This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication.
IEEE TRANSACTIONS ON DEPEDABLE AND SECURE COMPUTING

k-Zero Day Safety: A Network Security Metric for Measuring the Risk of Unknown Vulnerabilities

Lingyu Wang, *Member, IEEE,* Sushil Jajodia, *Fellow, IEEE,* Anoop Singhal, *Senior Member, IEEE,* Pengsu Cheng, and Steven Noel, *Member, IEEE,*

***Abstract*—By enabling a direct comparison of different security solutions with respect to their relative effectiveness, a network security metric may provide quantifiable evidences to assist security practitioners in securing computer networks. However, research on security metrics has been hindered by difficulties in handling zero day attacks exploiting unknown vulnerabilities. In fact, the security risk of unknown vulnerabilities has been considered as something unmeasurable due to the less predictable nature of software flaws. This causes a major difficulty to security metrics, because a more secure configuration would be of little value if it were equally susceptible to zero day attacks. In this paper, we propose a novel security metric, *k-zero day safety*, to address this issue. Instead of attempting to rank unknown vulnerabilities, our metric counts how many such vulnerabilities would be required for compromising network assets; a larger count implies more security since the likelihood of having more unknown vulnerabilities available, applicable, and exploitable all at the same time will be significantly lower. We formally define the metric, analyze the complexity of computing the metric, devise heuristic algorithms for intractable cases, and finally demonstrate through case studies that applying the metric to existing network security practices may generate actionable knowledge.**

***Index Terms*—Security metrics, network security, attack graph, network hardening**

I. Introduction

Computer networks have long become the nerve system of enterprise information systems and critical infrastructures on which our societies are increasingly dependent. However, the scale and severity of security threats to computer networks have continued to grow at an ever-increasing pace. Potential consequences of a security attack have also become more and more serious as many high-profile attacks are reportedly targeting not only computer applications but also industrial control systems at nuclear power plants, implanted heart defibrillators, and military satellites.

One of the main difficulties in securing computer networks is the lack of means for directly measuring the relative effectiveness of different security solutions in a given network, since "you cannot improve what you cannot measure". Indirect measurements, such as the false positive and negative rates of an intrusion detection system or firewall, may sometimes be obtained through laboratory testing, but they typically say very little about the actual effectiveness of the solution when it is deployed in a real world network which may be very different from the testing environment. In practice, choosing and deploying a security solution still heavily rely on human experts' experiences following a trial-and-error approach, which renders those tasks an art, instead of a science.

In such a context, a network security metric is desirable because it would enable a direct measurement and comparison of the amounts of security provided by different security solutions. Existing efforts on network security metrics typically assign numeric scores to vulnerabilities based on known facts about vulnerabilities. However, such a methodology is no longer applicable when we consider zero day attacks. In fact, a popular criticism of past efforts on security metrics is that they cannot deal with unknown vulnerabilities, which are generally believed to be unmeasurable [21]. Unfortunately, without considering unknown vulnerabilities, a security metric will only have questionable value at best, since it may determine a network configuration to be more secure while that configuration is in fact equally susceptible to zero day attacks. We thus fall into the agnosticism that security is not quantifiable until we can fix all potential security flaws but by then we certainly do not need security metric at all [21].

In this paper, we propose a novel network security metric, *k-zero day safety*, to address this issue. Roughly speaking, instead of attempting to measure *which* unknown vulnerabilities are more likely to exist, we start with the worst case assumption that this is not measurable. Our metric then simply counts *how many* zero day vulnerabilities are required to compromise a network asset. A larger count will indicate a relatively more secure network, since the likelihood of having more unknown vulnerabilities all available at the same time, applicable to the same network, and exploitable by the same attacker, will be lower. We will formally define the k-zero day safety metric based on an abstract model of networks and zero day attacks. We analyze the complexity of computing the metric and design heuristic algorithms for addressing this complexity in special cases. We demonstrate the usefulness of the metric by applying it to the evaluation of existing practices in network hardening through a series of case studies.

The contribution of this work is twofold. First, to the best of our knowledge, this is among the first efforts on network security metrics that is capable of modeling the security risk of unknown zero day attacks. Second, we believe the metric would bring about new opportunities to the quantitative evaluation, hardening, and design of secure networks.

The preliminary version of this paper has previously ap-

L. Wang and P. Cheng are with The Concordia Institute for Information Systems Engineering (CIISE), Concordia University, Montreal, QC H3G1M8 Canada e-mail: (wang@ciise.concordia.ca).

S. Jajodia and S. Noel are with The Center for Secure Information Systems, George Mason University, Fairfax, VA 22030, USA.

A. Singhal is with The Computer Security Division, National Institute of Standards and Technology, Gaithersburg, MD 20899, USA.

Digital Object Indentifier 10.1109/TDSC.2013.24

peared in [41]. In this paper, we have substantially improved and extended the previous version; the most significant extensions include a new heuristic algorithm for efficiently computing the metric in special cases (Section IV-C), the novel concept of sub-metrics for characterizing a network's security-related properties (Section V-B), discussions on steps taken to instantiate the metric model V-C, and finally a series of case studies for demonstrating how the proposed metric may be applied for various purposes (Section VI). In addition, we have designed a new, cleaner version of the metric model for faciliating a more focused discussion (Section III).

