t, the enterprise system consists of n entities (i.e., computers), denoted by the vector $C(t) = \{c_1(t), \ldots, c_n(t)\}$, where n can vary over time t. Each entity, $c_i(t)$, has a vector $v_i(t)$ of vulnerabilities, such as zero-day and/or some unpatched software vulnerabilities. *Red arrows* refer to *attacks* and *blue bars* represent *defenses*, where defense mechanisms are placed at both individual entities (e.g., anti-malware tools) and an enterprise system (e.g., firewalls). We use the *thickness* of red arrows and blue bars to illustrate the intuitive notion of *strength* (or power) of attack and defense. Some attacks penetrate through the defenses while others fail. The outcome of the attack-defense interaction at time t is reflected by a global security state vector $S(t) = \{s_1(t), \ldots, s_n(t)\}$, where $s_i(t) = 0$ means entity $c_i(t)$ is secure at time t and $s_i(t) = 1$ means entity $c_i(t)$ is compromised at time t. However, the defender's observation of the security state vector $S(t)$, denoted by $O(t) = \{o_1(t), \ldots, o_n(t)\}$, is imperfect due to detection errors or inherent noises.

3.4.2. Individual Systems. Figure 2 illustrates a *computer system* $c_i(t)$ at time t, representing an individual entity in the enterprise system discussed above. We use the same notations to represent the strength of defense and attack as used in an

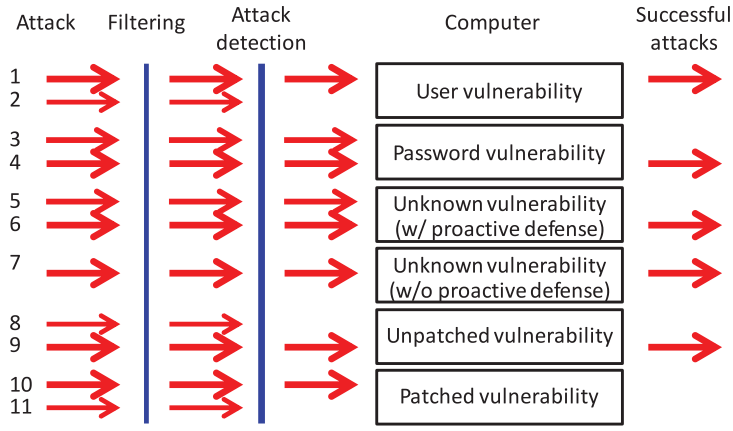

Fig. 2. Attack-defense interactions in a computer (or device), $c_i(t)$, in the enterprise system at time t.

enterprise system. Entity $c_i(t)$ may have a range of vulnerabilities (e.g., giving an error range), denoted by $v_i(t)$ in Figure 1. The vulnerabilities include a user's vulnerability (or susceptibility) to social-engineering attacks, the use of weak passwords, or software vulnerabilities. The defense for protecting entity $c_i(t)$ may include, for example, the use of some *filtering* mechanisms deployed at the enterprise system perimeter to block traffic from malicious or blacklisted IP addresses, the use of some *attack detection* mechanisms to detect and block attacks before they reach $c_i(t)$, and the use of proactive defense mechanisms (e.g., address space randomization) to mitigate vulnerabilities exploitable by attackers (i.e., exploitability).

An example scenario is illustrated in Figure 2 as follows. An attacker can perform a vector of 11 attacks, A_i for $i = 1, \ldots, 11$, some of which may successfully penetrate through the defense of $c_i(t)$. For instance, A_1 successfully compromises $c_i(t)$ because the user is lured into clicking a malicious URL; A_4 successfully compromises $c_i(t)$ because the user's password is correctly guessed; A_6 and A_7 successfully compromise $c_i(t)$ by exploiting a zero-day vulnerability, even under proactive defense mechanisms placed on $c_i(t)$; and A_9 successfully compromises $c_i(t)$ because the vulnerability is unpatched and the attack is neither filtered nor detected. Some defense mechanisms along with patches of potential vulnerabilities can block all of the other attacks.

3.4.3. Situation Understanding. The attack-defense interaction perspective leads to an intuitive formulation of security modeling. In principle, the outcome of attack-defense interaction is the evolution of the *situation* or *situation*(t), which is described by three classes of metrics including the aforementioned dynamic security state $S(t)$. That is, *situation*(t) can be represented as a mathematical function $f(\cdot)$:

$$\textit{situation}(t) = f(V(t), D(t), A(t)), \tag{1}$$

where $V(t)$ is a function of *vulnerabilities* at time t, $D(t)$ is a function of *defenses* at time t, and $A(t)$ is a function of *attacks* at time t. Note that $S(t)$ is a function of *security* at time t that can be captured as one of the factors captured in *situation*(t) as discussed in Section 7. $S(t)$ is naturally affected by $V(t)$, $D(t)$, and $A(t)$ as well. While still preliminary, initial progress has been made towards explicit representation of the various kinds of $f(\cdot)$'s corresponding to different kinds of attack-defense interactions [Xu 2014a]: preventive and reactive defense dynamics with or without accommodating the dependence between the relevant random variables [Li et al. 2011; Xu and Xu 2012; Xu et al. 2012b, 2014a, 2015a; Da et al. 2014]; proactive defense dynamics [Han et al.

Table I. Metrics and Measurement of Vulnerabilities, Defenses, Attacks, and Situations

	Vulnerability metrics	Defense metrics	Attack metrics	Situation metrics
Measurement	vulnerabilities of an enterprise system $V(t)$, or a computer system $v_i(t)$	strength of defense mechanisms $D(t)$	strength of attacks $A(t)$	*situation*(t), including system's security $S(t)$
Target	an enterprise system $C(t)$ or computer system $c_i(t)$	defense mechanisms $D(t)$ employed at $C(t)$ or $c_i(t)$	attacks $A(t)$ against $C(t)$ or $c_i(t)$	evolution of situation and environment
Types	users' vulnerabilities; interface-induced vulnerabilities; software vulnerabilities	preventive, reactive (e.g., detection), proactive (e.g., Moving-Target Defense (MTD)), and overall defense strength	zero-day attacks, targeted attack, botnets, malware spreading (e.g., infection rate), and evasion techniques	security state, security incidents, security investment
Reference Figures	Figures 1(a) and (b), Figure 2	Figure 1(c) and Figure 2	Figure 1(d) and Figure 2	Figure 1(e) and Figure 2

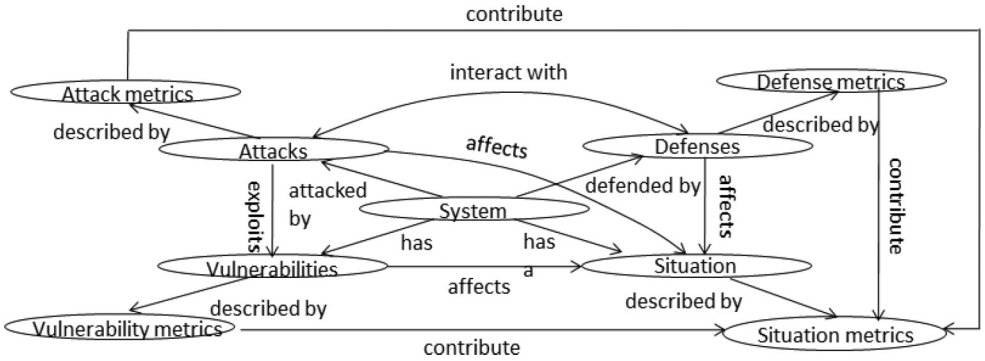

Fig. 3. A high-level ontology of systems security metrics consisting of four metrics.

2014]. The basic idea is to use some model parameters to represent the $V(t)$, $D(t)$, and $A(t)$ and then use some advanced models to describe the attack-defense interactions while accommodating these parameters. This allows one to characterize and predict what phenomena can happen under certain $f(\cdot)$'s and in certain parameter regimes. Moreover, Equation (1) offers a solution to compare the global effect of deploying two different sets of defenses, say, $D(t)$ and $D'(t)$, by comparing the corresponding outcome *situation*(t) and *situation*$'(t)$.

The attack-defense interaction accommodated in Equation (1) leads to natural taxonomies of systems security metrics that measure and represent vulnerabilities, defenses, attacks, and situations. Table I provides a summary of the four types of metrics for measuring the security state of a given system as illustrated in Figures 1 and 2.

3.4.4. Security Metric Ontology. A security metric ontology is illustrated based on the four sub-metrics in Figure 3. In this ontology, a system "has" *vulnerabilities*, can be "defended by" *defense mechanisms*, can be "attacked by" *attacks* (e.g., attack vectors), and is "described by" *situations* including its security state. Accordingly, vulnerabilities, defenses, attacks, and situations are respectively "described by" vulnerability metrics, defense metrics, attack metrics, and situations metrics. Each category of metrics are "measured by" a type of scales (e.g., nominal, ordinal, interval, etc.), which are omitted for simplicity in Figure 3. The relationship between the aspects are described as follows: Attacks "exploit" vulnerabilities; attacks and defense "interact with" each

other; vulnerabilities, attacks, and defenses all "affect" the situation. Corresponding to Equation (1), situation metrics are formulated by the contributions of vulnerability, attack, and defense metrics. Each sub-metric will have a corresponding sub-ontology that respectively corresponds to one of the following four sections.

4. VULNERABILITY METRICS

Vulnerability metrics refer to how to measure a level of system vulnerability. This section discusses how to measure the following vulnerabilities: *user vulnerabilities*, *interface-induced vulnerabilities*, and *software vulnerabilities*.

4.1. Metrics for Measuring User Vulnerabilities

We discuss two types of user vulnerabilities in terms of (i) users' cognitive bias (or error) and (ii) users' cognitive limitation. A user's susceptibility to phishing attacks or insider threat is a typical example of vulnerabilities exposed by the user's cognitive bias, while weak password often happens due to limited human memory. Weak password can be easily broken by attackers, breaking an authentication mechanism.

4.1.1. Metrics for Measuring Phishing Susceptibility. Typical metrics are false positives (FP) or false negatives (FN), where FP indicates the percentage of flagging genuine email as phishing email while FN captures the percentage of detecting a phishing email as a genuine email [Sheng et al. 2010]. Often human cognitive bias or personality traits can affect phishing susceptibility [Cho et al. 2016]. The phishing susceptibility can be measured in a *ratio scale*.

4.1.2. Metrics for Measuring Malware Susceptibility. Malware susceptibility is closely related to a user's online behavior. Users who often install many applications are more likely exposed to malware. In addition, if users visit many websites, then there is a higher vulnerability for malware infection [Levesque et al. 2013]. This malware susceptibility is also estimated in a *ratio scale*.

Discussion. It would be ideal if we could measure the intuitive metric of users' *susceptibility to attacks* such as social engineering. In order to estimate the overall vulnerability caused by a collection of individual vulnerabilities, it is critical to investigating the key factors a user relies on to make security decisions such as the user's characteristics, including disposition, personality, or biases [Neupane et al. 2015]. Although little work has investigated this issue [Howe et al. 2012; Sheng et al. 2010; Levesque et al. 2013], it is crucial to examining the relationships between an individual user's vulnerability based on his/her cognitive bias or personality traits in order to accurately assess vulnerability. Investigating users' personality traits, cognitive biases, and/or disposition and their impact on the users' susceptibility to attacks can be used to design defense mechanisms for mitigating the user's susceptibility to attacks.

4.1.3. Metrics for Measuring Password Vulnerabilities. Using a password is a common way to authenticate a user. While *entropy* is the most intuitive metric to measure the strength of a password, it is not a trivial task. Entropy is often estimated using heuristic rules such as the NIST rule, "*a bonus of up to 6 bits of entropy is added for an extensive dictionary check*" [Burr et al. 2006]. The estimated entropy of passwords is known to only offer a rough approximation of password weakness or strength. However, the estimated entropy cannot tell which passwords are easier to crack than others [Weir et al. 2010; Kelley et al. 2012]. Two password metrics have emerged recently, *password guessability* and *password meter*, which have been used in different contexts.

—**Password guessability metrics:** This metric aims to measure password vulnerability via the time or number of guesses that a password-cracking algorithm takes

to recover a password or a set of passwords [Ur et al. 2015, 2012; Kelley et al. 2012; DellAmico et al. 2010]. This metric is attractive because it is easy to use and can be used to compare password-cracking algorithms. This metric has two variants:

(1) **Statistical password guessability metric:** This metric measures the guessability of a set of passwords (rather than for individual passwords) by an "idealized" attacker with the assumption of a perfect guess [Bonneau 2012a, 2012b; Kelley et al. 2012]. This metric mostly uses a *ratio scale*.
(2) **Parameterized password guessability metric:** This metric measures the number of guesses an attacker needs to make via a particular cracking algorithm (i.e., a particular threat model) before recovering a particular password and training data [Weir et al. 2010; Bonneau 2012a; Kelley et al. 2012; Ur et al. 2015]. Note that using different password cracking algorithms leads to different results [Ur et al. 2015]. This means that when uncertainty towards attack behaviors is high, multiple cracking strategies need to be considered, which incurs high computational cost. This metric is easier to use than the former one. This metric uses an *absolute scale*.

—**Password meter metrics:** This metric is often used when one registers or updates a password [Castelluccia et al. 2012; Carnavalet and Mannan 2015]. *Password meters* use a more intuitive metric, called *password strength*. In principle, this metric measures the password strength via an estimated effort to guess the password, where the effort may be represented by a real number on a scale (e.g., probability of cracking), while considering factors such as the character set (e.g., special symbols are required or not), password length, whitespace permissibility, password entropy, and blacklisted passwords being prevented or not. One variant metric is *adaptive password strength*, which is used to improve the accuracy of strength estimation by computing the probability of occurrence of n-grams in passwords with respect to a n-gram database. Multiple scales can be used for measuring the strength of passwords, such as an *ordinal scale*.

