Metrics for Continuous Active Defence

George O.M. Yee
Computer Research Lab, Aptusinnova Inc., Ottawa, Canada
Dept. of Systems and Computer Engineering, Carleton University, Ottawa, Canada
email: george@aptusinnova.com, gmyee@sce.carleton.ca

***Abstract*—As a sign of the times, headlines today are full of attacks against an organization's computing infrastructure, resulting in the theft of sensitive data. In response, the organization applies security measures (e.g., encryption) to secure its vulnerabilities. However, these measures are often only applied once, with the assumption that the organization is then protected and no further action is needed. Unfortunately, attackers continuously probe for vulnerabilities and change their attacks accordingly. This means that an organization must also continuously check for new vulnerabilities and secure them, to continuously and actively defend against the attacks. This paper derives metrics that characterize the security level of an organization at any point in time, based on the number of vulnerabilities secured and the effectiveness of the securing measures. The paper then shows how an organization can apply the metrics for continuous active defence.**

Keywords- sensitive data; vulnerability; security measure; security level; metrics; continuous defence.

I. INTRODUCTION

Headlines today are full of news of attacks against computing infrastructure, resulting in sensitive data being compromised. These attacks have devastated the victim organizations. The losses have not only been financial (e.g., theft of credit card information), but perhaps more importantly, have damaged the organizations' reputation. Consider, for example, the following data breaches that occurred in 2017 [1]:

- March, 2017, Dun & Bradstreet: This business services company found its marketing database with over 33 million corporate contacts shared across the web. The company claimed that the breach occurred to businesses, numbering in the thousands, that had bought its 52 GB database. The leak may have included full names, work email addresses, phone numbers, and other business-related data from millions of employees of organizations such as the US Department of Defence, the US Postal Service, AT&T, Walmart, and CVS Health.
- September, 2017, Equifax: This is one of the three largest credit agencies in the US. It announced a breach that may have affected 143 million customers, one of the worst breaches ever due to the sensitivity of the data stolen. The compromised data included social security numbers, driver's license numbers, full names, addresses, birth dates, credit card numbers, and other personal information. Hackers had access to the company's system from mid-May to July by exploiting a vulnerability in website software. Equifax discovered the breach on July 29, 2017.

There were many more breaches in 2017, and in fact, no year can be said to have been breach-free. Moreover, the problem appears to be getting worst, as 2017 has been mentioned [2] as a "record-breaking year" for data breaches: a total of 5,207 breaches and 7.89 billion information records compromised.

In response to attacks, such as the ones described above, organizations determine their computer system vulnerabilities and secure them using security measures. Typical measures include firewalls, intrusion detection systems, two-factor authentication, encryption, and training for employees on identifying and resisting social engineering. However, once the security measures have been implemented, organizations tend to believe that they are safe and that no further actions are needed. Unfortunately, attackers do not give up just because the organization has secured its known computer vulnerabilities. Rather, the attackers will continuously probe the organization's computer system for new vulnerabilities that they can exploit. This means that the organization must continuously analyze its computer system vulnerabilities and secure any new ones that it discovers. In order to do this effectively, it is useful to have quantitative metrics of the security level at any particular point in time, based on the number of vulnerabilities secured and the effectiveness of the security measures, at that point in time. An acceptable security level can be set, so that if the security level falls below this acceptable level due to new vulnerabilities, the latter can be secured to bring the security level back to the acceptable level. This work derives such metrics and shows how to apply them for continuous active defence, i.e., continuous vulnerabilities evaluation and follow up.

The objectives of this work are: i) derive straightforward, clear metrics of the resultant protection level obtained by an organization at any point in time, based on the use of security measures to secure vulnerabilities and the effectiveness of the measures, ii) show how these metrics can be calculated, iii) show how the metrics can be applied for continuous active defence. We seek straightforward, easy to understand metrics since complicated, difficult to understand ones tend not to be used or tend to be misapplied. We base these metrics on securing vulnerabilities since this has been and continues to be the method organizations use to secure their computer infrastructure.

The rest of this paper is organized as follows. Section II discusses sensitive data, attacks, and vulnerabilities. Section III derives the metrics and presents various aspects of the metrics, including some of their strengths, weaknesses, and limitations. Section IV explains how to apply the metrics for continuous active defence. Section V discusses related work and Section VI gives conclusions and future research.

II. Sensitive Data, Attacks, and Vulnerabilities

Sensitive data is data that needs protection and must not fall into the wrong hands. It includes private or personal information [3], which is information about an individual, can identify that individual, and is owned by that individual. For example, an individual's height, weight, or credit card number can all be used to identify the individual and are considered as personal information or personal sensitive data. Sensitive data also includes non-personal information that may compromise the competitiveness of the organization if divulged, such as trade secrets or proprietary algorithms and formulas. For government organizations, non-personal sensitive data may include information that is vital for the security of the country for which the government organization is responsible.