The remainder of this paper is organized as follows. The rest of this section first builds intuitions through a running example. We then review related work in Section II, present our model and define the metric in Section III, study complexity and design algorithms in Section IV, apply the metric to network hardening in Section V, describe a series of case studies in Section VI, and finally conclude the paper in Section VII. All the proofs can be found in appendices.

A. Motivating Example

Figure 1 shows a toy example in which host 1 and 2 comprise the internal network. The firewall allows all outbound connection requests but blocks inbound requests to host 2. Assume the main security concern here is whether any attacker on host 0 can obtain the root privilege on host 2. Clearly, if we assume all the services to be free of known vulnerabilities, then a vulnerability scanner or attack graph will both draw the same conclusion that this network is secure (attackers on host 0 cannot obtain the root privilege on host 2.

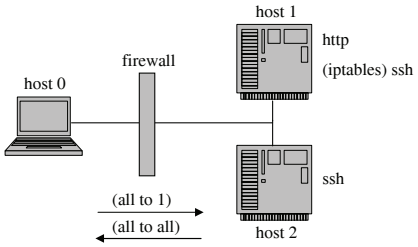

Fig. 1. An Example Network

Now consider the following two iptables policies.

- *Policy 1*: The iptables rules are left in a default configuration that accepts all requests.
- *Policy 2*: The iptables rules are configured to only allow specific IPs, excluding host 0, to access the ssh service.

Clearly, since the network is already secure, policy 1 will be preferable due to its simplicity (no special iptables rules need to be configured by the administrator) and functionality (any external host may connect to the ssh service on host 1).

However, a different conclusion can be drawn if we compare the above two policies with respect to *the network's resistance to potential zero-day vulnerabilities*. Specifically,

1) Under Policy 1, the upper diagram in Figure 2 (where each triple indicates an exploit ⟨vulnerability, source host, destination host⟩ and a pair indicates a condition ⟨condition, host⟩) illustrates three possible ways for compromising host 2:
   a) The attacker on host 0 exploits a zero-day vulnerability in the HTTP service on host 1 and then uses it as a stepping stone to exploit another zero-day vulnerability in the secure shell service on host 2.
   b) He/She exploits a zero-day vulnerability in the secure shell service on both host 1 and 2.
   c) He/She exploits a zero-day vulnerability in the firewall (e.g., a default password) to circumvent the traffic blocking before compromising host 2.

   The above first and third cases require two different zero-day vulnerabilities, whereas the second only requires one zero-day vulnerability (in the secure shell service). Therefore, the network can be compromised with at least one zero-day attack under Policy 1.
2) Under Policy 2, only the second case is different, as illustrated in the lower diagram in Figure 2.
   a) The same as the above 1(a).
   b) The attacker exploits a zero-day vulnerability to circumvent the iptables rules before exploiting the secure shell service on both host 1 and 2.
   c) The same as the above 1(c).

   All three cases now require two different zero-day vulnerabilities. The network can thus be compromised with at least two zero-day attacks under Policy 2.

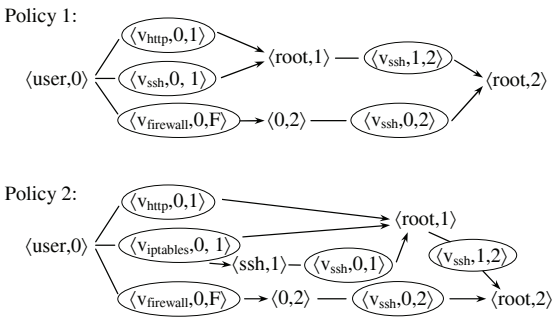

Fig. 2. Sequences of Zero Day Attacks

Considering the fact that each zero-day attack has only a limited lifetime (before the vulnerability is disclosed and fixed), it is reasonable to assume that the likelihood of having a larger number of distinct zero-day vulnerabilities all available at the same time in this particular network will be significantly smaller (the probability will decrease exponentially if the occurrences of different vulnerabilities can be regarded as independent events; however, our metric will not depend on any specific statistical model, considering the process of finding vulnerabilities is believed to be chaotic). To revisit the above example, the network can be regarded as more secure under Policy 2 than under Policy 1 since the former requires more (two) zero-day attacks to be compromised.

The key observation here is that, considering a network's resistance to potential zero-day vulnerabilities may assist in ranking the relative security of different network configurations, which may be otherwise indistinguishable under existing vulnerability analysis or attack graph-based techniques. The remainder of this paper will build upon this key observation and address remaining issues.

II. Related Work

Standardization Efforts There exist numerous standardization efforts on security metrics, such as the Common Vulnerability Scoring System (CVSS) [24] and, more recently, the Common Weakness Scoring System (CWSS) [37]. The former focuses on ranking known vulnerabilities, whereas the latter on software weaknesses. Both CVSS and CWSS measure the relative severity of individual vulnerabilities in isolation and do not address their overall impact. On the other hand, these efforts form a practical foundation for research on security metrics, as they provide security analysts and vendors standard ways for assigning numerical scores to known vulnerabilities which are already available in public vulnerability databases, such as the National Vulnerability Database (NVD) [25].