Discussion. Although passwords are commonly used as an authentication mechanism due to their simplicity, building a set of rigorous and well accepted rules for quantifying password vulnerabilities remains as an important research problem. For example, it should be avoided that a password is diagnosed as strong by one metric but treated as weak by another metric. There should be an effort to derive an accurate result based on multiple diagnosed results from different metrics so a unified metric can represent a valid quality of a given password. It seems ideal if we can consistently and systematically measure the intuitive metric of *password strength*, which may be considered in the worst-case or average-case scenario and may be considered in the parameterized setting.

4.2. Metrics for Measuring Interface-Induced Vulnerabilities

The interface to access software from the outside world (i.e., service access points) offers potential opportunities for launching cyber attacks against the software. *Attack surface* metrics [Manadhata and Wing 2011] aim to measure the ways by which an attacker can compromise a targeted software. This metric was devised because many attacks against a software can be conducted by entering data from the environment (in which the software runs) to the software (e.g., buffer overrun) or by receiving data via interactions with the software. These attacks typically interact with the software by connecting to a channel (e.g., socket) or invoking a method (e.g., Application Programming Interface) offered by the software or sending/receiving data items to/from the software.

The methods, channels, and data items are considered as *resources*. An entry-point-exit-point framework [Manadhata and Wing 2011] was introduced to identify relevant

resources, where an entry point is a method for receiving data from the environment, and an exit point is a method for sending data to the environment. Each resource may or may not contribute to a software's attack surface. The contribution of a resource to the software's attack surface is the likelihood that the resource can be abused to launch attacks, where the likelihood depends on the privilege needed for using the resource. The potential damage caused by the abused resource is called *damage potential*. On the other hand, an attacker needs to make *efforts* to have privileges to use resources.

The potential damage of a method m can be defined as the number of calls to the method, implying that the higher the number of m method calls, the greater the potential damage. The effort to call a method m can be defined as the number of methods, N_m. The smaller the N_m, the greater the effort because in a sense the attacker has fewer options. For resource r, the ratio of the *potential damage* and *effort* is $R_{d,e}(r) \mapsto q$, where $q \in \mathbb{Q}$ and $\mathbb{Q}$ is the set of rational numbers. The quantity $R_{d,e}(r)$ reflects the contribution of resource r to attack surface. This idea is equally applicable to all kinds of resources, namely methods (or entry and exit points, denoted by $\mathcal{M}$), channels offered by the software (denoted by $\mathcal{C}$), and data items of the software (denoted by $\mathcal{I}$). Therefore, the *attack surface* of the software measures the subset of resources that can be abused to compromise the software, and it is defined as a tuple as follows:

$$\left\langle \sum_{m \in \mathcal{M}} R_{d,e}(m), \sum_{c \in \mathcal{C}} R_{d,e}(c), \sum_{i \in \mathcal{I}} R_{d,e}(i) \right\rangle. \tag{2}$$

This metric is specific to an environment, implying that one can compare the attack surfaces of two different versions of the same software running on the same environment. This metric is measured by an *absolute scale*.

Discussion. Attack surface is not defined dependent on software vulnerabilities. Reducing the attack surface (e.g., uninstalling a security software) does not necessarily improve security [Nayak et al. 2014]. Putting the interface-induced vulnerabilities into the context of Equation (1), a desirable metric would be the susceptibility of a software system to certain attacks according to a threat model, perhaps through how the exercise of attack surface is dependent on the features of attack surfaces, measured by a *ratio scale*. It is critical for predicting interface-induced system susceptibilities, namely the interfaces that will be exploited to launch attacks in the near future. This will allow the defender to employ tailored defenses targeting for these interfaces.

4.3. Metrics for Measuring Software Vulnerabilities

We divide software vulnerability metrics into three categories: temporal attributes of vulnerabilities, individual vulnerabilities, and collective vulnerabilities.

4.3.1. Metrics for Measuring Temporal Attributes of Vulnerabilities. These metrics can be further divided into two sub-categories of metrics for measuring: (i) the evolution of vulnerabilities and (ii) vulnerability lifetime [Al-Shaer et al. 2008; Ahmed et al. 2008].

—**Metrics for measuring the evolution of vulnerabilities** include:

(1) ***Historical vulnerability metric*** measures the degree that a system is vulnerable (i.e., frequency of vulnerabilities) in the past.
(2) ***Historically exploited vulnerability metric*** measures the number of vulnerabilities exploited in the past.
(3) ***Future vulnerability metric*** measures the number of vulnerabilities that will be discovered during a future period of time.
(4) ***Future exploited vulnerability metric*** measures the number of vulnerabilities that will be exploited during a future period of time.

(5) ***Tendency-to-be-exploited metric*** measures the tendency that a vulnerability may be exploited, where the "tendency" may be derived from information sources such as posts on Twitter before vulnerability disclosures [Sabottke et al. 2015].

This type of metrics may be used to prioritize vulnerabilities for patching (e.g., depending on their tendency-to-be-exploited). This metric can use a *ratio scale* an *absolute scale* or a *distribution scale* depending on an application's need.

—**Metrics for measuring vulnerability lifetime** measures how long it takes to patch a vulnerability since its disclosure, measured by an *absolute scale*. The three vulnerabilities can be considered to measure different vulnerability lifetimes as:

(1) ***Client-end vulnerabilities*** are often exploited to launch targeted attacks (e.g., spear-fishing) [Hardy et al. 2014; Marczak et al. 2014]. It often takes a long time to patch all infected devices or possibly infeasible to patch all [Frei and Kristensen 2010; Nappa et al. 2015].

(2) ***Server-end vulnerabilities*** are usually more rapidly patched than client-end vulnerabilities. Some example cases of patching server-end vulnerabilities can be found in Yilek et al. [2009] and Durumeric et al. [2014].

(3) ***Cloud-end vulnerabilities*** have been reported by many clouds (e.g., Amazon), but the patching process is rather slow [Zhang et al. 2014].

Discussion. Vulnerability lifetime is a critical factor affecting vulnerabilities $V(t)$ and the situation $situation(t)$, especially the global security state $S(t)$. This insight offers a new problem that appears to not have been investigated such as prioritizing vulnerabilities according to their impact on $situation(t)$, especially $S(t)$, according to Equation (1). We should analyze the risk of not patching an entity $c_i(t)$, which is affected by the severity of vulnerabilities that will be discussed in the following sections.

4.3.2. Metrics for Measuring Severity of Individual Software Vulnerabilities. The CVSS aims to measure software vulnerabilities with an emphasis on ranking them for prioritizing patching operations [FIRST 2015]. The Common Weakness Scoring System (CWSS) aims to prioritize software weaknesses for a different purpose [MITRE 2014]. In this work, we focus on CVSS version 3.0 [FIRST 2015] based on three categories of the following metrics: *base* metrics (including exploitability and impact), *temporal* metrics, and *environmental* metrics. Due to space constraints, we provide a brief summary in Table II.

Discussion. Despite the tremendous effort such as the CVSS, our understanding is still limited because some intuitive metrics still remain to be investigated, such as *patching priority* and *global damage* of vulnerabilities.

4.3.3. Metrics for Measuring Severity of a Collection of Vulnerabilities. Many attacks in the real world are performed in multiple steps and exploit multiple vulnerabilities. Correspondingly, there have been efforts on analyzing the consequences of multiple vulnerabilities, including *attack graphs* [Phillips and Swiler 1998; Sheyner et al. 2002; Ritchey and Ammann 2000; Jha et al. 2002; Ammann et al.; Albanese et al. 2012; Homer et al. 2013; Cheng et al. 2014], Bayesian Networks [Liu and Man 2005; Frigault and Wang 2008; Frigault et al. 2008], *attack trees* [Schneier 2000], and *privilege trees* [Dacier et al. 1996; Ortalo et al. 1999]. We survey two types of metrics measuring the severity of a collection of vulnerabilities particularly in attack graphs: *deterministic* and *probabilistic*.

Deterministic Severity Metrics

These metrics are mainly defined over attack graphs. Two sub-metrics are as follows:

—**Topology metrics** are for measuring how the topological properties of attack graphs (e.g., connectivity) affect network attacks. The following two metrics are commonly discussed under the topology metrics:

Table II. CVSS Version 3.0 Metrics for Measuring Severity of Individual Software Vulnerabilities, Where '*' Means That the Scales Are the Same as the Base Metrics

Metric	Definition	Scale
Base metrics: Exploitability		
Attack vector	Describes whether a vulnerability can be exploited remotely, adjacently, locally, or physically (i.e., attacker must have physical access to the computer)	Nominal
Attack complexity	Describes the conditions that must hold before an attacker can exploit the vulnerability, such as low or high	Ordinal
Privilege required	Describes the level of privileges that an attacker must have in order to exploit a vulnerability, such as none, low, or high	Ordinal
User interaction	Describes whether the exploitation of a vulnerability requires the participation of a user (other than the attacker), such as none or required.	Nominal
Authorization scope	Describes whether or not a vulnerability has an impact on resources beyond its privileges (e.g., sandbox or virtual machine), such as *unchanged* or *changed*	Nominal
Base metrics: Impact		
Impact metrics	The impact of a successful exploitation of a vulnerability in terms of confidentiality, integrity, and availability, such as *none, low, high*	Ordinal
Temporal metrics		
Exploit code maturity	The likelihood a vulnerability can be attacked based on the current exploitation techniques, such as *undefined, unproven, proof-of-concept, functional, or high*	Ordinal
Remediation level	Describes whether or not a remediation method is available for a given vulnerability, such as *undefined, unavailable, workaround, temporary fix, or official fix*	Nominal
Report confidence	Measures the level of confidence for a given vulnerability as well as known technical details, such as *unknown, reasonable, or confirmed*	Nominal
Environmental metrics		
Security requirement	Describes environment-dependent security requirements in terms of confidentiality, integrity, and availability, such as not *defined, low, medium, or high*.	Nominal
Modified base	Base metrics customized to a specific environment	*

(1) ***Depth metric*** refers to a ratio of the diameter of a domain-level attack graph over the diameter in the most secure case, implying that the larger the diameter, the more secure the network [Noel and Jajodia 2014];
(2) ***Existence, number, and lengths of attack paths metrics*** use the attributes of attack paths from an initial state to the goal state [Ritchey and Ammann 2000; Sheyner et al. 2002; Jha et al. 2002; Cheng et al. 2014; Idika and Bhargava 2012]. These metrics can be used to compare two attacks. For example, an attack with a set X of attack paths is more powerful than an attack with a set Y of attack paths, where $Y \subset X$.

—**Effort metrics** capture the degree of effort by a defender to mitigate vulnerability exploitation by attackers or by an attacker to exploit a given vulnerability. The common metrics under this metric category are as follows:
(1) ***Necessary defense metric*** estimates a minimal set of defense countermeasures necessary for thwarting a certain attack [Sheyner et al. 2002];
(2) ***Effort-to-security-failure metric*** measures an attacker's effort to reach its goal state [Dacier et al. 1996; Ortalo et al. 1999];
(3) ***Weakest adversary metric*** estimates minimum adversary capabilities required to achieve an attack goal [Pamula et al. 2006];
(4) ***k-zero-day-safety metric*** measures a number of zero-day vulnerabilities for an attacker to compromise a target [Wang et al. 2010].

Discussion. *Deterministic severity metrics* are typically defined according to attack graphs based on known vulnerabilities. However, devastating attacks in the real world often exploit zero-day vulnerabilities whose measurement still remains unsolved. To measure system vulnerabilities, the following designs should be considered: (i) identification of vulnerabilities that have never been exploited, (ii) the degree of scanner capability in depth and completeness, and (iii) different types of zero-day attacks. The consideration of these components are closely related to measuring situation awareness as a vulnerability metric, which provides temporal vulnerability states of a system, namely $V(t)$ or $v_i(t)$. Assessing accurate vulnerability state $V(t)$ can provide accurate prediction of future attacks.

Probabilistic Severity Metrics

The deterministic metrics mentioned above assume that vulnerabilities can certainly be exploited, implying that they measure what is possible rather than what is more likely than others [Ou and Singhal 2011; Singhal and Ou 2011]. To be more realistic, one can associate an exploit node (i.e., a node that can be exploited) in an attack graph with a success probability. There are two approaches for defining probabilistic security metrics, differing in treating CVSS scores as atomic parameters or not.

—**Metrics treating CVSS scores as atomic parameters:** The *likelihood of exploitation* metric measures the probability that an exploit will be executed by an attacker with certain capabilities [Wang et al. 2008; Ou and Singhal 2011]. In an attack graph, there are two types of vertices, exploit node e and condition node c, and two types of arcs, $e \rightarrow c$ (meaning execution of exploit e leads to a satisfaction condition c) and $c \rightarrow e$ (meaning satisfaction condition c is *necessary* for executing exploit node e). Each exploit node e is associated with a probability $p(e)$ for the event that e is executed when all conditions c with $c \rightarrow e$ are satisfied, where $p(e)$ is determined based on expert knowledge or CVSS scores. Each condition node is associated with a probability $p(c)$ for the event that c is satisfied when all exploits e with $e \rightarrow c$ are executed. Given an attack graph (or the corresponding Bayesian Network) and the associated $p(e)$'s and $p(c)$'s, one can calculate the *likelihood of exploitation*, namely the overall probability that an exploit node e is executed by an attacker with certain capabilities.