DEFINITION 1: *Sensitive data* (SD) is information that must be protected from unauthorized access in order to safeguard the privacy of an individual, the well-being or expected operation of an organization, or the well-being or expected functioning of an entity for which the organization has responsibility.

DEFINITION 2: An *attack* is any action carried out against an organization's computer system that, if successful, compromises the system or the SD held by the system.

An attack that compromises a computer system is Distributed Denial of Service (DDoS). One that compromises the SD held by the system is a Trojan horse attack in which malicious software (the Trojan) is planted inside the system to steal SD. Attacks can come from an organization's employees, in which case the attack is an *inside attack*. For example, a disgruntled employee secretly keeps a copy of a SD backup and sells it on the "dark web".

DEFINITION 3: A *vulnerability* of a computer system is any weakness in the system that can be targeted by an attack with some expectation of success. A vulnerability can be secured to become a *secured vulnerability* through the application of a security measure.

An example of a vulnerability is a communication channel that is used to convey sensitive data in the clear. This vulnerability can be targeted by a Man-in-the-Middle attack with reasonable success of stealing the sensitive data. This vulnerability can become a secured vulnerability by encrypting the sensitive data that the communication channel carries.

A computer system can undergo upgrades, downgrades, and other modifications over time that changes its number of secured and unsecured vulnerabilities. It is thus necessary to specify a time *t* when referring to vulnerabilities. Clearly, the number of secured and unsecured vulnerabilities of a computer system at time *t* is directly related to the security level of the system at time *t*. This idea is formalized in the next definition.

DEFINITION 4: A computer system's security level (SL) at time t, or SL(t), is the degree of protection from attacks that results from having q(t) secured vulnerabilities, and p(t) unsecured vulnerabilities, where the system has a total of N(t) = p(t)+q(t) secured and unsecured vulnerabilities. SL(t) is uniquely represented by the pair (p(t), q(t)).

Clearly SL(t) increases with increasing q(t) and decreases with increasing p(t). Figure 1 shows 3 SL(t) points on the (p(t), q(t)) plane for N(t)=100.

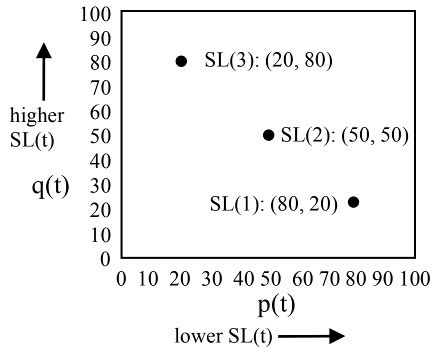

Figure 1. SL(t) points corresponding to a computer system with N(t)=100. SL(3) is higher security than SL(2), which is higher security than SL(1).

In Figure 1, the higher values of q(t) correspond to higher security levels, and the higher values of p(t) correspond to lower security levels.

III. Metrics for Continuous Active Defence

While the pair (p(t), q(t)) uniquely represents SL(t), it cannot be used to calculate the value of SL(t), which would be useful in tracking the security of a system over time as its vulnerabilities change. In this section, we derive two metrics for the value of SL(t), one assuming that the measures securing vulnerabilities are totally reliable; the other with the measures only partly reliable. Both metrics are applied right after the vulnerabilities have been determined, and possibly before any of them have actually been secured. Determining vulnerabilities is discussed in Section III.C below.

A. Metric with Totally Reliable Securing Measures

We seek a metric STRM(t) (STRM is an acronym for "SL with Totally Reliable Measures") for a computer system's SL(t), where all securing measures are totally reliable. Suppose that p(t) and q(t) are as in Definition 4. Let $P_t(e)$ represent the probability of event e at time t. Let "exploit" mean a successful attack on a vulnerability. Let "all exploits" mean exploits on 1 or more vulnerabilities. Let $U_k(t)$ denote an unsecured vulnerability k at time t. We have

$$SL(t) = P_t(\text{no exploits}) = 1\text{-}P_t(\text{all exploits}) \qquad (1)$$