Network Security Metrics The research on network security metrics has attracted much attention, as surveyed in [16], [33], [39]. In an early work [4], a metric is proposed as the time and efforts required by potential adversaries on the basis of a Markov model of attack stages. In another early work, the length of shortest attack paths, in terms of the number of exploits, conditions, or both, is taken as a security metric for measuring the amount of security of networks [30]. A follow-up work observes that a security metric based on the length of shortest paths will not consider the existence of multiple paths and proposes employing the number of such paths as a metric [26]. In [2], an attack tree marked with abstract exploitability and hazard is parsed to find sequences of attacks that correspond to the easiest paths followed by potential attackers, and the amount of minimum effort needed along such paths is used as a metric. Another similar work regards the arithmetic mean of all attack paths' lengths as a security metric of average attackers' expected efforts in compromising given critical assets [19]. The main limitation of those early work lies in that they generally do not consider the relative severity or likelihood of vulnerabilities.

In a later work, the Network Compromise Percentage Metric (NCP) is proposed while evaluating the so-called *defense in depth* strategy using attack graphs [20], which basically indicates the percentage of network assets that may be compromised by attackers. In a recent work [23], the authors rank states in an attack graph based on probabilities of attackers reaching these states during a random simulation; the PageRank algorithm is adapted for such a ranking; a key assumption made in this work is that attackers would progress along different paths in an attack graph in a random fashion. A similar work replaces attack trees with more advanced attack graphs and replace attack paths with attack scenarios [29]. A Mean Time-to-Compromise metric is proposed based on the predator state-space model (SSM) used in the biological sciences in [18]; defined as the average time required for compromising networks, the metric provides richer semantics than other abstract metrics; the main limitation of this work lies in an oversimplified model of network intrusions and differences between vulnerabilities.

More recently, the authors in [13] observe that different security metrics will provide only a partial view of security, and the authors then propose a framework for grouping such metrics based on their relative importance. A recent work proposes a risk management framework using Bayesian networks to quantify the chances of attacks and to develop a security mitigation and management plan [31]. Another recent study of several CVSS-based vulnerability metrics shows the correlation between those metrics and the time to compromise of a system [11]. In our recent work, we have proposed a general framework for designing network security metrics [43], Bayesian network-based metrics [9], a probabilistic approach [40]. Parallel to our work on probabilistic security metrics, the authors in [12] address several important issues in calculating such metrics including the dependencies between different attack sequences in an attack graph and cyclic structures in such graphs.

Zero Day Attack Most existing work focus on developing security metrics for known vulnerabilities in a network. A few exceptions include an empirical study on the total number of zero day vulnerabilities available on a single day based on existing facts about vulnerabilities [22], a report on the popularity of zero day vulnerabilities among attackers [10], an empirical study on software vulnerabilities' life cycles [34], and more recently an effort on estimating the effort required for developing new exploits [36]. We note that if statistical information about zero day vulnerabilities, such as the total number of such vulnerabilities, can be obtained or estimated based on such empirical studies, then such information can certainly be incorporated into our metric, for example, by dynamically adjusting the value of k (that is, a larger k is needed when more zero day vulnerabilities are available). A recent effort ranks different applications in the same system by how serious the consequence would be if there exists a single zero day vulnerability in those applications [14]. In contrast to our work, it has a different focus (ranking different applications inside the same system instead of ranking different network configurations) and different metric (seriousness of consequences instead of number of vulnerabilities).

Security Metrics in Other Areas Security metrics have also been proposed in areas other than network security; such studies have proved to be valuable to our research. In software security, the attack surface metric measures how likely a software is vulnerable to attacks based on the degree of exposure [28]. Our work borrows from attack surface the idea of focusing on interfaces (e.g., remote services) instead of internal details (e.g., local services and applications) modeling which may be practically infeasible, but we apply the idea to a network of computer systems rather than a single software system. In the context of software and application security, there has also been some pessimism about quantifying software security [3], [21], [39]. Our focuses on ranking, instead of quantifying, security threats at the network and system level essentially enables us to work with weaker assumptions that actually stem from such unmeasurability results [3], [21]. Finally, the study of privacy metrics has recently seen significant successes [7], [32], which clearly indicates the huge impact of developing suitable security metrics on related research. In this paper, the proposed zero day attack graph model borrows the compact model given in [1] while incorporating zero day vulnerabilities.

III. Modeling k-Zero Day safety

This section introduces the k-zero day safety metric model. In this paper, we have redesigned a light-weight version of the original model previously presented in [41]. This cleaner model will allow a more focused discussion on essential aspects of the k-zero day safety. Additional features will be introduced in later sections when they are needed.

First, we revisit our motivating example to illustrate the information necessary for establishing the network model.

Example 3.1: The discussion in Section I-A has involved following information about the network.

- A collection of hosts $\{0, 1, 2, F\}$ (F for the firewall),
- The connectivity relation $\{\langle 0, F\rangle, \langle 0, 1\rangle, \langle 0, 2\rangle, \langle 1, F\rangle, \langle 1, 0\rangle, \langle 1, 2\rangle, \langle 2, F\rangle, \langle 2, 0\rangle, \langle 2, 1\rangle\}$,
- Services $\{http, ssh, iptables\}$ on host 1, $\{ssh\}$ on host 2, and $\{firewall\}$ on host F.
- Privileges $\{user, root\}$.