—**Metrics not treating CVSS scores as atomic parameters:** Given an attack graph and a subset of exploits, the *effective base* metric aims to adjust the CVSS base metric by taking the highest value of the ancestors of an exploit to reflect the worst-case scenario, while the *effective base score* metric is calculated based on the effective base metric [Cheng et al. 2012].

Discussion. The preceding metrics are combinatorial in nature and assume a static system environment. While the temporal metrics surveyed in Section 4.3.1 consider dynamic aspects of security, it is not clear how to reconcile these combinatorial metrics with the continuous evolution of systems rather than investigating these systems at individual snapshots over time. Along this direction, a starting point is to convert an attack graph into a Bayesian Network where *nodes* are binary variables corresponding to the exploit nodes in the attack graph and *arcs* encode conditional relationships between the variables [Frigault and Wang 2008; Frigault et al. 2008].

5. DEFENSE METRICS

Defense metrics aim to measure the strength of defense mechanisms placed in a system. We discuss how to measure the strength of *preventive*, *reactive*, and *proactive* defense mechanisms in addition to the strength of an overall system defense.

5.1. Metrics for Measuring the Strength of Preventive Defenses

Preventive defenses aim to block attacks. We focus on measuring the strength of the following preventive defenses: *blacklisting*, *Data Execution Prevention* (DEP, also known as W⊕X), and *Control-Flow Integrity* (CFI).

5.1.1. Metrics for Blacklisting. Blacklisting is a useful, lightweight defense mechanism. Suppose a malicious entity (e.g., attacking computer, IP address, malicious URL, botnet command-and-control server, and drop-zone server) is observed at time t. Then, traffic flowing to or from the malicious entity can be blocked starting at some time $t' \geq t$. Under this situation, two metrics can be derived:

—***Reaction time metric*** captures the delay between the observation of the malicious entity at time t and the blacklisting of the malicious entity at time t' (i.e., $t' - t$) [Kührer et al. 2014], measured by an *absolute scale*.
—***Coverage metric*** estimates the portion of blacklisted malicious players [Kührer et al. 2014]. This metric is measured by an *ordinal scale*.

Discussion. These metrics can be used to compare the strength of different blacklists and guide the design of blacklisting solutions. Based on our envisaged overall security metric in Equation (1), delay and coverage can be used to derive the strength of dynamic attacks based on the probabilities that a certain fraction of entities are compromised at time t, contributing to obtaining the dynamic security state. A measure such as the probability that a malicious entity (e.g., computer, IP address, URL) is or is not blacklisted can be useful in capturing the dynamic security state.

5.1.2. Metrics for DEP. Code injection was a popular method for injecting some malicious code into a running program and directing the processor to execute it. The attack requires the presence of a memory region executable and writable because operating systems do not distinguish code from data (e.g., native stacks on Linux and Windows were executable). The attack can be defeated by deploying DEP, which ensures that a memory page can be writable or executable at any time but not both. To the best of our knowledge, no metrics have been defined to measure the effectiveness of DEP. The effectiveness of DEP can be measured based on the probability of being compromised by a certain attack $A(t)$ over all possible classes of attacks. This requires the precise definition and measurement of this metric.

5.1.3. Metrics for CFI. Memory corruption is a key factor in code injection and code reuse attacks. CFI aims to mitigate such attacks by extracting a program's Control-Flow Graph (CFG), representing the possible execution paths of the program. CFI further instruments the binary code to abide by the CFG at runtime. This is implemented by runtime checking of the tags assigned to the indirect branches in the CFG, such as indirect calls, indirect jumps, and returns, while noting that direct branches do not need validity checking because the pertinent targets are hard coded and cannot be modified by the attacker. CFI mechanisms vary greatly in terms of the types of indirect branches subject to validation and the number of targets with respect to a branch. These factors lead to different tradeoffs between security and performance. We discuss the well-known three metrics as a measure of CFI's quality as follows:

—***Average indirect target reduction*** measures the overall reduction in terms of the number of targets exploitable by the attacker where smaller targets are more secure [Burow et al. 2016]. A defense leading to few but large targets offers less security than a defense leading to more, but smaller, targets. This metric can be measured by an *absolute scale*.

—***Average target size*** is defined as the ratio between the size of the largest target and the number of targets. The smaller the ratio, the better the security [Burow et al. 2016]. An inherent tradeoff exists between efficiency of enforcing CFI and the accuracy of CFG (e.g., granularity of CFI vs. accuracy of CFG affected by the soundness and completeness of the pointer analysis) [Göktas et al. 2014; Davi et al. 2014; Carlini and Wagner 2014].
—***Evasion resistance*** is measured against control flow bending attacks, reflecting the effort (or premises) that an attacker must make (or satisfy) for evading the CFI scheme [Carlini et al. 2015; Niu and Tan 2015].

Discussion. Applying the metrics of CFI strength in Equation (1), the metric for enforcing CFI can be applied as the probability that code-reuse attacks are blocked, representing the strength of CFI against the attacks. To measure evasion resistance against attacks, we need to have a formalism to *precisely* classify attacks. It is a challenging task to derive a right formalism under a high volume of attack types.

5.2. Metrics for Measuring the Strength of Reactive Defenses

Detection mechanisms are well-known strategies for reactive defenses, including intrusion detection systems (IDSs) and anti-malware programs. We discuss metrics for monitoring and measuring their individual, relative, and collective strengths.

5.2.1. Metrics for Monitoring. Attackers can be detected by monitoring mechanisms. Current monitoring practices only consider configurations of intrusion detection monitors, but not monitoring cost or impact of the existing compromised monitors. The common metrics to measure the quality of monitoring mechanisms are [Thakore 2015]:

—***Coverage metric*** measures the fraction of events detectable by a specific sensor deployment, reflecting a defender's need in monitoring events.
—***Redundancy metric*** estimates the amount of evidence provided by a specific sensor deployment to detect an event. The amount of redundancy can be counted by the amount of sensors providing same information towards a same event.
—***Confidence metric*** measures how well-deployed sensors detect an event in the presence of compromised sensors. This task needs to quantify truthfulness of the reports received from individual sensors.
—***Cost metric*** measures the amount of resources consumed by deploying sensors including the cost for operating and maintaining sensors.

Discussion. It is important to define metrics for systematically measuring the strength and robustness of monitor deployments against possibly sophisticated attacks. Since a monitor deployment would have an inherent capability in terms of the set of attack vectors it can detect, it is important to characterize the attack vectors that can and cannot be detected by a specific monitor deployment. These metrics could be immediately used as model parameters $D(t)$ in Equation (1).

5.2.2. Metrics for Detection Mechanisms. Detection mechanisms can be measured via their individual, relative, and collective strengths.

Metrics for the Individual Strength of Defense Mechanisms

—***Detection time:*** For instrument-based attack detection, this metric is used to measure the delay between the time t_0 at which a compromised computer sends its first scan packet and the time t that a scan packet is observed by the instrument [Rajab et al. 2005]. This metric depends on several factors, including the size of the monitored IP address space and the locations of the instrument.

—***Intrusion detection metrics:*** For IDS, including anomaly-based, host-based, and network-based IDS, their strength can be measured by:

(1) ***True-positive rate*** ($\Pr(\mathsf{A}|\mathsf{I})$) is the probability that an intrusion, I, is detected as an attack, A.

(2) ***False-negative rate*** ($\Pr(\neg\mathsf{A}|\mathsf{I})$) is the probability that an intrusion, I, is not detected as an attack, $\neg\mathsf{A}$.

(3) ***True-negative rate*** ($\Pr(\neg\mathsf{A}|\neg\mathsf{I})$) is the probability that a non-intrusion, $\neg\mathsf{I}$, is not detected as an attack, $\neg\mathsf{A}$.

(4) ***False-positive rate*** ($\Pr(\mathsf{A}|\neg\mathsf{I})$), or *false alarm rate*, is the probability that a non-intrusion, $\neg\mathsf{I}$, is detected as an attack, A. Note that $\Pr(\mathsf{A}|\mathsf{I})+\Pr(\neg\mathsf{A}|\mathsf{I}) = \Pr(\neg\mathsf{A}|\neg\mathsf{I})+\Pr(\mathsf{A}|\neg\mathsf{I}) = 1$.

(5) ***Intrusion detection capability metric*** is the normalized metric of $\mathbf{I}(\mathcal{I}, \mathcal{O})$ with respect to $H(\mathcal{I})$ based on the base rate where $\mathcal{I}$ is the input to the IDS as a stream of 0/1 random variables (0 for benign/normal and 1 for malicious/abnormal), $\mathcal{O}$ is the output of the IDS as a stream of 0/1 random variables (0 for no alert or normal; 1 for alert / abnormal), $H(\mathcal{I})$ and $H(\mathcal{O})$, respectively, denote the entropy of $\mathcal{I}$ and $\mathcal{O}$, and $\mathbf{I}(\mathcal{I}, \mathcal{O}) = H(\mathcal{I}) - H(\mathcal{I}|\mathcal{O})$ is the mutual information between $\mathcal{I}$ and $\mathcal{O}$ [Gu et al. 2006].

(6) ***Receiver operating characteristic*** (ROC) curve reflects the dependence of the true-positive rate $\Pr(\mathsf{A}|\mathsf{I})$ on the false-positive rate $\Pr(\mathsf{A}|\neg\mathsf{I})$, reflecting a tradeoff between the true-positive and the false-positive rates.

(7) ***Intrusion detection operating characteristic*** (IDOC) curve describes the dependence of the true positive rate $\Pr(\mathsf{A}|\mathsf{I})$ on the Bayesian detection rate $\Pr(\mathsf{I}|\mathsf{A})$, while accommodating the base rate $\Pr(\mathsf{I})$ [Cardenas et al. 2006].

(8) ***Cost metric*** includes the *damage* cost incurred by undetected attacks, the *response* cost spent on the reaction to detected attacks including both true and false alarms, and the *operational* cost for running an IDS [Stolfo et al. 2000; Lee et al. 2002; Strasburg et al. 2009].

Discussion. The metrics mentioned above are mainly geared towards measuring the detection strength of individual detection systems and comparing the strength of two detection systems. The base rate $\Pr(\mathsf{I})$ of intrusions, if not adequately treated, may cause misleading results; this is called the *base-rate fallacy* [Axelsson 2009]. For example, the ROC curve does not consider the base rate information and therefore can be misleading when the base rate is very small; whereas the IDOC curve accommodates the base rate information, meaning that it does not suffer from this problem and can be used to compare different IDSs that operate in environments with different base rates [Cardenas et al. 2006]. The use of the ROC curve and cost metrics together to evaluate IDSs is introduced in Gaffney Jr and Ulvila [2001]. A general cost-sensitive evaluation of IDSs should consider both the security objective and the incurred cost, while noting that the cost incurred by an IDS should not be exceed the loss caused by intrusions [Lee et al. 2002; Cardenas et al. 2006; Strasburg et al. 2009; Stakhanova et al. 2012]. Gu et al. [2008] present a method for aggregating the decisions of multiple IDSs by taking into consideration the false-alarm cost and the false-negative damage. According to a review of the publications appeared between 2000 and 2008 [Tavallaee et al. 2010], most experimental studies in anomaly-based intrusion detection lack a scientific rigor. Milenkoski et al. [2015] discuss more details for metrics of detection mechanisms. When modeling intrusion detection systems in a holistic perspective, $D(t)$ in Equation (1), the *detection probability* metric refers to the conditional probability that a compromised computer at time t is indeed detected as compromised at time t, $\Pr(o_i(t) = 1|s_i(t) = 1)$. This detection probability is often dependent on other factors such as attack severity or strength of defense mechanisms.

Metrics for the Relative Strength of Defense Mechanisms
This metric reflects the strength of a defense tool when employed in addition to other defense tools [Boggs and Stolfo 2011; Boggs et al. 2014]. A defense tool does not offer any extra strength if it cannot detect any attack undetected by other defense tools in place. Let $\mathcal{A}$ denote a set of attacks, $\mathcal{D} = \{d_1, \ldots, d_n\}$ denote the set of n defense tools, and X_d denote the set of attacks detected by a defense tool $d \in \mathcal{D}$. The *relative strength* of a defense tool $d' \in \mathcal{D}'$ with respect to a set of defense tools $\mathcal{D} \subset \mathcal{D}'$ is defined as $\frac{|X_{d'} - \cup_{d \in D} X_d|}{|\mathcal{A}|}$. The metric is measured by a *ratio scale*.

Discussion. These metrics can be applied as parameters $D(t)$ in Equation (1) for characterizing the defense tools. They can be useful in predicting future or unknown attacks, which can enhance security decision making capability.

Metrics for the Collective Strength of Defense Mechanisms
This metric measures the collective strength of IDSs and anti-malware programs [Boggs and Stolfo 2011; Morales et al. 2012; Boggs et al. 2014; Mohaisen and Alrawi 2014; Yardon 2014]. Denote by $\mathcal{A}$ a set of attacks, $\mathcal{D} = \{d_1, \ldots, d_n\}$ a set of defense tools and X_d the set of attacks detected by defense tool $d \in \mathcal{D}$. The collective detection strength of defense tools is defined as $\frac{|\cup_{d \in \mathcal{D}} X_d|}{|\mathcal{A}|}$ [Boggs and Stolfo 2011; Boggs et al. 2014]. For malware detection, the collective use of multiple anti-malware programs still cannot detect all malware infections [Morales et al. 2012; Mohaisen and Alrawi 2014; Yardon 2014]. For example, Yardon [2014] showed that anti-malware tools are only able to detect 45% of attacks. This metric is measured by a *ratio scale*.