However, the only exploitable vulnerabilities are the unsecured vulnerabilities since the securing measures are totally reliable. Therefore

$$P_t(\text{all exploits}) = \Sigma_k\ [P_t(\text{exploit of } U_k(t))]$$

by applying the additive rule for the union of probabilities, assuming that 2 or more exploits do not occur simultaneously. Let $u_k(t)$ be a real number with $0 < u_k(t) \le p(t)$ and $\Sigma_k u_k(t) = p(t)$. Set

$$P_t(\text{exploit of } U_k(t)) \approx u_k(t)/(p(t)+q(t)) \qquad (2)$$

By substitution using (2)

$$\begin{aligned} P_t(\text{all exploits}) &\approx \Sigma_k\ [u_k(t)/(p(t)+q(t))] \\ &= \Sigma_k u_k(t)/(p(t)+q(t)) \\ &= p(t)/(p(t)+q(t)) \qquad (3) \end{aligned}$$

The condition $0 < u_k(t) \le p(t)$ is needed to ensure that there is some probability for an unsecured vulnerability to be exploited. The condition $\Sigma_k u_k(t) = p(t)$ is necessary in order for $P_t(\text{all exploits}) \le 1$. Expression (2) gives a way of assigning values for $P_t(\text{exploit of } U_k(t))$ based on a risk analysis [3]. However, expression (3) ensures that such assignment is not needed for calculating STRM(t). In other words, the fact that some vulnerabilities are more likely to be exploited than others does not affect the value of STRM(t).

Substituting (3) into (1) gives

$$\begin{aligned} SL(t) &\approx 1\text{-}[p(t)/(p(t)+q(t))] \\ &= q(t)/(p(t)+q(t)) \quad \text{if } p(t)+q(t) > 0 \\ &= 1 \quad \text{if } p(t)+q(t) = 0 \end{aligned}$$

We obtain STRM(t) by assigning as follows:

$$\begin{aligned} \mathbf{STRM(t)} &= \mathbf{q(t)/(p(t)+q(t))} \quad \textbf{if } \mathbf{p(t)+q(t) > 0} \qquad (4) \\ &= \mathbf{1} \quad \textbf{if } \mathbf{p(t)+q(t) = 0} \qquad (5) \end{aligned}$$

We see from (4) that $0 \le STRM(t) \le 1$ if $p(t)+q(t) > 0$ and has value 0 if $q(t)=0$ (the system has no secured vulnerabilities) and 1 if $p(t)=0$ (all of its vulnerabilities are secured). We see from (5) that $STRM(t)=1$ if $p(t)+q(t)=0$ (no vulnerabilities, which is unlikely). The values of the metric are therefore as expected.

B. Metric with Partially Reliable Securing Measures

Here, we seek a metric SPRM(t) (SPRM is an acronym for "SL with Partially Reliable Measures") for a computer system's SL(t) where the measures securing the vulnerabilities are only partially reliable.

Let $V_k(t)$ denote a secured vulnerability k at time t. The reliability $r_k(t)$ of the measure securing $V_k(t)$ can be defined as the probability that the measure remains operating from time zero to time t, given that it was operating at time zero [4]. The unreliability of the measure is then $1\text{-}r_k(t)$. We have the events

[exploit of $V_k(t)$] if and only if [$V_k(t)$ selected for exploit] AND [measure securing $V_k(t)$ unreliable]

Since the two right-hand side events are independent,

$$P_t(\text{exploit of } V_k(t)) = P_t(V_k(t) \text{ selected for exploit}) \times P_t(\text{measure securing } V_k(t) \text{ unreliable})$$

Set

$$P_t(V_k(t) \text{ selected for exploit}) \approx 1/(p(t)+q(t)) \qquad (6)$$

since attackers will have no preference to attack one secured vulnerability over another secured vulnerability (they should not even see them as vulnerabilities). Again, applying the additive rule for the union of probabilities,

$$\begin{aligned} P_t(\text{all } V_k(t) \text{ exploits}) &= \Sigma_k[P_t(V_k(t) \text{ selected for exploit}) \times P_t(\text{measure securing } V_k(t) \text{ unreliable})] \\ &= \Sigma_k\ [(1/(p(t)+q(t)))(1\text{-}r_k(t))] \\ &= [\Sigma_k(1\text{-}r_k(t)]/[p(t) + q(t)] \\ &= [q(t)\text{-}\Sigma_k r_k(t)]/[p(t) + q(t)] \\ &= [q(t)/(p(t)+q(t))]\text{-}\Sigma_k r_k(t)/(p(t) + q(t)) \qquad (7) \end{aligned}$$