The main design rationale here is to hide internal details of hosts while focusing on the interfaces (services and connectivity) and essential security properties (privileges). A few subtleties are as follows. First, hosts are meant to include not only computers but all networking devices potentially vulnerable to zero-day attacks (e.g., firewalls). Second, a currently disabled connectivity (e.g., $\langle 0, 2\rangle$ in the above example) still needs to be considered since it may potentially be re-enabled through zero-day attacks (e.g., on firewalls). Third, only *remote services* (those remotely accessible over the network), and *security services* (those used for regulating accesses to remote services) are considered. Modeling local services or applications is not always feasible (e.g., attackers may install their own applications after obtaining initial accesses to a host). Instead, we will model the effect of compromising such applications through privilege escalation. For this purpose, privileges under which services are running, and those that can be potentially obtained through a privilege escalation, will both be considered.

The following formalizes our network model.

Definition 1 (Network): The network model includes:

- the sets of hosts H, services S, and privileges P.
- the mappings from hosts to sets of services $serv(.) : H \rightarrow 2^S$ and privileges $priv(.) : H \rightarrow 2^P$.
- the relation of connectivity $conn \subseteq H \times H$.

Next, we model zero day exploits. The very notion of *unknown* vulnerability means that we cannot assume any vulnerability-specific property, such as exploitability or impact. Instead, our model is based on generic properties of existing vulnerabilities. Specifically, we define two types of zero-day vulnerabilities. First, a zero-day vulnerability in services are those whose details are unknown except that their exploitation requires a network connection between the source and destination hosts, a remotely accessible service on the destination host, and existing privilege on the source host. In addition, exploiting such a vulnerability can potentially yield any privilege on the destination host. Those assumptions are formalized as the first type of zero-day exploits in Definition 2. The second type of zero-day exploits in the definition represent privilege escalation following the exploitation of services.

Definition 2 (Zero-Day Exploit): Given a network,

- for each remote service s, we define a zero-day vulnerability v_s such that the zero-day exploit $\langle v_s, h, h'\rangle$ has three pre-conditions, $\langle s, h'\rangle$ (existence of service), $\langle h, h'\rangle$ (connectivity), and $\langle p, h\rangle$ (attacker's existing privilege); it has one post-condition $\langle p_s, h'\rangle$ where p_s is the privilege of service s on h'.
- for each privilege p, we define a zero day vulnerability v_p such that the pre-conditions of the zero-day exploit $\langle v_p, h, h\rangle$ include the privileges of remote services on h, and the post-condition is $\langle p, h\rangle$.

Now that we have defined zero-day exploits, it is straightforward to extend a traditional attack graph with zero-day exploits. Specifically, a *zero-day attack graph* is simply a directed graph composed of both zero-day and known exploits, with edges pointing from pre-conditions to corresponding exploits and from exploits to their post-conditions.

Example 3.2: Figure 3 shows the zero day attack graph of our (in this special case, all exploits are zero day).

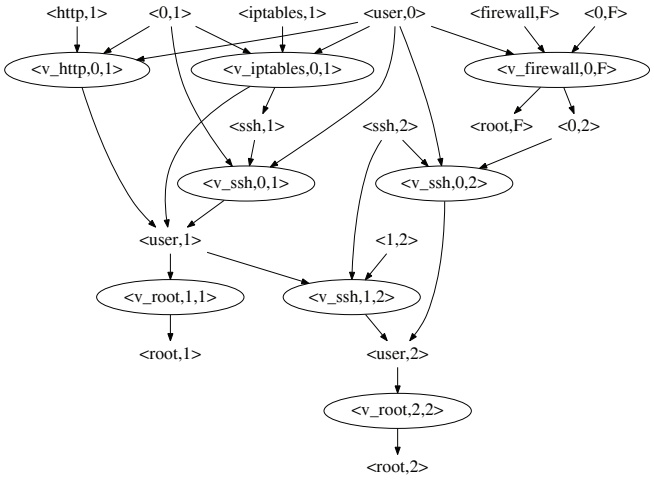

Fig. 3. An Example of Zero Day Attack Graph

In a zero-day attack graph, we use the notion of *initial condition* for conditions that are not post-conditions of any exploit (e.g., initially satisfied conditions, or those as the result of insider attacks or user mistakes). We also need the notion of *attack sequence*, that is, any sequence of exploits in which the pre-conditions of every exploit are either initial conditions, or post-conditions of some preceding exploits (intuitively, this indicates an executable sequence of attacks). Finally, we regard a given condition a as the *asset* (which can be extended to multiple assets with different values [41]) and use the notation $seq(a)$ for any attack sequence that leads to a.

Example 3.3: In our running example, following attack sequences all lead to the asset $\langle root, 2\rangle$.

1) $\langle v_{http}, 0, 1\rangle, \langle v_{ssh}, 1, 2\rangle, \langle v_{root}, 2, 2\rangle$
2) $\langle v_{iptables}, 0, 1\rangle, \langle v_{ssh}, 1, 2\rangle, \langle v_{root}, 2, 2\rangle$
3) $\langle v_{iptables}, 0, 1\rangle, \langle v_{ssh}, 0, 1\rangle, \langle v_{ssh}, 1, 2\rangle, \langle v_{root}, 2, 2\rangle$
4) $\langle v_{firewall}, 0, F\rangle, \langle v_{ssh}, 0, 2\rangle, \langle v_{root}, 2, 2\rangle$

We are now ready to define the k-zero day safety metric. In Definition 3, we do so in three steps.