Discussion. These metrics can be used to compare the collective strength of two sets of detection tools. They can be used as parameters $D(t)$ in Equation (1) for characterizing the strength of defense in depth. They might need to be estimated with respect to known and unknown attacks.

5.3. Metrics for Measuring the Strength of Proactive Defenses

We discuss metrics for measuring two major proactive defense mechanisms, *Address Space Layout Randomization* (ASLR) and *Moving Target Defense* (MTD). These mechanisms are proactive because a system can be constantly re-configured to hinder the attack process. On the other hand, preventive and reactive defenses often only incur changes to the defense tools (e.g., updating malware detection signatures or rules).

5.3.1. Address Space Layout Randomization (ASLR). ASLR was first introduced by the Linux PaX project to defend against code reuse attacks by randomizing the base addresses (i.e., shuffling the code layout in the memory) such that the attacker cannot find useful gadgets. Coarse-grained ASLR has the vulnerability that the leak or exposure of a single address gives the attacker adequate information to extract all code addresses. Fine-grained ASLR does not suffer from this problem (e.g., page-level randomization [Backes and Nürnberger 2014; Larsen et al. 2014]) but is still susceptible to attacks that craft attack payloads from Just-In-Time (JIT) code [Snow et al. 2013].

This attack can be defeated by *destructive code read*, namely that the code in executable memory pages is garbled once it is read [Tang et al. 2015]. ASLR can also be enhanced by preventing the leak of code pointers, while rendering leakage of other information (e.g., data pointers) useless for deriving code pointers [Lu et al. 2015]. Two metrics for measuring the strength of ASLR are as follows:

—***Entropy metric*** measures the entropy of a memory section, while noting that a greater entropy would mean a greater effort in order for an attacker to compromise the system. For example, a brute-force attack can feasibly defeat a low-entropy ASLR on 32-bit platforms [Shacham et al. 2004].

—***Effective entropy metric*** measures the entropy in a memory section that the attacker cannot circumvent by exploiting the interactions between memory sections [Herlands et al. 2014].

5.3.2. Moving Target Defense (MTD). There have been some initial efforts to measure the strength of MTD. Han et al. [2014] proposed a metric to measure the effectiveness of MTD by measuring the degree that an enterprise system can tolerate some undesirable security configurations in order to direct the global security state $S(t)$ towards a desired stable system state. Similar studies have been conducted via experimentation [Zaffarano et al. 2015], emulation [Eskridge et al. 2015], and simulation [Prakash and Wellman 2015]. However, they do not explicitly define the metrics while measuring an abstract level of metrics through the experimental platform.

Discussion. The effectiveness of individual or collective proactive defense mechanisms can be measured based on the probabilities in a *ratio scale* or *distribution scale*. The resulting metrics, especially those measuring their strength on individual computers, could be incorporated into Equation (1) as model parameters $D(t)$ to reason security state $S(t)$ as part of *situation*(t). Potential useful metrics can capture *security gain* or an extra effort required by attackers after the use of MTD.

5.4. Metrics for Measuring the Strength of Overall Defenses

We discuss the two metrics aiming to measure the strength of overall defenses of a given system as follows:

—**Penetration resistance** (PR) can be measured by running a penetration test to estimate the level of effort (e.g., person-day or cost) required for a red team to penetrate into a system [Levin 2003; Carin et al. 2008]. This metric can be used to compare the defense strength of two systems against a same red team.
—**Network diversity** (ND) measures the least or average effort an attacker must make to compromise a target entity based on the causal relationships between resource types to be considered as the inclusion in an attack graph [Zhang et al. 2016]. For example, in the attack graph, this metric can be defined as the ratio of the minimum number of pathwise distinct resources among all attack paths, representing the least effort an attacker must make to the length of the shortest attack path.

Discussion. PR is estimated based on a particular red team scenario. An important research question is how to bridge the gaps between identification of security holes and quantification of security [Sanders 2014]. PR can consider novel attacks, which may exploit some known or zero-day vulnerabilities based on "what-if" analysis scenarios. Key aspects to investigate or improve ND include software diversity and the associated maintenance overhead considering dynamics of a given setting.

6. ATTACK METRICS

Attack metrics measure the strength of attacks performed against a system. We discuss the following attacks, which have high impact when they are defeated: *zero-day attacks*, *targeted attacks*, *botnets*, *malware spreading*, and *evasion techniques*.

6.1. Metrics for Measuring Zero-Day Attacks

Two metrics to measure how many zero-day attacks were launched during certain past period are as follows [Bilge and Dumitras 2012]:

—***Lifetime of zero-day attacks*** measures the period of time between when an attack was launched and when the corresponding vulnerability is disclosed to the public.

—***Victims by zero-day attacks*** measures the number of computers compromised by zero-day attacks.

Discussion. Zero-day metrics typically measure the consequence in retrospect. However, if we can capture the degree of the *susceptibility of a device to zero-day attacks*, then we can prioritize resource allocation to the ones requiring higher attention for mitigating the damage. This can be considered as the part of the attack abstraction $A(t)$ in Equation (1) based on the *number of zero-day attacks* at time t.

6.2. Metrics for Measuring Targeted Attacks

The success of targeted attacks or Advanced Persistent Threats often depends on the delivery of malware and the tactics to lure a target to open malicious email attachments. Let α denote a social engineering tactic, ranging from the least sophisticated to the most sophisticated (e.g., $\alpha \in \{0, \ldots, 10\}$). Let β denote a technical sophistication of the malware in the attacks, ranging from the least sophisticated to the most sophisticated (e.g., $\beta \in [0, 1]$). The ***targeted threat index*** metric, indicating the level of targeted malware attacks, can be defined as $\alpha \times \beta$ [Hardy et al. 2014].

Discussion. Little work has been done in measuring the intensity or severity of targeted attacks. Measuring the level of susceptibility to a given attack (e.g., social-engineering attacks such as phishing emails) for a particular type of user (e.g., personality traits, cognitive biases/limitations) can be a first step to enhance defense mechanisms and to allocate right defense resources at the right places. Progress on this matter can be incorporated into Equation (1) as model parameters $A(t)$.

6.3. Metrics for Measuring Botnets

The threat of *botnets* can be characterized by the following metrics:

—***Botnet size*** refers to the number of bots, x, that can be instructed to launch attacks (e.g., distributed denial-of-service attacks) at time t, denoted by $y(t)$. Due to time zone difference, $y(t)$ is often much smaller than the actual x as some of x is turned off during night time at time zones [Dagon et al. 2006].
—***Network bandwidth*** indicates the network bandwidth that a botnet can use to launch denial-of-service attacks [Dagon et al. 2007].
—***Botnet efficiency*** can be defined as the network diameter of the botnet network topology [Dagon et al. 2007]. It measures a botnet's capability in communicating command-and-control messages and updating bot programs.
—***Botnet robustness*** measures the robustness of botnets under random or intelligent disruptions [Dagon et al. 2007]. The idea was derived from the concept of *complex network robustness* [Albert and Barabasi 2002], characterized by the *percolation threshold* under which a network is disrupted into small components.

Discussion. Defining a metric of *botnet attack power* is a key to prioritize the countermeasures against botnets. A metric of measuring *dynamic robustness* can reflect the degree of countermeasures that can be exploited by attackers when defenders fight against the botnets. These metrics can be incorporated into Equation (1) as model parameters $A(t)$. Metrics of *botnet attack power* and *botnet resilience with counter-countermeasures* (i.e., the attacker attempts to dynamically mitigate the defender's process for disrupting a botnet) can be useful in guiding defense operations.

6.4. Metrics for Measuring Malware Spreading

Malware spreading is a common attack where a malware can spread out at a certain rate. The ***infection rate*** metric, denoted by γ, measures the average number of vulnerable computers that are infected by a compromised computer (per time unit) at the

early stage of spreading. Intuitively, γ depends on the scanning strategy. Let z denote the number of scans and infections made by an infected computer (per time unit) and w denote the number of vulnerable computers. With random scanning over the IPv4 address space, the infection rate is $\gamma = zw/2^{32}$ [Chen and Ji 2007].

6.5. Metrics for Measuring Attack Evasion Techniques

Sophisticated attacks can evade defense mechanisms placed in a system using several strategies. We discuss two types of metrics for measuring the strength of those types of attacks: *adversarial machine learning* and *obfuscation*.

6.5.1. Metrics for Measuring Adversarial Machine Learning Attacks. Attackers can manipulate some features that are used in the detection models (e.g., classifiers). This problem is generally known as *adversarial machine learning* [Dalvi et al. 2004; Lowd and Meek 2005; Huang et al. 2011; Šrndic and Laskov 2014]. Depending on the knowledge an attacker has about a detection model, various evasion scenarios can be possible. The spectrum of evasion scenarios includes that an attacker knows (i) a set of features used by a defender; (ii) both a set of features and training samples used by the defender; and (iii) a set of features, the training samples, and the attack detection model (e.g., classifiers) used by the defender [Šrndic and Laskov 2014; Xu et al. 2014b]. The strength of attacks can be measured by typical detection metrics such as *increased false-positive* and *increased false-negative* rates as a consequence of applying a certain evasion method. A framework for evaluating classifiers at their learning phase via "what-if" analysis of attacks is presented in Biggio et al. [2014].

6.5.2. Metrics for Measuring Obfuscation Attacks. Obfuscation based on tools such as runtime packers have been widely used by malware writers to defeat static analysis. Little is understood about how to quantify the obfuscation capability of malware, except for the following two metrics:

—***Obfuscation prevalence metric*** measures the occurrence of obfuscation in malware samples [Roundy and Miller 2013].
—***Structural complexity metric*** measures the runtime complexity of packers in terms of their layers or granularity [Ugarte-Pedrero et al. 2015].

Discussion. Measuring the *evasion capability* of attacks not only allows comparing the evasion power of two attacks but also allows computing the damage caused by evasion attacks. There is potential in measuring the *obfuscation sophistication* of malware in terms of the amount of effort required for unpacking packed malware. This can help distinguish the malware needing a manual unpacking process from the malware that can be unpacked automatically. These metrics can be incorporated as model parameters $A(t)$ in Equation(1).

7. SITUATION METRICS

Situation metrics, denoted as $situation(t)$ in this work, reflect the comprehensive manifestation of attack-defense interactions with respect to an enterprise or computer system. These metrics can be divided into three sub-categories: *security states* $S(t)$, *historical security incidents*, and *historical security investments*.

7.1. Metrics for Measuring Security State

As illustrated in Figures 1 and 2, the security state of an enterprise system $S(t) = (s_1(t), \ldots, s_n(t))$ and the security state $s_i(t)$ of computer $c_i(t)$ both dynamically evolve as an outcome of attack-defense interactions. We discuss the following two types of security state metrics in this section: *data driven* and *model driven*.

7.1.1. Data-Driven State Metrics. This type of metric measures system state based on data. The examples of data-driven state metrics are as follows:

—***Network maliciousness metric*** estimates the fraction of blacklisted IP addresses in a network. According to Zhang et al. [2014], there were 350 autonomous systems that had at least 50% of their IP addresses blacklisted. Moreover, there was a correlation between mismanaged networks and malicious networks, where "mismanaged networks" are those networks that do not follow accepted policies/guidelines.
—***Rogue network metric*** captures the population of networks used to launch drive-by download or phishing attacks [Stone-Gross et al. 2009].
—***ISP badness metric*** quantifies the effect of spam from one ISP or Autonomous System (AS) on the rest of the Internet [Johnson et al. 2012].
—***Control-plane reputation metric*** calibrates the maliciousness of attacker-owned (i.e., rather than legitimate but mismanaged/abused) ASs based on their control plane information (e.g., routing behavior), which can achieve an *early-detection time* of 50–60 days (before these malicious ASs are noticed by other defense means) [Konte et al. 2015].
—***Cybersecurity posture metric*** measures the dynamic threat imposed by the attacking computers [Zhan et al. 2014]. It may include the attacks observed at honeypots, network telescopes, and/or production enterprise systems. These metrics reflect the aggressiveness of cyber attacks.

7.1.2. Model-Driven Metrics. This type of metric measures system states in terms of the outcome of attack-defense interaction models.

—***Fraction of compromised computers*** is denoted by $|\{i : i \in \{1, \ldots, n\} \wedge s_i(t) = 1\}|/n$. Under certain circumstances, there is a fundamental connection between a global security state and a very small number of nodes that can be monitored carefully [Xu et al. 2012a].
—***Probability a computer is compromised at time t*** is represented as $\Pr[s_i(t) = 1]$ as illustrated in Figure 1. This metric is proposed to quantify the degree of security in enterprise systems by using modeling techniques based on a holistic perspective [LeMay et al. 2011; Xu and Xu 2012; Da et al. 2014; Xu 2014a; Zheng et al. 2015; Xu et al. 2015a, 2015b; Xu 2014b; Han et al. 2014; Xu et al. 2014a; Lu et al. 2013; Xu et al. 2012a, 2012b; Li et al. 2011]. These models aim to quantify the laws governing the evolution of the security state $S(t)$.

Discussion. As indicated by Equation (1), it is vital for the defender to have knowledge of dynamic states of system security that provides right directions to defend against attacks. For example, if a compromising probability towards a device is known, it is helpful for a defender to use an appropriate threshold cryptographic mechanism [Desmedt and Frankel 1989] to mitigate the compromising action. Moreover, it is important to know $S(t)$ and $s_i(t)$ for any t rather than for $t \to \infty$ [Xu 2014a].

7.2. Metrics for Measuring Security Incidents

Measuring security incidents is another aspect of $situation(t)$. To obtain the severity and impact of incurred security incidents, we discuss how to measure the frequency and damage of the security incidents.