Now, since both $U_k(t)$ and $V_k(t)$ can be exploited,

$$\begin{aligned} P_t(\text{all exploits}) &= P_t(\text{all } U_k(t) \text{ exploits}) + P_t(\text{all } V_k(t) \text{ exploits}) \\ &\approx [p(t)/(p(t)+q(t))] + [q(t)/(p(t)+q(t))]\text{-} \Sigma_k r_k(t)/(p(t) + q(t)) \\ &= 1 - \Sigma_k r_k(t)/(p(t) + q(t)) \qquad (8) \end{aligned}$$

by substitution using (3) and (7), where (3) is $P_t(\text{all } U_k(t) \text{ exploits})$. Finally, by substitution using (1) and (8),

$$\begin{aligned} SL(t) &\approx 1 - 1 + \Sigma_k r_k(t)/(p(t) + q(t)) \\ &= \Sigma_k r_k(t)/(p(t) + q(t)) \quad \text{if } p(t) \ge 0,\ q(t) > 0 \\ &= 1 \quad \text{if } p(t)+q(t) = 0 \\ &= 0 \quad \text{if } p(t)>0,\ q(t) = 0 \end{aligned}$$

We obtain SPRM(t) by assigning as follows:

$$\begin{aligned} \mathbf{SPRM(t)} &= \mathbf{\Sigma_k r_k(t)/(p(t)+q(t))} \quad \textbf{if } \mathbf{p(t) \ge 0,\ q(t) > 0} \qquad (9) \\ &= \mathbf{1} \quad \textbf{if } \mathbf{p(t)+q(t) = 0} \qquad (10) \\ &= \mathbf{0} \quad \textbf{if } \mathbf{p(t)>0,\ q(t)=0} \qquad (11) \end{aligned}$$

We see from (9) that $0 < SPRM(t) < 1$ for $p(t) \ge 0$, $q(t) > 0$ (all vulnerabilities may or may not be secured), and from (10) that $SPRM(t) = 1$ for $p(t)+q(t) = 0$ (no vulnerabilities, which is unlikely). We see from (11) that $SPRM(t) = 0$ for $p(t)>0$, $q(t) = 0$ (no secured vulnerabilities). We also see that for $r_k(t) = 1$, SPRM(t) is the same as STRM(t). The values of the metric are therefore as expected.

C. Calculating the Metrics

Calculating STRM(t) requires the values of p(t) and q(t) at a series of time points of interest. SPRM(t) requires the values of p(t), q(t), and the reliability value for each measure used to secure the vulnerabilities.

To obtain the values of p(t) and q(t), an organization may perform a threat analysis of vulnerabilities in the organization's computer system that could allow attacks to occur. Threat analysis or threat modeling is a method for systematically assessing and documenting the security risks associated with a system (Salter et al. [5]). Threat modeling involves understanding the adversary's goals in attacking the system based on the system's assets of interest. It is predicated on that fact that an adversary cannot attack a system without a way of supplying it with data or otherwise accessing it. In addition, an adversary will only attack a system if it has some assets of interest. The method of threat analysis given in [5] or any other method of threat analysis will yield the total number N(t) of vulnerabilities to attacks at time t. Once this number is known, the organization can select which vulnerabilities to secure and which security measures to use, based on a prioritization of the vulnerabilities and the amount of budget it has to spend. A way to optimally select which vulnerabilities to secure is described in [6]. Once vulnerabilities have been selected to be secured, we have q(t). Then p(t) = N(t) – q(t). The threat analysis may be carried out by a project team consisting of the system's design manager, a security and privacy analyst, and a project leader acting as facilitator. In addition to having security expertise, the analyst must also be very familiar with the organization's computer system. Further discussion on threat analysis is outside the scope of this paper. More details on threat modeling can be found in [6]. Vulnerabilities may be prioritized using the method in [3], which describes prioritizing privacy risks.

The reliability values for hardware measures used to secure the selected vulnerabilities may be obtained from the hardware's manufacturers (e.g., hardware firewall). Reliability values for software and algorithmic measures are more difficult to obtain (e.g., encryption algorithm). For these, it may be necessary to estimate the reliability values based on the rate of progress of technology. For example, one could estimate the reliability of an encryption algorithm based on estimates of the computer resources that attackers have at their disposal. If they have access to a super computer, an older encryption algorithm may not be sufficiently reliable. One could also opt to be pessimistic and assign low reliability values, which would have the net effect of boosting security by securing more vulnerabilities, in order to meet a certain SL(t) level (see Section IV). Reliability values for security measures represent a topic for future research.

It is important to note that at each time point where the metrics are calculated, the values of p(t) and q(t) are generated anew. Vulnerabilities secured previously with totally reliable measures would not appear again as vulnerabilities. On the other hand, vulnerabilities secured with only partially reliable measures should be identified again as vulnerabilities. Further, it is not necessary to have actually implemented the securing measures before calculating the metrics.

D. Graphing the Metrics

The metrics STRM(t) and SPRM(t) are both functions of p(t), q(t), and t. Figure 2 shows a 3-dimensional graph of these metrics with axes for STRM(t)/SPRM(t), p(t), and q(t). Time is not shown explicitly as an axis since we would need 4 dimensions, but is instead represented as time period displacements of the metrics' values.