First, we model two different cases in which two zero day exploits should be counted only once, that is, either when they involve the same zero day vulnerability or when they

correspond to a trivial privilege escalation due to the lack of isolation techniques. Although the equivalence relation in those two cases has very different semantics, the effect on our metric will be the same. The metric function $k0d(.)$ counts how many exploits in their symmetric difference are distinct (not related through $\equiv_v$). Defining this function over the symmetric difference of two sets allows it to satisfy the required algebraic properties, as formally stated in Theorem 1. The k-zero day safety metric is defined by applying the metric function $k0d(.)$ to the minimal attack sequences leading to an asset. We note that $k0d(a)$ is always unique even though multiple attack sequences may lead to the same asset. The empty set in the definition can be interpreted as the conjunction of all initial conditions (which are initially satisfied).

Definition 3 (k-Zero Day Safety): Given the set of zero-day exploits E_0, we define

- a relation $\equiv_v \subseteq E_0 \times E_0$ such that $e \equiv_v e'$ indicates either e and e' involve the same zero day vulnerability, or $e = \langle v_s, h_1, h_2\rangle$ and $e' = \langle v_p, h_2, h_2\rangle$ are true, and exploiting s yields p. e and e' are said distinct if $e \not\equiv_v e'$.
- a function $k0d(.) : 2^{E_0} \times 2^{E_0} \rightarrow [0, \infty]$ as $k0d(F, F') = max(\{\ |F''| : F'' \subseteq (F \triangle F'), (\forall e_1, e_2 \in F'')\ (e_1 \not\equiv_v e_2)\})$ where $|F''|$ denotes the cardinality, $max(.)$ the maximum value, and $F \triangle F'$ the symmetric difference $(F \setminus F') \cup (F' \setminus F)$.
- for an asset a, we use $k = k0d(a)$ for $min(\{k0d(q \cap E_0, \phi) : q \in seq(a)\})$ where $min(.)$ denotes the minimum value. For any $k' \in [0, k)$, we say a is k'-zero day safe (we may also say a is k-zero day safe when the meaning is clear from the context).

Theorem 1: $k0d(.)$ is a metric function.

Proof: This is to prove, for all $F, F', F'' \subseteq E_0$, we have [6]

1) $k0d(F, F') = 0$ iff $F = F'$: This is straightforward since $k0d(F, F') = 0$ iff $F \triangle F' = \phi$, and the latter is equivalent to $F = F'$.
2) $k0d(F, F') = k0d(F', F)$: This property is satisfied by the definition of symmetric difference.
3) $k0d(F, F') + k0d(F', F'') \geq k0d(F, F'')$: Denote by $tmp(G)$ the function $max(\{\ |G'| : G' \subseteq G, \forall e_1, e_2 \in G'\ (e_1 \not\equiv_v e_2)\})$. First, the symmetric difference satisfies the triangle inclusion relation $F \triangle F'' \subset (F \triangle F') \cup (F' \triangle F'')$ [6]. So, $tmp((F \triangle F') \cup (F' \triangle F'')) \geq tmp(F \triangle F'')$ holds. Next, we only need to show $tmp(F \triangle F') + tmp(F' \triangle F'') \geq tmp((F \triangle F') \cup (F' \triangle F''))$ is true. It suffices to show the function $tmp(.)$ to be subadditive, that is, $tmp(G) + tmp(G') \geq tmp(G \cup G')$ holds for any $G, G' \subseteq E_0$. This follows from the fact that if the relation $e \equiv_v e'$ holds for any $e, e' \in G$ (or $e, e' \in G'$), it also holds in $G \cup G'$ (the converse is not necessarily true).

□

Example 3.4: For the running example, suppose all exploits of services involve distinct vulnerabilities except $\langle v_{ssh}, 0, 1\rangle$, $\langle v_{ssh}, 1, 2\rangle$, and $\langle v_{ssh}, 0, 2\rangle$. Assume ssh and $http$ are not protected by isolation but $iptables$ is protected. Then, the relation $\equiv_v$ is shown in Table I where 1 indicates two exploits are related and 0 the opposite. Clearly, if we assume $A = \{\langle root, 2\rangle\}$ then we have $k0d(A) = 2$, and the network is 0 or 1-zero day safe (we may also say it is 2-zero day safe when the meaning is clear from the context).

	$\langle v_{iptables}, 0, 1\rangle$	$\langle v_{http}, 0, 1\rangle$	$\langle v_{ssh}, 0, 1\rangle$	$\langle v_{root}, 1, 1\rangle$	$\langle v_{ssh}, 1, 2\rangle$	$\langle v_{firewall}, 0, F\rangle$	$\langle v_{ssh}, 0, 2\rangle$	$\langle v_{root}, 2, 2\rangle$
$\langle v_{iptables}, 0, 1\rangle$	1	0	0	0	0	0	0	0
$\langle v_{http}, 0, 1\rangle$	0	1	0	1	0	0	0	0
$\langle v_{ssh}, 0, 1\rangle$	0	0	1	1	1	0	1	0
$\langle v_{root}, 1, 1\rangle$	0	1	1	1	0	0	0	0
$\langle v_{ssh}, 1, 2\rangle$	0	0	1	0	1	0	1	1
$\langle v_{firewall}, 0, F\rangle$	0	0	0	0	0	1	0	0
$\langle v_{ssh}, 0, 2\rangle$	0	0	1	0	1	0	1	1
$\langle v_{root}, 2, 2\rangle$	0	0	0	0	1	0	1	1

TABLE I
AN EXAMPLE OF RELATION $\equiv_v$

IV. COMPUTING k-ZERO DAY SAFETY

This section presents algorithms for computing the proposed metric. The first two algorithms have appeared in [41] and the third algorithm is a new contribution of this paper.