7.2.1. Measuring Frequency of Security Incidents. Security incidents can be measured by the following:

—***Encounter rate*** measures the fraction of computers that encountered some malware or attack during a period of time [Yen et al. 2014; Mezzour et al. 2015].

—***Incident rate*** measures the fraction of computers successfully infected or attacked at least once during a period of time [CIS 2010]. An incident rate tends to be substantially smaller than the corresponding encounter rate because the encountered malware may not successfully infect a system [Microsoft 2014].
—***Blocking rate*** is the rate an encountered attack is successfully defended by a deployed defense [Yen et al. 2014].
—***Breach frequency metric*** measures how often breaches occur [Edwards et al. 2015].
—***Breach size metric*** gives the number of records breached in individual breaches [Edwards et al. 2015].
—***Time-between-incidents metric*** measures the period of time between two incidents [CIS 2010; Holm 2014].
—***Time-to-first-compromise metric*** estimates the duration of time between when a computer starts to run and the first malware alarm is triggered on the computer where the alarm indicates detection rather than infection [Jonsson and Olovsson 1997; Madan et al. 2002; Holm 2014].

Discussion. These metrics may be used as alternatives to the global security state $S(t)$, especially when $S(t)$ is difficult to obtain for arbitrary t. An ideal metric may be an *incident occurrence frequency* as an approximation of the number of compromised computers, namely $|\{i : i \in \{1, \ldots, n\} \wedge s_i(t) = 1\}|$, represented as a mathematical function of system features. It is also important to predict a *frequency of incident occurrence* as an approximate number of compromised computers at a future time t. A recent study shows the possibility of predicting data breaches from symptoms such as network mismanagement or blacklisted IP addresses [Liu et al. 2015].

7.2.2. Measuring Damage of Security Incidents. The damage to a system after a security incident occurs can be estimated based on the degree of impact caused by the security incident. The impact can be estimated by the following metrics:

—***Delay in incident detection*** measures the time between the occurrence and detection [CIS 2010], implying that a longer delay is a higher damage.
—***Cost of incidents*** may include both the direct cost (e.g., the amount of lost money) and the indirect cost (e.g., negative publicity and/or the recovery cost) [CIS 2010].

Discussion. Accurate estimation of potential damage (or impact) due to a security incident is critical to estimating the degree of risk on security failure. The level of risk one can take is significantly affected by the degree of consequence on the security incident (e.g., risk-seeking, risk-neutral, or risk-averse).

7.3. Metrics for Measuring Security Investment

Investment to ensure an enterprise's security can be measured as follows:

—***Security spending*** indicates a percentage of IT budget [Chew et al. 2008; CIS 2010]. This metric is important because enterprises want to know whether their security expenditure is justified by the security performance and is comparable to other organizations' security investments.
—***Security budget allocation*** estimates how the security budget is allocated to various security activities and resources [CIS 2010].
—***Return on security investment*** (ROSI) is a variant of the classic *return on investment* (ROI) metric [Berinato 2002; Böhme and Nowey 2008], measuring the financial net gain of an investment based on the gain from investment minus the cost of investment. Since security is not a real investment (i.e., not generating a revenue), the ROSI metric actually measures the reduction in the loss caused by incompetent security [Wei et al. 2001; Berinato 2002; Böhme and Nowey 2008].

—***Net present value*** measures the difference between the present economic value of future inflows and the present economic value of outflows with respect to an investment [Gordon and Loeb 2006]. When determining the benefit of an investment decision (e.g., adopting a new security/defense mechanism), a positive net present value means a profitable investment.

Discussion. Measuring security investment is important for security practitioners to justify security investment. Although they need to juggle between cost and benefit where security investment does not generate revenues but aims to prevent loss, the absence of effective and efficient systematic security metric makes this decision more challenging. Understanding the *payoff of security investment* requires in-depth investigation on the relationship between the cost and benefit of using security/defense mechanisms to be employed in a system. Although security investment is indirectly accommodated in Equation (1) via vulnerability parameters $V(t)$ and defense parameters $D(t)$, the investment metrics are closely related to the dynamic security state $S(t)$ to justify necessity of the investment.

8. DISCUSSION ON OPEN PROBLEMS IN SECURITY METRICS RESEARCH

8.1. Critical Dimensions of Measurements

Figure 4 gives a pictorial description of the taxonomy of the four types of systems security metrics reviewed in the previous sections. Figure 5 is a pictorial description of the taxonomy of the desirable systems security metrics reviewed in the previous sections. Table 1 in the Appendix summarizes these metrics with references (i.e., second column), and the desirable metrics corresponding to the sub-categories of security metrics (i.e., third column). We observe that there are big gaps between the existing metrics and the desirable metrics. The gaps further highlight an important issue, namely "what can be measured" (i.e., measurability) vs. "what to measure" (i.e., core attributes to measure). This issue has been discussed as a fundamental but challenging problem in measurements and metrics [Pfleeger 2009]. To some extent, measuring the four types of security metrics can answer the question of "what to measure."

As highlighted in Equation (1), we proposed four dimensions of security metrics: $situation(t)$ including the dynamic system security state $S(t)$, vulnerabilities $V(t)$, defenses $D(t)$, and attacks $A(t)$. For each of these metrics, we need to identify the representations of the dynamic vulnerabilities $V(t)$, defenses $D(t)$, and attacks $A(t)$. Each metric function can be described by a few key parameters. This explains why we used the *thickness* of blue bars and red arrows in Figures 1 and 2 to *illustrate* the defense strength and attack strength, respectively. However, it remains to discover what metrics can adequately reflect the *thickness*. As surveyed in this article, many metrics have been proposed to measure defense strength and attack strength. Although we have identified some metrics that well describe the respective factors in the third column of Table 1, whether these metrics are sufficient for fulfilling the vision described in Equation (1) remains unanswered.

8.2. Completeness of Security Metrics

At a high level, $V(t)$ can be represented by one or multiple model parameters measuring an *overall system vulnerability*, $D(t)$ measuring an *overall defense strength*, and $A(t)$ measuring an *overall attack strength*. In devising each metric function, we need to consider (i) how many parameters we need to consider for each metric as a minimum, (ii) what attributes we need to consider the intuitive meaning of each metric, and (iii) how to represent parameters in each metric. Although there can be more, we limit our discussion due to space constraints. In particular, the issues of (i) and (ii) are interwoven because as more key attributes are considered, more parameters need to

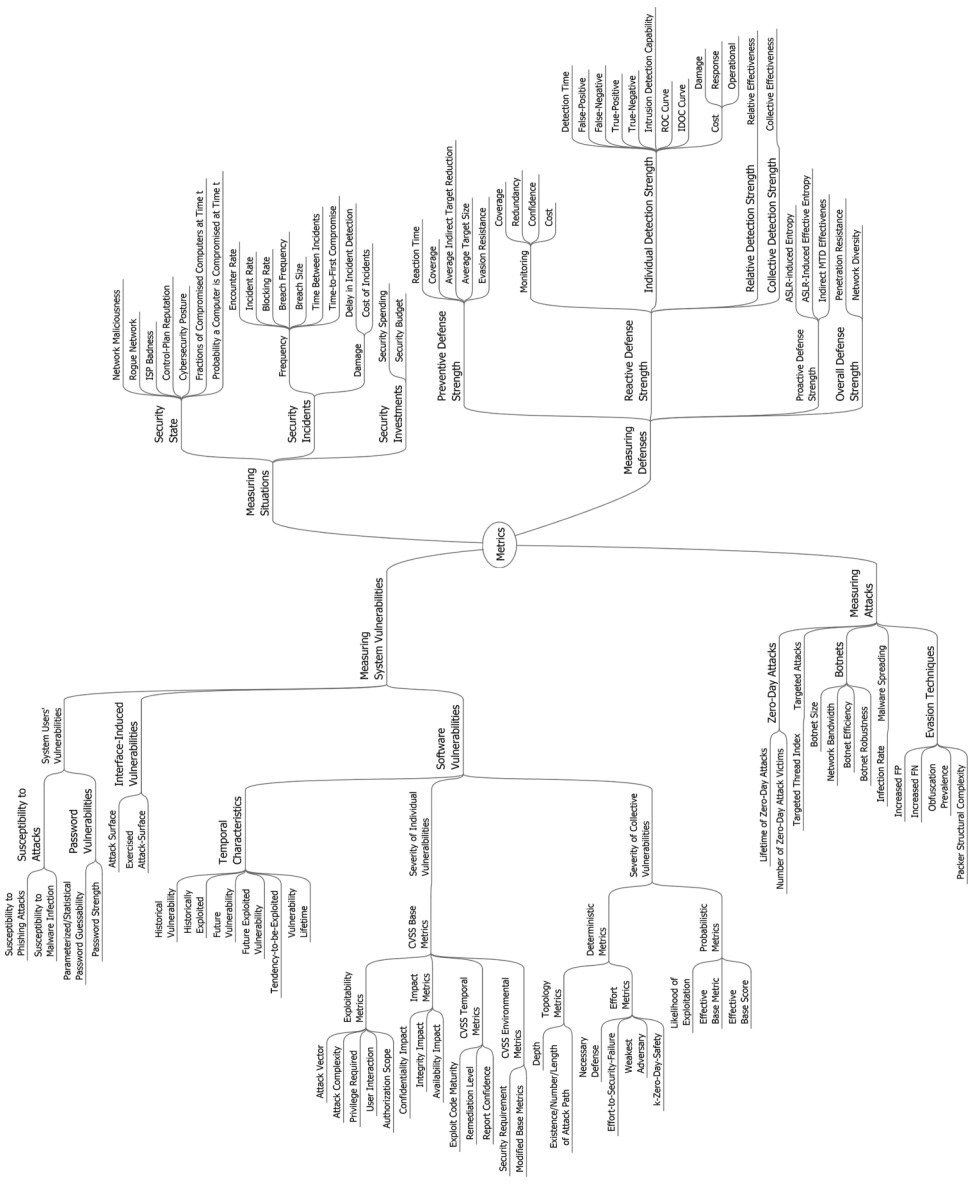

Fig. 4. A pictorial taxonomy of the systems security metrics surveyed in the text.

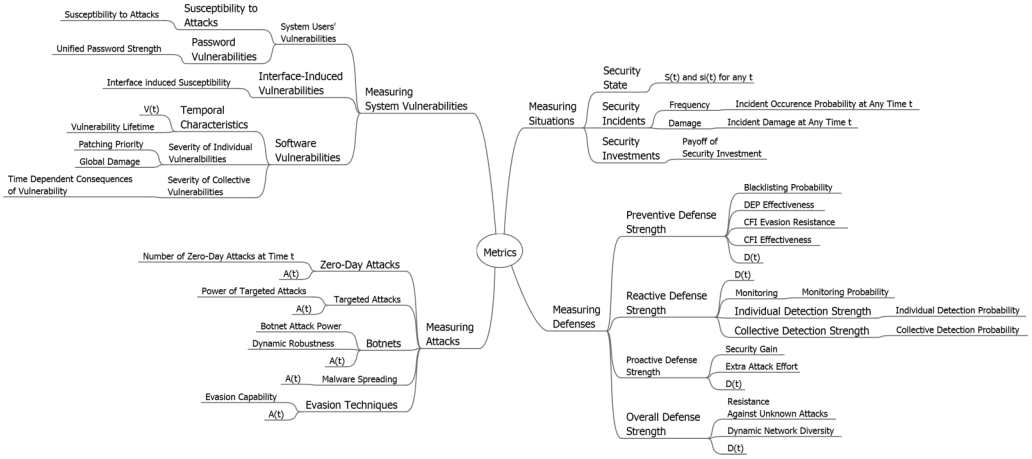

Fig. 5. A pictorial taxonomy of the desirable systems security metrics discussed in the text.

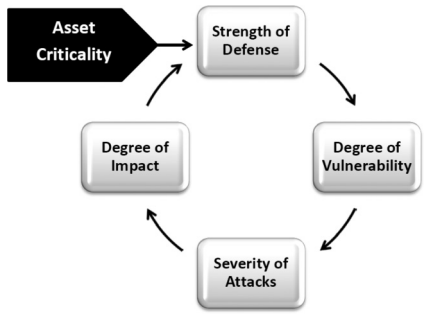

Fig. 6. An illustration of the system security state as determined by the metrics of vulnerabilities, defenses, and attacks.

be considered. As a potential approach to embrace comprehensive measurements of attributes that maximize the completeness of metrics, we can derive a set of sufficient attributes or dimensions that can adequately reflect the intuition for each metric behind. After then, by leveraging aggregation methods, we can categorize a set of sufficient dimensions into key measurement attributes. However, it is unavoidable to deal with a tradeoff issue between dimensions of data and accuracy. That is, how to reflect the completeness of attributes measured in metrics explains the part of why developing good metrics is a hard problem.

8.3. Relationships between Metrics

In this section, we summarize the relationships of the proposed security metric framework consisting of the four sub-metrics as shown in Figure 6. Given an asset with a certain level of criticality, a different level of defense mechanism(s) can be deployed to the asset. Based on the level of defense deployed in each asset, a different level of vulnerability can be exposed, which also affects an attacker's effort to penetrate into an asset or system. The attack's strength can make a different impact to the asset or an entire system with a different level of damage made to it. The outcome of the defense-attack interactions can be fed back to enhance or maintain a current defense state. This suggests that investigating critical dependence between key components of metrics can contribute to building desirable design features a security metric framework

should deal with to maximize adaptive, learning decision making taken by defenders considering dynamics and novel behaviors of attackers.