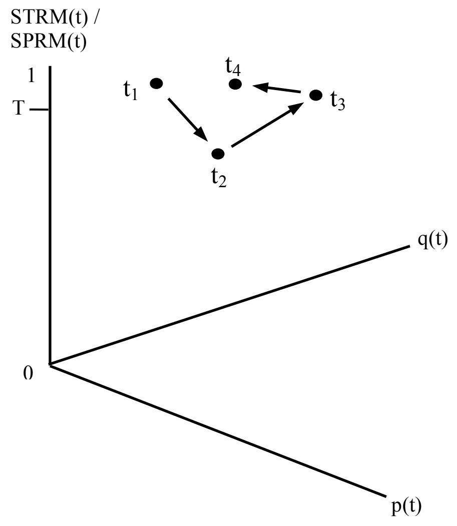

Figure 2. STRM(t)/SPRM(t) values at times $t_1 < t_2 < t_3 < t_4$.

Figure 2 shows 4 values of one of the metrics, labeled according to the times it was evaluated, namely t_1, t_2, t_3, and t_4 where $t_1 < t_2 < t_3 < t_4$. The intervals between these times may be 1 week or 1 month, for example. T is a threshold, below which the metric values should not drop (see Section IV.A). At t_1, one of the metrics was evaluated producing the value shown. At t_2, the metric was again evaluated, but this time the value was found to be much lower than at t_1, and in fact, the value dropped below T. The reason for this was that new vulnerabilities were found that had not been secured. The organization decides to secure the additional vulnerabilities. At t_3, another evaluation was carried out, and this time, the metric had improved, reaching above T. The organization finds some surplus money in its budget and decides to secure 2 other vulnerabilities. An evaluation of the metric at t_4 finds the value a little higher than at t_3, due to the 2 additional vulnerabilities secured. It is thus seen that the security level of a computer system changes over time, in accordance with the system's number of secured and unsecured vulnerabilities.

E. Strengths, Weaknesses, and Limitations

Some strengths of the metrics are: a) conceptually straightforward, and easily explainable to management, and b) flexible and powerful, i.e., they have many application areas, as described in Section IV.

Some weaknesses are: a) threat modeling to determine the vulnerabilities is time consuming and subjective, and b) the SL may involve more factors than vulnerabilities and secured vulnerabilities. For weakness a), it may be possible to automate or semi-automate the threat modeling. Related works [13] and [19] are good starting points for further research. For weakness b), it may be argued that the metrics as presented are sufficient for their envisaged application when other sources of error are considered (e.g., it is difficult to tell where an attacker will strike or how he will strike), and that adding more factors would only make the metrics unnecessarily more cumbersome and time consuming to evaluate with little additional benefit.

Some limitations of the metrics follow. First of all, the metrics are only estimates of the security level, not the security level itself. This was indicated in assigning the probabilities as approximate in expressions (2) and (6) of Section III. Second, as noted in Section III, it makes no difference to the values of the metrics whether one unsecured vulnerability is more likely to be exploited than another. This may be due to the fact that the metrics are estimating the total security of the computer system, and therefore the total number of exploitable vulnerabilities is what's important, not the order in which they are exploited. Third, we applied the additive rule for the union of probabilities in Section III, requiring that 2 or more exploits do not occur simultaneously. This condition holds in general but if it is violated, the metrics will be inaccurate. Other limitations may be that there are vulnerabilities that have not been identified, and a secured vulnerability may not in reality be secured because the attacker has a secret way of defeating the securing measure. However, these other limitations are true of other security methods as well.

IV. Application Areas

In this section, we present some applications for the metrics. In Section IV.A, we discuss how they can be used for continuous active defence of a computer system. In Section IV.B, we present other application areas, such as critical infrastructure and defence.

A. Continuous Active Defence

Attackers do not attack once, and finding that you are well protected, go away. Rather, they continuously probe your defences in order to find new vulnerabilities to exploit. It is thus necessary to continuously evaluate the computer system's vulnerabilities using threat modeling, and add additional security by securing new vulnerabilities when necessary. We call this "Continuous Active Defence" or CAD. How do we know when it is necessary to add more security? This is where the metrics can be applied. Continuous Active Defence involves the following steps:

1. Decide on a threshold for SL(t) below which the values of the metrics should not drop.
2. Decide on the frequency with which to perform threat modeling, e.g., every week, every month, exceptions.
3. Begin Continuous Active Defence by carrying out the threat modeling at the frequency decided above. After each threat modeling exercise, calculate either STRM(t) (if reliability data is not available) or SPRM(t) (if reliability data is available). If the value of the metric falls below T (see Figure 2), secure additional vulnerabilities until the value is above T.
4. If there has been a change to the system, such as new equipment or new software, do an immediate threat analysis, calculate one of the metrics, and add security if necessary based on T. Then, proceed with the frequency for threat modeling decided above.