A. Computing the Value of k

To compute the k-zero day safety of a network, Procedure $k0d\_Bwd$ shown in Figure 4 first derives a logic proposition of each asset in terms of exploits. In the disjunctive normal form (DNF) of the derived proposition, each conjunctive clause will correspond to a minimal set of exploits that can jointly compromise the asset. Therefore, the metric value of that asset can be determined by applying the metric function $k0d(.)$ to such conjunctive clauses and taking the minimum value among the results. Note that a negated condition in an asset will be replaced with the negation of exploits, whereas the latter will not be further processed (as indicated in line 6).

Procedure $k0d\_Bwd$
Input: Zero day attack graph G, a set of assets A with the valuation function $v(.)$
Output: An integer k
Method:
1. **For** each asset $a \in A$
2. **Let** L be the logic proposition representing a
3. **While** at least one of the following is possible, do
4. **Replace** each initial condition c with $TRUE$
5. **Replace** each condition c with $\bigvee_{e \in \{e' : c \in post(e')\}} e$
6. **Replace** each non-negated exploit e with $e \wedge (\bigwedge_{c \in pre(e)} c)$
7. **Let** $L_1 \vee L_2 \vee \ldots L_n$ be the DNF of L
8. **Let** $k_a = min(\{k0d(F_i \cap E_0, \phi) : F_i$ is set of non-negated exploits in $L_i, 1 \leq i \leq n\})$
9. **Return** $\lceil \sum_{a \in A}(k_a \cdot v(a)) / \sum_{a \in A} v(a) \rceil - 1$

Fig. 4. Computing the Value of k

Complexity The procedure's worst-case complexity is exponential in the size of the zero day attack graph. Indeed, Theorem 2 shows that the problem of computing k-zero day safety is NP-hard.

Theorem 2: Given a zero day attack graph and an asset a, finding an attack sequence $q \in seq(a)$ to minimize $k0d(q \cap E_0, \phi)$ is NP-complete.

Proof (Sketch): First, the problem is NP because, given a sequence of exploits q, it is easy to see that $q \in seq(a)$ and

$k0d(q \cap E_0, \phi) = k$ can both be verified in polynomial time (in the size of the zero day attack graph).

Next, we reduce the known NP-hard problem of finding the minimum attack (that is, an attack sequence with the minimum number of exploits) in attack graph [1], [35] to the current problem. First of all, the reduction cannot be trivially achieved by simply replacing each known exploit with a zero day exploit in a given attack graph of know exploits, because, unlike the former, the latter has a fixed number of hard-coded pre- and post-conditions that may prevent them from fitting in the position of a known exploit.

We construct a zero day attack graph G' by injecting a zero day exploit before each known exploit. First, let $G' = G$. Then, for each known exploit e of a service s from host h_1 to h_2, we inject a zero day exploit e' with the post-conditions $\{\langle s, h_2\rangle, p_{useless}\}$ where $p_{useless}$ is a privilege designed not to be the pre-condition of any exploit. We then have the following two facts. First, executing e requires e' to be executed first; conversely, if e' needs to be executed, then the only reason must be to satisfy the condition $\langle s, h_2\rangle$ and consequently to execute e. Second, among the three conditions in $pre(e') = \{\langle s', h_2\rangle, \langle h_1, h_2\rangle, \langle p_{least}, h_1\rangle\}$, the first is an initial condition and the last two are members of $pre(e)$. Therefore, G and G' are isomorphic if we regard e and e' as a single exploit and ignore the initial condition $\langle s', h_2\rangle$. Next, for each known exploit e involving only one host h, we replace e with a zero day exploit e' and a known exploit e'' satisfying that $post(e'') = post(e)$, $pre(e'') = pre(e) \setminus \{\langle p, h\rangle\} \cup \{\langle p', h\rangle\}$ where $\langle p, h\rangle \in pre(e)$ and $\{\langle p', h\rangle\}$ are two privileges. We also let $post(e') = \{\langle p', h\rangle\}$, and design relation $\equiv_v$ in such a way that e' is not related to any other zero day exploits in h. We then have two similar facts as above.

Based on the above construction, given any asset a, for any attack sequence $q' \in seq(a)$ in G', the known exploits in q also form an attack sequence $q \in seq(a)$ in G (note that a will always be the post-condition of known exploits due to our construction). Moreover, if we design $\equiv_v$ in such a way that no two zero day exploits are related, then we have $\mid q \mid = k0d(q' \cap E_0, \phi)$. Therefore, for any non-negative integer k, finding q' in G' to minimize $k0d(q' \cap E_0, \phi)$ will immediately yield q in G that also minimizes $\mid q \mid$, and the latter is essentially the minimum attack problem. This shows the former to be an NP-hard problem and concludes the proof. □

Note that the intractability result here only implies that a single algorithm is not likely to be found to efficiently determine k for all possible inputs (that is, arbitrary zero day attack graphs). However, efficient solutions still exist for practical purposes. We next examine two such cases.