Figure 3 presents a higher-level ontology of vulnerability metrics, defense metrics, attack metrics, and situation metrics and their relationships. We have defined an ontology to describe the relationships between the systems security metrics. Due to space limitation and the infeasibility of depicting the entire ontology with any reasonable visual effect, we use Figure 7 to show a portion of the vulnerability sub-ontology corresponding to two sub-categories of metrics for measuring vulnerabilities, *temporal characteristics of vulnerabilities* and *severity of individual vulnerabilities*. The ontology captures relationships between the following security metrics:

—*Exploit maturity* "increases" *future exploited vulnerability* and *tendency-to-be-exploited* metrics because attack tools become available.
—*Attack complexity*, *privilege required*, and *user interaction* metrics "affect" *future exploited vulnerability* and *tendency-to-be-exploited* metrics because some attacks demand more attack skills or cooperation of the innocent user.

8.4. Scales for Measurements

Security metrics are difficult to measure in practice due to their inherent existence of uncertainty. The uncertainty can be derived from multiple reasons. First, **unknown attack behaviors** are hard to be accurately predicted by a defender. For example, vulnerabilities are dynamically discovered while an attacker may identify some zero-day vulnerabilities unknown to the defender. Moreover, the defender is uncertain about what exploits the attack possess. Some attack incidents are never detected by the defender. These indicate that uncertainty is inherent to the threat model the defender is confronted with. As a consequence, security metrics should often be treated as random variables rather than numbers. This implies that there should be an effort to characterize distributions of random variables representing security metrics rather than the means of random variables only.

Second, uncertainty is often caused by **estimation error(s)** in that observed evidence does not necessarily reflect an actual system state such as $S(t) \neq O(t)$ as illustrated in Figure 1. The estimation error(s) can be from detection errors by machines (e.g., misdetection) or cognitive limitations/bias by humans where both can affect security decisions to make. In addition, this justifies why measurements based on random variables can be useful to reflect such as distributions of observations rather than giving a single value to represent a quality of system security. Probabilistic metrics are also another promising direction to consider the degree of uncertainty in metrics [Wang et al. 2008; Ou and Singhal 2011; Cheng et al. 2012]. However, more effort is strongly encouraged in this research because the evolution of security state $S(t)$ inherently follows some stochastic processes.

8.5. Granularity vs. Overall Security

It is not a trivial issue to determine whether to aggregate multiple metrics into one or not. For example, if we aggregate multiple metrics to represent an overall system state with a single value, granularity of representing many different metrics will be lost. In particular, when we want to make defense decisions based on the level of system security observed, some decisions associated with resource allocation need more detailed information of metrics (e.g., what attack is performed where, how, and what target). On the other hand, if we do not aggregate any measurements into a certain high-level system attribute (e.g., availability, integrity, reliability), it would be endless to report all the details of attack, impact, and countermeasures, and so forth. This is known as the challenge in "measurability."

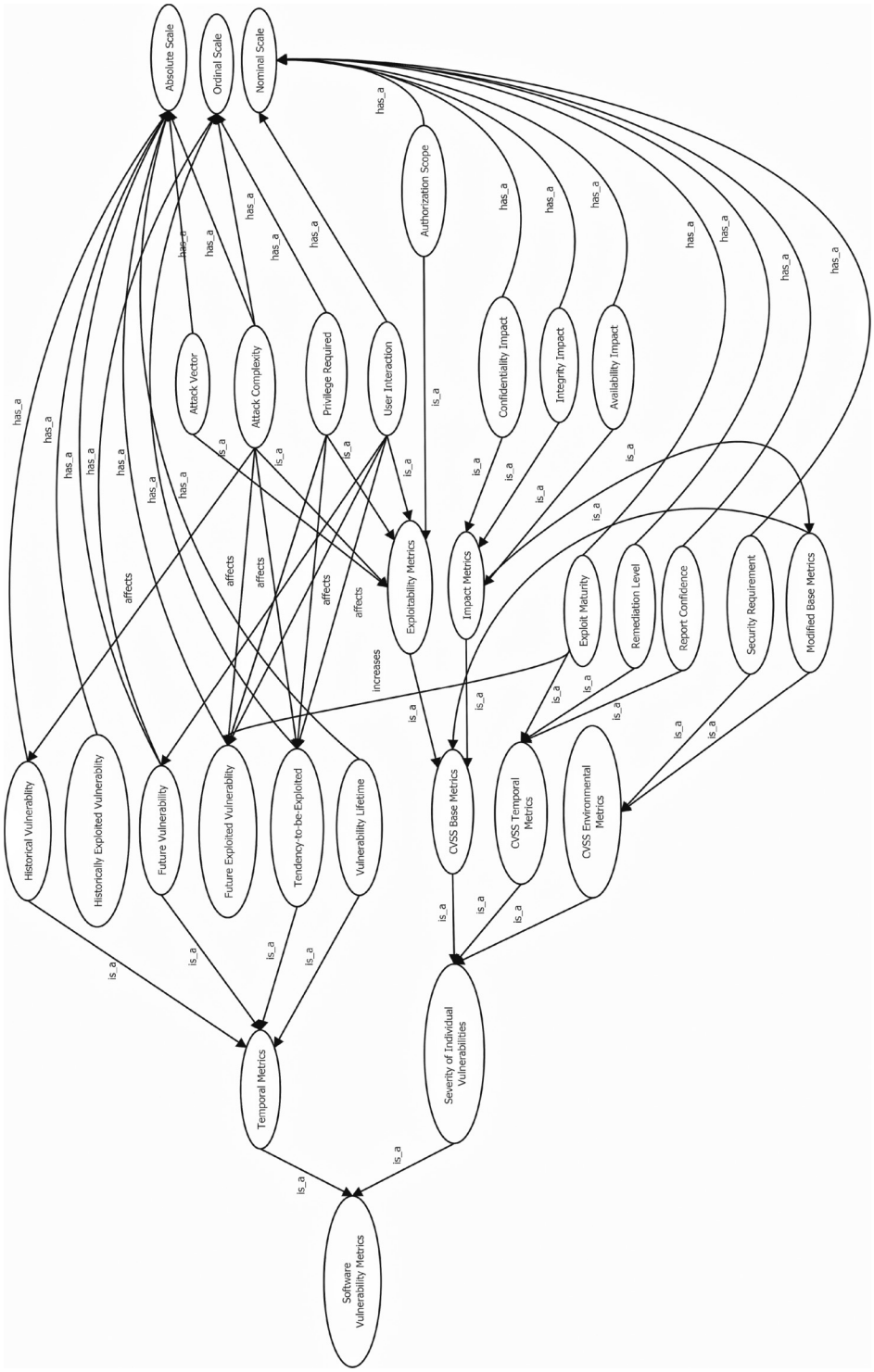

Fig. 7. Sub-ontology metrics describing *temporal characteristics of vulnerabilities* and *severity of individual vulnerabilities*.

Lampson [2006] first raised an issue of why security should be considered at multiple abstraction layers. For example, we should use a specific metric to measure the effectiveness of CFI in stopping control-flow hijack attacks because an overall security metric would be useful for decision making in cyber defense operations but may not give any specific solution to this specific problem. Although some specific attacks can be detected by specific security metrics, an overall security metric based on the aggregation of multiple security metrics also gives great insights to system designers who aim to build defense mechanisms under limited resources. Therefore, the key issue is narrowed down **how to aggregate and to what extent to aggregate** [Pfleeger and Cunningham 2010; Pfleeger 2009].

Regarding how to aggregate multiple security metrics, the main challenge comes from the dependence between security metrics. For example, some dependence may exist between the *security investment*, *security coverage*, and *damage of incidents*. When the mean of random variables, representing metrics, is the only concern, we may indeed aggregate multiple measures into a single value via linear combination techniques. However, a mean of random variables is just one decision-making factor because we often need to consider the variance of random variables as a measure of uncertainty or confidence.

8.6. Key Properties of Security Metrics

Now we discuss several ***perspectives towards good metrics*** as follows:

—**Conceptual perspective** says that a good metric should be intuitively easy to understand with high *usability* to both researchers and defense operators [Reiter and Stubblebine 1999; Lippmann et al. 2012].
—**Measurement perspective** claims that a good metric should be in units of measure, be represented numerically, have a specific context (i.e., a threat model), be measured with high consistency and repeatability, and be easily and cost-effectively (i.e., inexpensively) to collect [Jaquith 2007].
—**Utility perspective** provides a view that a good metric should allow both *horizontal comparison* between enterprise systems and *temporal comparison* (e.g., an enterprise system in the present year vs. the same enterprise system in the last year) [Lippmann et al. 2012].

Security metrics do not possess the properties of metrics in mathematics, where a metric measures the distance between any two points in some metric space. As discussed in McHugh [2006] and Wagner and Eckhoff [2015], security (and privacy) metrics are not necessarily meant to measure the distance between the security of one system and the security of another system. Instead, security metrics are commonly used to measure, among other sense, the security of a system. Even in the narrower context of measuring security of systems (because there are other kinds of security properties that need to be measured as well, such as the *power of exploit*), it is not immediately clear under what conditions the distance between the security of one system and the security of another system makes a good sense or under what conditions we can compare the security of two systems. This research direction needs a good progress.

8.7. Key Suggestions

In this section, we discuss how security metrics research can be further stimulated to gear towards developing desirable metrics to measure levels of system security. We suggest the following action items:

—**Security publications should specify explicit definitions of security metrics they use.** This effort can be made in terms of both *bottom-up* and *top-down*

approaches. For the bottom-up approach, each publication should define specific security metrics and their attributes. For the top-down approach, the security metrics used should achieve security goals in a broad sense such as five security goals including availability, integrity, confidentiality, non-repudiation, and authentication. One may consider a security metric in terms of its *temporal* characteristics, its *spatial* characteristics, and connect them with the above high-level security goals.

—**Security curriculum should include substantial materials for educating and training future generations of security researchers and practitioners with a systematic body of knowledge in security metrics.** This has been largely hindered by the lack of systematic treatment. We hope to see more curriculum materials on security metrics.

—**Security metrics research should be proceeded based on productive collaboration between the government, industry, and academia.** Only leading by one party, either the government or industry/academia, cannot achieve developing a generic security metric framework that can be widely accepted and used by research community. In particular, while academic researchers tend to be obligated to propose *what to measure*, they often encounter the lack of real data for verification and validation of their proposed metrics. The industry may have datasets but is often prohibited from sharing them with academic researchers because of legitimate concerns (e.g., privacy). Although the government has already incentivized data sharing through projects such as PREDICT (www.predict.org), our research experience hints that semantically richer data is imperative for tackling the problem of security metrics. Thus, the productive collaboration between these three parties is the first step for security metrics research to be fruitful in a meaningful way.

9. CONCLUSION

This article conducted a systematic survey of systems security metrics. Based on the in-depth survey and discussions related to each metric, we proposed the following dimensions of metrics to represent an overall system security metric: vulnerabilities, defenses, attacks, and situations. In particular, the situation metric is centered on the current security state of a given system at a particular time in order to consider dynamics related to system security states including the level of vulnerabilities, attacks, and system defenses. We discussed the gaps between the state-of-the-art metrics and the desirable metrics when discussing each dimension of these metrics. We also raised fundamental challenges that must be adequately addressed for bridging these gaps and resolving the current challenges, including what academia, industry and government should do in order to catalyze research in security metrics domain.

ACKNOWLEDGMENTS

We thank the reviewers for their constructive comments that have guided us in improving the article. We thanks Steven Noel and Xinming Ou for sending us pointers to articles. The views and opinions of the author(s) do not reflect those of the Army, NSF, ASD (R&E), or DoD.

REFERENCES

M. Ahmed, E. Al-Shaer, and L. Khan. 2008. A novel quantitative approach for measuring network security. In *IEEE INFOCOM'2008*.

E. Al-Shaer, L. Khan, and M. Ahmed. 2008. A comprehensive objective network security metric framework for proactive security configuration. In *Proc. CSIIRW'08*. 42:1–42:3.

M. Albanese, S. Jajodia, and S. Noel. 2012. Time-efficient and cost-effective network hardening using attack graphs. In *Proc. IEEE DSN'12*. 1–12.

R. Albert and A. Barabasi. 2002. Statistical mechanics of complex networks. *Rev. Mod. Phys.* 74 (2002), 47–97.

P. Ammann, D. Wijesekera, and S. Kaushik. 2002. Scalable, graph-based network vulnerability analysis. In *Proc. ACM CCS'02*. 217–224.

S. Axelsson. 2009. The base-rate fallacy and its implications for the difficulty of intrusion detection. In *Proc. ACM CCS'09*. 1–7.

M. Backes and S. Nürnberger. 2014. Oxymoron: Making fine-grained memory randomization practical by allowing code sharing. In *Proc. USENIX Security Symposium*. 433–447.

S. Berinato. 2002. Finally, a Real Return on Security Spending. Retrieved from http://www.cio.com/article/2440999/metrics/finally- -a-real-return-on-security-spending.html. (2002).

B. Biggio, G. Fumera, and F. Roli. 2014. Security evaluation of patternclassifiers under attack. *IEEE Trans. Knowl. Data Eng.* 26, 4 (2014), 984–996.

L. Bilge and T. Dumitras. 2012. Before we knew it: An empirical study of zero-day attacks in the real world. In *Proc. ACM CCS'12*. 833–844.

N. Boggs, S. Du, and S. Stolfo. 2014. Measuring drive-by download defense in depth. In *Proc. RAID'14*. 172–191.

N. Boggs and S. Stolfo. 2011. ALDR: A new metric for measuring effective layering of defenses. In *Proc. Layered Assurance Workshop (LAW'11)* (2011).