The value of T and the frequency of threat modeling can be determined by the same threat analysis team mentioned above. The values would depend on the following:

- The potential value of the sensitive data – the more valuable the data is to a thief, a malicious entity, or a competitor, the higher the threshold and frequency should be.
- The damages to the organization that would result, if the sensitive data were compromised – of course, the higher the damages, the higher the threshold and frequency.
- The current and likely future attack climate – consider the volume of attacks and the nature of the victims, say over the last 6 months; if the organization's sector or industry has sustained a large number of recent attacks, then the threshold and frequency need to be higher.
- Consider also potential attacks by nation states as a result of the political climate; attacks by individual hacktivist groups such as Anonymous or WikiLeaks may also warrant attention.

In general, a computer system should be as secure as possible. Therefore, T above 80% and a frequency of weekly would not be uncommon. However, whatever the threshold and frequency, the organization must find them acceptable after considering the above factors. The financial budget available for securing vulnerabilities also plays an important role here, since higher thresholds call for securing more vulnerabilities, which means more financial resources will be needed.

B. Other CAD Application Areas

CAD may also be applied to a specific type of vulnerabilities. An example of this application is dealing with inside attacks. If the organization is particularly susceptible to inside attacks, it can decide to apply CAD to vulnerabilities that can be exploited for inside attacks. In this case, some of the vulnerabilities may be weaknesses of the organization itself, e.g., ineffective screening of job applicants, and the securing measures may not be technological, e.g., having an ombudsman for employee concerns. A list of questions that can be used to identify vulnerabilities to inside attack is given in [6].

CAD can be applied to a specific subset of vulnerabilities that the organization deems are crucial to its mission. For example, a cloud service provider would deem the protection of clients' data crucial to its mission. It can choose to apply CAD to vulnerabilities that are specific to its data storage capabilities, and also apply CAD to its computer system as a whole.

CAD may also be applied to code level vulnerabilities. In this case, the frequency of application will depend on how often the code is changed, due to patching and the addition or deletion of functionality. The threat modeling would have to be tailored to code and would be more of a code inspection exercise.

Finally, CAD may be applied to protect critical infrastructure and defence systems. The power grid is an example of critical infrastructure. The development of the metrics only considers vulnerabilities and reliabilities, which are also found in critical infrastructure and defence systems. However, the threat analyses would involve different types of threats, and the securing measures, would of course, need to be appropriate for the vulnerability. For example, the vulnerability of transformer sabotage in a power grid may need to be secured by the use of intrusion alarms. As another example, the vulnerability of a retaliatory missile site being preemptively destroyed may need to be secured by putting the missile on a mobile platform. The application of CAD to protect critical infrastructure and defence systems is a subject of future research.

V. Related Work

Related work found in the literature includes attack surface metrics, risk and vulnerabilities assessment, vulnerabilities classification, threat analysis, other, and this author's previous work.

A system's attack surface is related to a SL; it is proportional to the inverse of a SL since the lower the attack surface, the higher the SL. Stuckman and Purtilo [7] present a framework for formalizing code-level attack surface metrics and describe activities that can be carried out during application deployment to reduce the application's attack surface. They also describe a tool for determining the attack surface of a web application, together with a method for evaluating an attack surface metric over a number of known vulnerabilities. Munaiah and Meneely [8] propose function and file level attack surface metrics that allow fine-grained risk assessment. They claim that their metrics are flexible in terms of granularity, perform better than comparable metrics in the literature, and are tunable to specific products to better assess risk.