B. Determining k-Zero Day Safety for a Given Threshold

For many practical purposes, it may suffice to know that every asset in a network is k-zero day safe for a given threshold k, even though the network may in reality be k'-zero day safe for some unknown $k' > k$ (determining k' is intractable). Figure 5 shows a recursive Procedure $k0d\_Fwd$ whose complexity is polynomial in the size of a zero day attack graph if k is a constant compared to that size. Roughly speaking, the procedure attempts to compromise each asset with less than k distinct zero day exploits through a forward search of limited depth. The asset is not k-zero day safe if any branch of the search succeeds, and vice versa.

Procedure $k0d\_Fwd$
Input: A zero day attack graph G, an asset a, $k > 0$, $T_e = \phi$, $T_c = C_I$
//T_e and T_c denotes exploits and conditions visited so far, respectively
Output: $TRUE$, if $k0d(a) > k$; $FALSE$, otherwise
Method:
1. **If** $k0d\_reachable(T_e, T_c) \wedge k0d(T_e) < k$
2. **Return** $FALSE$
3. **ElseIf** $k0d(T_e) \geq k$
4. **Return** $TRUE$
5. **Else**
6. **For** each $e \in E_0 \setminus T_e$ satisfying $pre(e) \subseteq T_c$
7. **If** $\neg\ k0d\_Fwd(G, a, k, T_e \cup \{e\}, T_c \cup post(e))$
8. **Return** $FALSE$
9. **Return** $TRUE$

Sub-Procedure $k0d\_Reachable$
Input: T_e, T_c
Output: $TRUE$ or $FALSE$
Method:
10. **While** $(\exists e \in E_1 \setminus T_e)(pre(e) \subseteq T_c)$
11. **Let** $T_e = T_e \cup \{e\}$
12. **Let** $T_c = T_c \cup post(e)$
13. **Return** $(\bigwedge_{c \in T_c} c \rightarrow a)$

Fig. 5. Determining k-Zero Day Safety for a Given k

Complexity The complexity of this procedure is polynomial in the size of the zero day attack graph if k is a constant (details can be found in [41].

C. Computing k-Zero Day Safety as Shortest Paths in a DAG

Although determining the value of k is NP-hard in general, efficient solutions may exist for special cases of practical relevance. We study one such case where k can be determined in polynomial time when the following two assumptions hold on the given zero-day attack graph.

- First, the conjunctive relationships between conditions are mostly limited to be within each host or each small group of hosts. This is true if remote hosts are mostly used as stepping stones so each remote exploit will require only a privilege on the remote host. We can then regard each such condition as the vertex of an acyclic directed graph (DAG) (cycles will be avoided in deriving logic propositions [42]). Determining the value of k amounts to finding the *shortest* path along which the collection of zero-day exploits yields the minimum metric value.
- Second, the similarity between zero-day exploits modeled by the relation $\equiv_v$ will also mostly be limited to be with a host or small group of hosts. For those zero-day exploits that may be later related to other exploits by $\equiv_v$, we keep them in a set as the first part of a distance. For all other exploits, we only keep the result of applying the $k0d(.)$ metric as the second part of the distance. We can then propagate such distances along each edge.

In Figure 6, the Sub_Procedure $k0d\_Graph$ builds a DAG based on a given zero day attack graph and asset. The main procedure then imitates a standard algorithm for finding the shortest path in a DAG [5], with a few modifications. First, instead of a single number, each distance is now a set of pairs $\langle x, y\rangle$ where x denotes the result of applying $k0d(.)$ to exploits that later will not be related to others by $\equiv_v$, whereas y denotes the converse. Second, the reachable edges are collected in

order to determine whether an exploit may later be related to others by $\equiv_v$ (line 8).

Procedure $k0d\_Shortest$
Input: A zero day attack graph G, an asset L
Output: A non-negative real number k
Method:
1. **Let** G_s be a DAG with vertex L, and A be an empty array
2. **Let** $\langle G_s, A\rangle = k0d\_Graph(G, L, G_s, A)$
3. **Let** $vlist$ be any topological sort of G_s
4. **Let** $dist_L = \{\langle 0, \phi\rangle\}$ and $dist_x = \{\langle \infty, \phi\rangle\}$ for any other vertex x
5. **While** $vlist$ is not empty, do
6. **Delete** the first vertex u from $vlist$
7. **For** each outgoing edge $\langle u, v\rangle$ of u
8. **Let** $elist$ be the set of all edges reachable from v
9. **For** each $\langle x, y\rangle \in dist_u$
10. **Let** $y' = \{e : e \in y \cup A[\langle u, v\rangle], \exists e' \in elist\ \exists e'' \in A[e']\ e \equiv_v e''\}$
11. **Let** $x' = x + k0d((y \cup A[\langle u, v\rangle] \setminus y') \cap E_0, \phi)$
12. **Let** $dist_v = dist_v \cup \langle x', y'\rangle$
13. **While** $(\exists \langle x, y\rangle, \langle x', y'\rangle \in dist_v)(x \geq (x' + k0d(y' \cap E_0, \phi)))$
14. **Delete** $\langle x, y\rangle$ from $dist_v$
15. **Return** $min(\{x : \langle x, \phi\rangle \in dist_d, d$ is a dummy vertex $\})$