R. Böhme and F. Freiling. 2008. *Dependability Metrics: Advanced Lectures*, 7–13.

R. Böhme and T. Nowey. 2008. Dependability metrics. Chapter Economic Security Metrics, 176–187.

J. Bonneau. 2012a. The science of guessing: Analyzing an anonymized corpus of 70 million passwords. In *Proc. IEEE Symposium on Security and Privacy*. 538–552.

J. Bonneau. 2012b. Statistical metrics for individual password strength. In *Proc. International Conference on Security Protocols*. 76–86.

N. Burow, S. Carr, S. Brunthaler, M. Payer, J. Nash, P. Larsen, and M. Franz. 2016. Control-flow integrity: Precision, security, and performance. *CoRR* abs/1602.04056 (2016).

W. Burr, D. Dodson, and W. Polk. 2006. *Electronic Authentication Guideline*. NIST Publication 800-63 Version 1.0.2. Retrieved from http://csrc.nist.gov/publications/nistpubs/800-63/SP800-63V1_0_2.pdf.

A. Cardenas, J. Baras, and K. Seamon. 2006. A framework for the evaluation of intrusion detection systems. In *Proc. IEEE 2006 Symposium on Security and Privacy*.

L. Carin, G. Cybenko, and J. Hughes. 2008. Cybersecurity strategies: The QuERIES methodology. *IEEE Comput.* 41, 8 (2008), 20–26.

N. Carlini, A. Barresi, M. Payer, D. Wagner, and T. Gross. 2015. Control-flow bending: On the effectiveness of control-flow integrity. In *24th USENIX Security Symposium*. 161–176.

N. Carlini and D. Wagner. 2014. ROP is still dangerous: Breaking modern defenses. In *Proc. USENIX Security Symposium*. 385–399.

X. De Carné De Carnavalet and M. Mannan. 2015. A large-scale evaluation of high-impact password strength meters. *ACM TISSEC* 18, 1 (May 2015), 1:1–1:32.

C. Castelluccia, M. Dürmuth, and D. Perito. 2012. Adaptive password-strength meters from markov models. In *Proc. NDSS'12*.

V. Chandola, A. Banerjee, and V. Kumar. 2009. Anomaly detection: A survey. *ACM Comput. Surv.* 41, 3 (2009), 15:1–15:58.

Z. Chen and C. Ji. 2007. Measuring network-aware worm spreading ability. In *Proc. INFOCOM'2007*. 116–124.

P. Cheng, L. Wang, S. Jajodia, and A. Singhal. 2012. Aggregating CVSS base scores for semantics-rich network security metrics. In *Proc. IEEE SRDS'12*. 31–40.

Y. Cheng, J. Deng, J. Li, S. DeLoach, A. Singhal, and X. Ou. 2014. Metrics of security. In *Cyber Defense and Situational Awareness*. Vol. 62.

E. Chew, M. Swanson, K. Stine, N. Bartol, A. Brown, and W. Robinson. 2008. *NIST Special Publication 800-55 Revision 1: Performance Measurement Guide for Information Security*.

J. Cho, H. Cam, and A. Oltramari. 2016. Effect of personality traits on trust and risk to phishing vulnerability: Modeling and analysis. In *Proc. IEEE CogSIMA'16*.

CIS. 2010. The CIS Security Metrics (ver 1.1.0). Retrieved from http://benchmarks.cisecurity.org/downloads/metrics/. (2010).

INFOSEC Research Council. 2007. Hard Problem List. Retrieved from http://www.infosec-research.org/docs_public/20051130-IRC-HPL-FINAL.pdf. (2007).

G. Da, M. Xu, and S. Xu. 2014. A new approach to modeling and analyzing security of networked systems. In *Proc. HotSoS'14*.

M. Dacier, Y. Deswarte, and M. Kaâniche. 1996. Models and tools for quantitative assessment of operational security. In *Proc. IFIP Security Conference*. 177–186.

D. Dagon, G. Gu, C. Lee, and W. Lee. 2007. A taxonomy of botnet structures. In *Proc. ACSAC'07*. 325–339.

D. Dagon, C. Zou, and W. Lee. 2006. Modeling botnet propagation using time zones. In *Proc. NDSS'06*.

N. Dalvi, P. Domingos, Mausam, S. Sanghai, and D. Verma. 2004. Adversarial classification. In *Proc. KDD'04*. 99–108.

L. Davi, A. Sadeghi, D. Lehmann, and F. Monrose. 2014. Stitching the gadgets: On the ineffectiveness of coarse-grained control-flow integrity protection. In *Proc. USENIX Security Symposium*. 401–416.

M. DellAmico, P. Michiardi, and Y. Roudier. 2010. Password strength: An empirical analysis. In *Proc. INFOCOM'10*. 983–991.

Y. Desmedt and Y. Frankel. 1989. Threshold cryptosystems. In *Proc. Crypto*. 307–315.

Z. Durumeric, J. Kasten, D. Adrian, J. Halderman, M. Bailey, F. Li, N. Weaver, J. Amann, J. Beekman, M. Payer, and V. Paxson. 2014. The matter of heartbleed. In *Proc. ACM IMC'14*. 475–488.

B. Edwards, S. Hofmeyr, and S. Forrest. 2015. Hype and heavy tails: A closer look at data breaches. In *Proc WEIS'15*. 67–78.

T. Eskridge, M. Carvalho, E. Stoner, T. Toggweiler, and A. Granados. 2015. VINE: A cyber emulation environment for MTD experimentation. In *Proc. ACM MTD'15*. 43–47.

FIRST. 2015. Forum of Incident Response and Security Teams: Common Vulnerability Scoring System (CVSS) Version 3.0. Retrieved from https://www.first.org/cvss. (2015).

S. Frei and T. Kristensen. Feb. 2010. The Security Exposure of SOoftware Portfolios. Retrieved from https://secunia.com/gfx/pdf/Secunia_RSA_Software_Portfolio_Security_Exposure.pdf. (Feb. 2010).

M. Frigault and L. Wang. 2008. Measuring network security using bayesian network-based attack graphs. In *Proc. IEEE CompSAC'08*. 698–703.

M. Frigault, L. Wang, A. Singhal, and S. Jajodia. 2008. Measuring network security using dynamic bayesian network. In *Proc. QoP'08*. 23–30.

J. Gaffney Jr and J. Ulvila. 2001. Evaluation of intrusion detectors: A decision theory approach. In *Proc. IEEE Symposium on Security and Privacy*. 50–61.

E. Göktas, E. Athanasopoulos, H. Bos, and G. Portokalidis. 2014. Out of control: Overcoming control-flow integrity. In *Proc. IEEE Security and Privacy*. 575–589.

L. Gordon and M. Loeb. 2006. Budgeting process for information security expenditures. *Commun. ACM* 49, 1 (Jan. 2006), 121–125.

G. Gu, A. Cárdenas, and W. Lee. 2008. Principled reasoning and practical applications of alert fusion in intrusion detection systems. In *Proc. ACM ASIACCS'08*. 136–147.

G. Gu, P. Fogla, D. Dagon, W. Lee, and B. Skorić. 2006. Measuring intrusion detection capability: An information-theoretic approach. In *Proc. AsiaCCS'06*. 90–101.

Y. Han, W. Lu, and S. Xu. 2014. Characterizing the power of moving target defense via cyber epidemic dynamics. In *Proc. HotSoS'14*. 10:1–10:12.

S. Hardy, M. Crete-Nishihata, K. Kleemola, A. Senft, B. Sonne, G. Wiseman, P. Gill, and R. Deibert. 2014. Targeted threat index: Characterizing and quantifying politically-motivated targeted malware. In *Proc. USENIX Security Symposium*.

W. Herlands, T. Hobson, and P. Donovan. 2014. Effective entropy: Security-centric metric for memory randomization techniques. In *Workshop on Cyber Security Experimentation and Test*.

H. Holm. 2014. A large-scale study of the time required to compromise a computer system. *IEEE TDSC* 11, 1 (2014), 2–15.

J. Homer, S. Zhang, X. Ou, D. Schmidt, Y. Du, S. Rajagopalan, and A. Singhal. 2013. Aggregating vulnerability metrics in enterprise networks using attack graphs. *J. Comput. Secur.* 21, 4 (2013), 561–597.

A. Howe, I. Ray, M. Roberts, M. Urbanska, and Z. Byrne. 2012. The psychology of security for the home computer user. In *IEEE Symp. on Security and Privacy*. 209–223.

L. Huang, A. Joseph, B. Nelson, B. Rubinstein, and J. Tygar. 2011. Adversarial machine learning. In *Proc. ACM AISec'11*. 43–58.

N. Idika and B. Bhargava. 2012. Extending attack graph-based security metrics and aggregating their application. *IEEE TDSC* 9, 1 (Jan. 2012), 75–85.

W. Jansen. 2009. Directions in Security Metrics Research. Retrieved from http://csrc.nist.gov/publications/nistir/ir7564/nistir-7564_metrics-research.pdf. (2009).

A. Jaquith. 2007. *Security Metrics: Replacing Fear, Uncertainty, and Doubt*. Addison-Wesley Professional.

S. Jha, O. Sheyner, and J. Wing. 2002. Two formal analys s of attack graphs. In *Proc. IEEE CSF*. 49–59.

B. Johnson, J. Chuang, J. Grossklags, and N. Christin. 2012. Metrics for measuring ISP badness: The case of spam. In *Proc. FC'12*. 89–97.

E. Jonsson and T. Olovsson. 1997. A quantitative model of the security intrusion process based on attacker behavior. *IEEE Trans. SE* 23, 4 (1997), 235–245.

P. Kelley, S. Komanduri, M. Mazurek, and R. Shay. 2012. Guess again (and again and again): Measuring password strength by simulating password-cracking algorithms. In *Proc. IEEE Symposium on Security and Privacy*. 523–537.

M. Konte, R. Perdisci, and N. Feamster. 2015. ASwatch: An as reputation system to expose bulletproof hosting ases. In *Proc. ACM SIGCOMM'15*. 625–638.

M. Kührer, C. Rossow, and T. Holz. 2014. Paint it black: Evaluating the effectiveness of malware blacklists. In *Proc. RAID'14*. 1–21.

B. Lampson. 2006. *Practical Principles for Computer Security*. (2006).

C. Landwehr, A. Bull, J. McDermott, and W. Choi. 1994. A taxonomy of computer program security flaws. *ACM Comput. Surv.* 26, 3 (1994), 211–254.

P. Larsen, A. Homescu, S. Brunthaler, and M. Franz. 2014. SoK: Automated software diversity. In *Proc. 2014 IEEE Symposium on Security and Privacy*. 276–291.

W. Lee, W. Fan, M. Miller, S. Stolfo, and E. Zadok. 2002. Toward cost-sensitive modeling for intrusion detection and response. *J. Comput. Secur.* 10, 1–2 (July 2002), 5–22.

E. LeMay, M. Ford, K. Keefe, W. Sanders, and C. Muehrcke. 2011. Model-based security metrics using adversary view security evaluation (ADVISE). In *Proc. QEST'11*. 191–200.

L. Levesque, N. Fanny, J. Fernandez, S. Chiasson, and A. Somayaji. 2013. A clinical study of risk factors related to malware infections. In *Proc. ACM CCS'13*. 97–108.

D. Levin. 2003. Lessons learned in using live red teams in IA experiments. In *Proc. DISCEX-III*. 110–119.

X. Li, P. Parker, and S. Xu. 2011. A stochastic model for quantitative security analysis of networked systems. *IEEE TDSC* 8, 1 (2011), 28–43.

R. Lippmann, J. Riordan, T. Yu, and K. Watson. 2012. *Continuous Security Metrics for Prevalent Network Threats: Introduction and First Four Metrics*. Technical Report IA-3. MIT Lincoln Laboratory. Retrieved from https://www.ll.mit.edu/mission/cybersec/publications/publication-files/full_papers/2012_05_22_Lippmann_TechReport_FP.pdf.

Y. Liu and H. Man. 2005. Network vulnerability assessment using Bayesian networks. In *Data Mining, Intrusion Detection, Information Assurance, and Data Networks Security 2005*, B. V. Dasarathy (Ed.), Vol. 5812. 61–71.

Y. Liu, A. Sarabi, J. Zhang, P. Naghizadeh, M. Karir, M. Bailey, and M. Liu. 2015. Cloudy with a chance of breach: Forecasting cyber security incidents. In *USENIX Security Symposium*. 1009–1024.

D. Lowd and C. Meek. 2005. Adversarial learning. In *KDD'05*. 641–647.

K. Lu, C. Song, B. Lee, S. Chung, T. Kim, and W. Lee. 2015. ASLR-guard: Stopping address space leakage for code reuse attacks. In *Proc. ACM CCS'15*. 280–291.

W. Lu, S. Xu, and X. Yi. 2013. Optimizing active cyber defense dynamics. In *Proc. GameSec'13*. 206–225.

B. Madan, K. Gogeva-Popstojanova, K. Vaidyanathan, and K. Trivedi. 2002. Modeling and quantification of security attributes of software systems. In *Proc. DSN'02*. 505–514.

P. Manadhata and J. Wing. 2011. An attack surface metric. *IEEE Trans. Software Eng.* 37, 3 (2011), 371–386.

W. Marczak, J. Scott-Railton, M. Marquis-Boire, and V. Paxson. 2014. When governments hack opponents: A look at actors and technology. In *Proc. USENIX Security Symposium*. 511–525.