In terms of risk and vulnerabilities assessment, Islam et al. [9] present a risk assessment framework that starts with a threat analysis followed by a risk assessment to estimate the threat level and the impact level. This leads to an estimate of a security level for formulating high-level security requirements. The security level is qualitative, such as "low", "medium", and "high". Vanciu et al. [10] compare an architectural-level approach with a code-level approach in terms of the effectiveness of finding security vulnerabilities. Wang et al. [11] discuss their work on temporal metrics for software vulnerabilities based on the Common Vulnerability Scoring System (CVSS) 2.0. They use a mathematical model to calculate the severity and risk of a vulnerability, which is time dependent as in this work. Gawron et al. [12] investigate the detection of vulnerabilities in computer systems and computer networks. They use a logical representation of preconditions and post conditions of vulnerabilities, with the aim of providing security advisories and enhanced diagnostics for the system. Wu and Wang [13] present a dashboard for assessing enterprise level vulnerabilities that incorporates a multi-layer tree-based model to describe the vulnerability topology. Vulnerability information is gathered from enterprise resources for display automatically. Farnan and Nurse [14] describe a structured approach to assessing low-level infrastructure vulnerability in networks. The approach emphasizes a controls-based evaluation rather than a vulnerability-based evaluation. Instead of looking for vulnerabilities in infrastructure, they assume that the network is insecure, and determine its vulnerability based on the controls that have or have not been implemented. Neuhaus et al. [15] present an investigation into predicting vulnerable software components. Using a tool that mines existing vulnerability databases and version archives, mapping past vulnerabilities to current software components, they were able to come up with a predictor that correctly identifies about half of all vulnerable components, with two thirds of the predictions being correct. Roumani et al. [16] consider modeling of vulnerabilities using time series. According to these researchers, time series models provide a good fit to vulnerability datasets and can be used for vulnerability prediction. They also suggest that the level of the time series is the best estimator for prediction.

With regard to vulnerabilities classification, Spanos et al. [17] look at ways to improve CVSS. They propose a new vulnerability scoring system called the Weighted Impact Vulnerability Scoring System (WIVSS) that incorporates the different impact of vulnerability characteristics. In addition, the MITRE Corporation [18] maintains the Common Vulnerability and Exposures (CVE) list of vulnerabilities and exposures, standardized to facilitate information sharing.

In terms of threat analysis, Schaad and Borozdin [19] present an approach for automated threat analysis of software architecture diagrams. Their work gives an example of automated threat analysis. Sokolowski and Banks [20] describe the implementation of an agent-based simulation model designed to capture insider threat behavior, given a set of assumptions governing agent behavior that pre-disposes an agent to becoming a threat. Sanzgiri and Dasgupta [21] present a taxonomy and classification of insider threat detection techniques based on strategies used for detection.

The following publications fall into the other category. Kotenko and Doynikova [22] investigate the selection of countermeasures for ongoing network attacks. They suggest a selection technique based on the countermeasure model in open standards. The technique incorporates a level of countermeasure effectiveness that is related to the reliability of measures securing vulnerabilities, used in the SPRM(t) metric proposed in this work. Ganin et al. [23] present a review of probabilistic and risk-based decision-making techniques applied to cyber systems. They propose a decision-analysis-based approach that quantifies threat, vulnerability, and consequences through a set of criteria designed to assess the overall utility of cybersecurity management alternatives.

This author's directly related work includes [24] and [6], where the latter is an expanded version of the former. This work improves on these previous works by adding a) time dependency, together with the notion that an organization's security level needs to be continuously evaluated, b) a new metric incorporating the reliability of the securing measures, and c) a description of new application areas.

VI. CONCLUSIONS AND FUTURE RESEARCH

Since attackers continuously probe for new vulnerabilities to exploit, an organization cannot afford to assess its computer system's vulnerabilities once, secure some of the vulnerabilities, and then do nothing further. Rather, the organization needs to assess and secure its vulnerabilities on a continuous basis, i.e., perform CAD. This work has proposed two conceptually clear SL metrics that can be used to evaluate a computer system's security level at any point in time for CAD. One metric assumes that the measures securing vulnerabilities are totally reliable; the other considers the measures to be only partially reliable. CAD may be applied to specific types of vulnerabilities (e.g., vulnerabilities to insider attack), groupings of vulnerabilities that require special attention, specific application areas such as critical infrastructure and defence, and even at the code level.

There are many security metrics in the literature, as seen in Section V. The metrics in this work have the advantages of being easy to understand, and easy to calculate, which may be needed to convince management to provide the necessary resources required for CAD.

Future research includes formulations of other security metrics, the application of security metrics to critical infrastructure and defence, improving the methods for threat modeling, and exploring how this work may complement work in the literature and in the standardization community.

REFERENCES

[1] Identity Force, "2017 Data breaches – the worst so far," retrieved: July, 2018. https://www.identityforce.com/blog/2017-data-breaches

[2] Dark Reading, "2017 Smashed world's records for most data breaches, exposed information," retrieved: July, 2018. https://www.darkreading.com/attacks-breaches/2017-smashed-worlds-records-for-most-data-breaches-exposed-information/d/d-id/1330987?elq_mid=83109&elq_cid=1734282&_mc=NL_DR_EDT_DR_weekly_20180208&cid=NL_DR_EDT_DR_weekly_20180208&elqTrackId=700ff20d23ce4d3f984a1cfd31cb11f6&elq=5c10e9117ca04ba0ad984c11a7dfa14b&elqaid=83109&elqat=1&elqCampaignId=29666

[3] G. Yee, "Visualization and prioritization of privacy risks in software systems," International Journal on Advances in Security, issn 1942-2636, vol. 10, no. 1&2, pp. 14-25, 2017.