J. McHugh. 2006. Quality of protection: Measuring the unmeasurable? In *Proc QoP'06*. 1–2.

G. Mezzour, K. Carley, and L. Carley. 2015. An empirical study of global malware encounters. In *Proc. HotSoS'15*. 8:1–8:11.

Microsoft. 2013-2014. Security Intelligence Report. Retrieved from http://www.microsoft.com/security/sir/default.aspx. (2013-2014).

A. Milenkoski, M. Vieira, S. Kounev, A. Avritzer, and B. Payne. 2015. Evaluating computer intrusion detection systems: A survey of common practices. *ACM Comput. Surv.* 48, 1 (2015).

MITRE. 2014. Common Weakness Scoring System (CWSS version 1.0.1). Retrieved from https://cwe.mitre.org/cwss/cwss_v1.0.1.html. (2014).

A. Mohaisen and O. Alrawi. 2014. Av-meter: An evaluation of antivirus scans and labels. In *Proc. DIMVA'2014*. 112–131.

J. Morales, S. Xu, and R. Sandhu. 2012. Analyzing malware detection eciency with multiple anti-malware programs. *ASE Sci. J.* 1, 2 (2012), 56–66.

A. Nappa, R. Johnson, L. Bilge, J. Caballero, and T. Dimitras. 2015. The attack of the clones: A study of the impact of shared code on vulnerability patching. In *Proc. IEEE Symposium on Security and Privacy*.

K. Nayak, D. Marino, P. Efstathopoulos, and T. Dumitras. 2014. Some vulnerabilities are different than others. In *Proc. RAID'14*. 426–446.

A. Neupane, M. Rahman, N. Saxena, and L. Hirshfield. 2015. A multi-modal neuro-physiological study of phishing detection and malware warnings. In *Proc. ACM CCS'15*. 479–491.

D. Nicol, B. Sanders, J. Katz, B. Scherlis, T. Dumitra, L. Williams, and M. Singh. 2015. The Science of Security 5 Hard Problems. Retrieved from http://cps-vo.org/node/21590. (2015).

D. Nicol, W. Sanders, and K. Trivedi. 2004. Model-based evaluation: From dependability to security. *IEEE TDSC* 1, 1 (2004), 48–65.

B. Niu and G. Tan. 2015. Per-input control-flow integrity. In *Proc. CCS'15*. 914–926.

S. Noel and S. Jajodia. 2014. Metrics suite for network attack graph analytics. In *Proc. CISR'14*. 5–8.

R. Ortalo, Y. Deswarte, and M. Kaâniche. 1999. Experimenting with quantitative evaluation tools for monitoring operational security. *IEEE Trans. Softw. Eng.* 25, 5 (Sept. 1999), 633–650.

X. Ou and A. Singhal. 2011. *Quantitative Security Risk Assessment of Enterprise Networks*. Springer.

J. Pamula, S. Jajodia, P. Ammann, and V. Swarup. 2006. A weakest-adversary security metric for network configuration security analysis. In *Proc. ACM QoP'06*. 31–38.

S. Pfleeger and R. Cunningham. 2010. Why measuring security is hard. *IEEE Secur. Priv.* 8, 4 (July 2010), 46–54.

S. Pfleeger. 2009. Useful Cybersecurity Metrics. *IT Profess.* 11, 3 (2009), 38–45.

C. Phillips and L. Swiler. 1998. A graph-based system for network-vulnerability analysis. In *Proc. NSPW'98*. 71–79.

A. Prakash and M. Wellman. 2015. Empirical game-theoretic analysis for moving target defense. In *Proc. ACM MTD'15*. 57–65.

M. Rajab, F. Monrose, and A. Terzis. 2005. On the effectiveness of distributed worm monitoring. In *Proc. USENIX Security Symposium*.

M. Reiter and S. Stubblebine. 1999. Authentication metric analysis and design. *ACM TISSEC* 2, 2 (May 1999), 138–158.

R. Ritchey and P. Ammann. 2000. Using model checking to analyze network vulnerabilities. In *Proc. IEEE Symposium on Security and Privacy*. 156–165.

F. Roberts. 1979. *Measurement Theory, with Applications to Decision Making, Utility and the Social Sciences*. Addison-Wesley, Boston.

K. Roundy and B. Miller. 2013. Binary-code obfuscations in prevalent packer tools. *ACM Comput. Surv.* 46, 1 (July 2013), 4:1–4:32.

C. Sabottke, O. Suciu, and T. Dumitras. 2015. Vulnerability disclosure in the age of social media: Exploiting twitter for predicting real-world exploits. In *Proc. USENIX Security Symposium*. 1041–1056.

W. Sanders. 2014. Quantitative security metrics: Unattainable holy grail or a vital breakthrough within our reach? *IEEE Secur. Priv.* 12, 2 (2014), 67–69.

B. Schneier. 2000. *Secrets & Lies: Digital Security in a Networked World*. John Wiley & Sons, Inc.

National Science and Technology Council. 2011. Trustworthy Cyberspace: Strategic Plan for the Federal Cybersecurity Research and Development Program. Retrieved from https://www.nitrd.gov/SUBCOMMITTEE/csia/Fed_Cybersecurity_RD_Strategic_Plan_2011.pdf. (2011).

H. Shacham, M. Page, B. Pfaff, E. Goh, N. Modadugu, and D. Boneh. 2004. On the effectiveness of address-space randomization. In *Proc. ACM CCS'04*.

S. Sheng, M. Holbrook, P. Kumaraguru, L. Cranor, and J. Downs. 2010. Who falls for phish? A demographic analysis of phishing susceptibility and effectiveness of interventions. In *Proc. CHI'10*. 373–382.

O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing. 2002. Automated generation and analysis of attack graphs. In *IEEE Symp. on Security and Privacy*. 273–284.

A. Singhal and X. Ou. 2011. Security Risk Analysis of Enterprise Networks Using Probabilistic Attack Graphs. National Institute of Standards and Technology. Retrieved from http://csrc.nist.gov/publications/nistir/ir7788/NISTIR-7788.pdf.

K. Snow, F. Monrose, L. Davi, A. Dmitrienko, C. Liebchen, and A. Sadeghi. 2013. Just-in-time code reuse: On the effectiveness of fine-grained address space layout randomization. In *Proc. IEEE Symposium on Security and Privacy*. 574–588.

N. Stakhanova, C. Strasburg, S. Basu, and J. Wong. 2012. Towards cost-sensitive assessment of intrusion response selection. *J. Comput. Secur.* 20, 2–3 (2012), 169–198.

S. Stevens. 1946. On the theory of scales of measurement. (1946).

S. Stolfo, W. Fan, W. Lee, A. Prodromidis, and P. Chan. 2000. Cost-based modeling for fraud and intrusion detection: Results from the JAM project. In *Proc. DISCEX'00*. 130–144.

B. Stone-Gross, C. Kruegel, K. Almeroth, A. Moser, and E. Kirda. 2009. FIRE: FInding rogue networks. In *Proc. ACSAC'09*. 231–240.

C. Strasburg, N. Stakhanova, S. Basu, and J. Wong. 2009. A framework for cost sensitive assessment of intrusion response selection. In *Proc. COMPSAC'09*. 355–360.

Adrian Tang, Simha Sethumadhavan, and Salvatore Stolfo. 2015. Heisenbyte: Thwarting memory disclosure attacks using destructive code reads. In *Proc. ACM CCS'15*. 256–267.

M. Tavallaee, N. Stakhanova, and A. Ghorbani. 2010. Toward credible evaluation of anomaly-based intrusion-detection methods. *IEEE Trans. Syst, Man Cybernet. C* 40, 5 (2010), 516–524.

U. Thakore. 2015. A quantitative methodology for evaluating and deploying security monitors. Retrieved from https://www.ideals.illinois.edu/handle/2142/88103. (2015).

X. Ugarte-Pedrero, D. Balzarotti, I. Santos, and P. Bringas. 2015. SoK: Deep packer inspection: A longitudinal study of the complexity of run-time packers. In *2015 IEEE Symposium on Security and Privacy*. 659–673.

B. Ur, P. Kelley, S. Komanduri, J. Lee, M. Maass, M. Mazurek, T. Passaro, R. Shay, T. Vidas, L. Bauer, N. Christin, and L. Cranor. 2012. How does your password measure up? The effect of strength meters on password creation. In *Proc. USENIX Security Symposium*.

B. Ur, S. Segreti, L. Bauer, N. Christin, L. Cranor, S. Komanduri, D. Kurilova, M. Mazurek, William Melicher, and Richard Shay. 2015. Measuring real-world accuracies and biases in modeling password guessability. In *Proc. 24th USENIX Security Symposium*.

P. Velleman and L. Wilkinson. 1993. Nominal, ordinal, interval, and ratio typologies are misleading. *Am. Stat.* 47, 1 (1993), 65–72.

V. Verendel. 2009. Quantified security is a weak hypothesis: A critical survey of results and assumptions. In *Proc. NSPW'09*. 37–50.

C. Villarrubia, E. Fernandez-Medina, and M. Piattini. 2004. Towards a classification of security metrics. In *Proc. ICEIS'04*. 341–350.

Nedim Šrndic and Pavel Laskov. 2014. Practical evasion of a learning-based classifier: A case study. In *Proc. IEEE Symposium on Security and Privacy*. 197–211.

I. Wagner and D. Eckhoff. 2015. *Technical Privacy Metrics: A Systematic Survey*. Technical Report 1512.00327. arXiv. Retrieved from http://arxiv.org/abs/1512.00327 arXiv: 1512.00327.

L. Wang, T. Islam, T. Long, A. Singhal, and S. Jajodia. 2008. An attack graph-based probabilistic security metric. In *Proc. IFIP Conf. on Data and App. Security*. 283–296.

L. Wang, S. Jajodia, A. Singhal, and S. Noel. 2010. k-zero day safety: Measuring the security risk of networks against unknown attacks. In *Proc. ESORICS'10*. 573–587.

H. Wei, D. Frinke, O. Carter, and C. Ritter. 2001. Cost-benefit analysis for network intrusion detection systems. In *Proceedings of the 28th Annual Computer Security Conference*. Washington, D.C.

M. Weir, S. Aggarwal, M. Collins, and H. Stern. 2010. Testing metrics for password creation policies by attacking large sets of revealed passwords. In *Proc. ACM CCS'10*. 162–175.

L. Xu, Z. Zhan, S. Xu, and K. Ye. 2014b. An evasion and counter-evasion study in malicious websites detection. In *Proc. IEEE CNS'14*. 265–273.

M. Xu, G. Da, and S. Xu. 2015a. Cyber epidemic models with dependences. *Internet Math.* 11, 1 (2015), 62–92.

M.Xu and S. Xu. 2012. An extended stochastic model for quantitative security analysis of networked systems. *Internet Math.* 8, 3 (2012), 288–320.

S. Xu. 2014a. Cybersecurity dynamics. In *Proc. HotSoS'14*. 14:1–14:2.

S. Xu. 2014b. Emergent behavior in cybersecurity. In *Proc. HotSoS'14*. 13:1–13:2.

S. Xu, W. Lu, and H. Li. 2015b. A stochastic model of active cyber defense dynamics. *Internet Math.* 11, 1 (2015), 23–61.

S. Xu, W. Lu, and L. Xu. 2012a. Push- and pull-based epidemic spreading in arbitrary networks: Thresholds and deeper insights. *ACM TAAS* 7, 3 (2012), 32:1–32:26.

S. Xu, W. Lu, L. Xu, and Z. Zhan. 2014a. Adaptive epidemic dynamics in networks: Thresholds and control. *ACM TAAS* 8, 4 (2014), 19.

S. Xu, W. Lu, and Z. Zhan. 2012b. A stochastic model of multivirus dynamics. *IEEE Trans. Depend. Secure Comput.* 9, 1 (2012), 30–45.

D. Yardon. May 4, 2014. Symantec Develops New Attack on Cyberhacking. Retrieved from http://www.wsj.com/articles/SB10001424052702303417104579542140235850578. (May 4, 2014).

T. Yen, V. Heorhiadi, A. Oprea, M. Reiter, and A. Juels. 2014. An epidemiological study of malware encounters in a large enterprise. In *Proc. ACM CCS'14*. 1117–1130.

S. Yilek, E. Rescorla, H. Shacham, B. Enright, and S. Savage. 2009. When private keys are public: Results from the 2008 debian OpenSSL vulnerability. In *Proc. ACM IMC'09*. 15–27.

K. Zaffarano, J. Taylor, and S. Hamilton. 2015. A quantitative framework for moving target defense effectiveness evaluation. In *Proc. ACM MTD'15*. 3–10.

Z. Zhan, M. Xu, and S. Xu. 2014. A characterization of cybersecurity posture from network telescope data. In *Proc. InTrust'14*. 105–126.

J. Zhang, Z. Durumeric, M. Bailey, M. Liu, and M. Karir. 2014. On the mismanagement and maliciousness of networks. In *Proc. NDSS'14*.

M. Zhang, L. Wang, S. Jajodia, A. Singhal, and M. Albanese. 2016. Network diversity: A security metric for evaluating the resilience of networks against zero-day attacks. *IEEE T-IFS* 11, 5 (May 2016), 1071–1086.

S. Zhang, X. Zhang, and X. Ou. 2014. After we knew it: Empirical study and modeling of cost-effectiveness of exploiting prevalent known vulnerabilities across IaaS cloud. In *Proc. ACM AsiaCCS*. 317–328.

R. Zheng, W. Lu, and S. Xu. 2015. Active cyber defense dynamics exhibiting rich phenomena. In *Proc. HotSoS'15*. 2:1–2:12.

Received January 2016; revised October 2016; accepted October 2016