[4] ITEM Software Inc.,"Reliability prediction basics", retrieved: July, 2018. http://www.reliabilityeducation.com/ReliabilityPredictionBasics.pdf

[5] C. Salter, O. Saydjari, B. Schneier, and J. Wallner, "Towards a secure system engineering methodology," Proc. New Security Paradigms Workshop, pp. 2-10, 1998.

[6] G. Yee, "Optimal security protection for sensitive data," International Journal on Advances in Security, vol. 11, no. 1&2, pp. 80-90, 2018.

[7] J. Stuckman and J. Purtilo, "Comparing and applying attack surface metrics," Proceedings of the 4th International Workshop on Security Measurements and Metrics (MetriSec '12), pp. 3-6, Sept. 2012.

[8] N. Munaiah and A. Meneely, "Beyond the attack surface," Proceedings of the 2016 ACM Workshop on Software Protection (SPRO '16), pp. 3-14, October 2016.

[9] M. Islam, A. Lautenbach, C. Sandberg, and T. Olovsson, "A risk assessment framework for automotive embedded systems," Proc. 2nd ACM International Workshop on Cyber-Physical System Security (CPSS '16), pp. 3-14, 2016.

[10] R. Vanciu, E. Khalaj, and M. Abi-Antoun, "Comparative evaluation of architectural and code-level approaches for finding security vulnerabilities," Proceedings of the 2014 ACM Workshop on Security Information Workers (SIW '14), pp. 27-34, Nov. 2014.

[11] J. A. Wang, F. Zhang, and M. Xia, "Temporal metrics for software vulnerabilities," retrieved: July, 2018. http://www.cs.wayne.edu/fengwei/paper/wang-csiirw08.pdf

[12] M. Gawron, A. Amirkhanyan, F. Cheng, and C. Meinel, "Automatic vulnerability detection for weakness visualization and advisory creation," Proc. 8th International Conference on Security of Information and Networks (SIN '15), pp. 229-236, 2015.

[13] B. Wu and A. Wang, "A multi-layer tree model for enterprise vulnerability management," Proceedings of the 2011 Conference on Information Technology Education (SIGITE '11), pp. 257-262, October 2011.

[14] O. Farnan and J. Nurse, "Exploring a controls-based assessment of infrastructure vulnerability," Proc. International Conference on Risks and Security of Internet and Systems (CRiSIS 2015), pp. 144-159, 2015.

[15] S. Neuhaus, T. Zimmermann, C. Holler, and A. Zeller, "Predicting vulnerable software components," Proc. 14th ACM Conference on Computer and Communications Security (CCS '07), pp. 529-540, 2007.

[16] Y. Roumani, J. Nwankpa, and Y. Roumani, "Time series modeling of vulnerabilities," Computers and Security, Vol. 51 Issue C, pp. 32-40, June 2015.

[17] G. Spanos, A. Sioziou, and L. Angelis, "WIVSS: A new methodology for scoring information system vulnerabilities," Proc. 17th Panhellenic Conference on Informatics, pp. 83-90, 2013.

[18] MITRE, "Common vulnerabilities and exposures", retrieved: July, 2018. https://cve.mitre.org/

[19] A. Schaad and M. Borozdin, "TAM2: Automated threat analysis," Proc. 27th Annual ACM Symposium on Applied Computing (SAC '12), pp. 1103-1108, 2012.

[20] J. Sokolowski and C. Banks, "An agent-based approach to modeling insider threat," Proc. Symposium on Agent-Directed Simulation (ADS '15), pp. 36-41, 2015.

[21] A. Sanzgiri and D. Dasgupta, "Classification of insider threat detection techniques," Proc. 11th Annual Cyber and Information Security Research Conference (CISRC '16), article no. 25, pp. 1-4, 2016.

[22] I. Kotenko and E. Doynikova, "Dynamical calculation of security metrics for countermeasure selection in computer networks," Proc. 2016 24th Euromicro International Conference on Parallel, Distributed, and Network-Based Processing, pp. 558-565, 2016.

[23] A. Ganin, P. Quach, M. Panwar, Z. A. Collier, J. M. Keisler, D. Marchese, and I. Linkov, "Multicriteria decision framework for cybersecurity risk assessment and management," Risk Analysis, pp. 1-17, 2017.

[24] G. Yee, "Assessing security protection for sensitive data," Proc. Eleventh International Conference on Emerging Security Information, Systems and Technologies (SECURWARE 2017), pp. 111-116, 2017.