Sub_Procedure $k0d\_Graph$
Input: A zero day attack graph G, an asset L, a DAG G_s, an array A
Output: Updated G_s and $elable$
Method:
16. **Do**
17. (Lines 4-6 of Procedure $k0d\_Backward$)
18. **Let** L be its DNF
19. **While** there exists a conjunctive clause l in L including multiple conditions
20. **for** each conjunctive clause l in L
21. **If** l includes a condition c
22. **Add** vertex c and edge $\langle L, c\rangle$ to G_s
23. **Let** $A[\langle L, c\rangle]$ be the set of exploits in l
24. **Let** $\langle G_s, A\rangle = k0d\_Graph(G, c, G_s, A)$
25. **Else**
26. **Add** a dummy vertex d and edge $\langle L, d\rangle$ to G_s
27. **Let** $A[\langle L, d\rangle]$ be the set of exploits in l
28. **Return** G_s

Fig. 6. Computing k-Zero Day Safety as Shortest Paths in a DAG

Complexity The complexity of Sub-Procedure $k0d\_Graph$ is exponential in the number of exploits and conditions involved in the loop at lines 16-19. Therefore, if the first assumption perfectly holds, this loop will always terminate after processing a single host. If we regard the number of exploits and conditions on each host as a constant, then the complexity of the sub-procedure will be linear in the number of hosts. Second, the complexity of the main procedure depends on the size of the distance of each vertex. If the second assumption holds perfectly such that each distance has a negligible size, then the complexity of the main procedure will be dominated by processing the reachable edges in $elist$ and their labels A (line 10). Since each edge in G_s is visited exactly once by the main loop and the size of $elist$ is linear in the number of such edges, the processing of $elist$ takes quadratic time in the number of edges in G_s, which is roughly $O(|H|^4)$. Finally, multiplying this by the size of A, we have $|H|^4 \cdot |E_0|$.

V. Applying k-Zero Day Safety

In this section, we first demonstrate the power of our metric through applying it to network hardening. We also extend the basic metric model to define sub-metrics for measuring the potential of hardening options. Finally, we discuss practical issues in instantiating the model from given networks.

A. Redefining Network Hardening

Network hardening is to improve the security of existing networks through deploying security solutions or making configuration changes. In most existing work, network hardening is defined as a reachability problem in attack graphs, that is, finding a set of security conditions, disabling which will render goal conditions (assets) not reachable from initial conditions [17], [35], [42]. Since the reachability is a binary property, such a definition is qualitative in nature. Each network hardening solution is either valid or invalid, and all valid solutions will be deemed as equally good in terms of security (although those solutions may be ranked from other aspects, such as their costs [42]).

Based on the proposed k-zero day safety metric, we can now redefine network hardening as *rendering a network k-zero day safe for a larger k*. Clearly, such a concept generalizes the above qualitative approaches. Specifically, under our model, those qualitative approaches essentially achieve $k > 0$, meaning that attacks are no longer possible with known vulnerabilities only. In contrast to those qualitative approaches, our definition can rank network hardening solutions based on the relative degree of security guarantee provided by those solutions. Such a ranking would enable us to model network hardening as various forms of optimization problems, either with k as the objective function and cost as constraints (that is, to maximize security) or vice versa.

Moreover, the metric also provides insights to specific hardening options, since any means for increasing k would now become a potential hardening option. For clarify purposes, we unfold k based on our model in Equations (1) through (4). Based on those equations, we can see that k may be increased in many ways, including:

- *Increasing Diversity* Increasing the diversity of services will enable stronger assumptions about distinct zero day exploits (less exploits related by $\equiv_v$) in Equation (3), and consequently likely (but not necessarily, which is exactly why a metric is needed) increase k.
- *Strengthening Isolation* Strengthening isolation around services will provide a similar effect as the above option.
- *Disabling Services* Disabling or uninstalling unnecessary services will disable corresponding initial conditions and therefore yield longer attack sequences in Equation (4) and consequently a larger k.
- *Firewalls* Blocking unnecessary connectivity will provide a similar effect as the above option since connectivity is a special type of initial conditions.
- *Stricter Access Control* Enforcing stricter policies may improve user security and lessen the risk of insider attacks or unintentional user mistakes and thus disable existing initial conditions in Equation (4) and lead to a larger k.
- *Asset Backup* Asset backup will lead to more conjunctive clauses of conditions in the definitions of assets, and consequently longer attack sequences and a larger k.
- *Detection and Prevention* Protecting services and assets with intrusion detection and prevention efforts will lead to negation of conditions in the definition of assets and consequently a similar effect as the above option.
- *Security Services* Introducing more security services to restrict accesses to remote services may also disable initial conditions and consequently lead to longer attack sequences and a larger k.
- *Patching Known Vulnerabilities* Since known vulnera-

$$k = k0d(A) = \sum_{a \in A} (k0d(a) \cdot v(a)) / \sum_{a \in A} v(a) \quad (1)$$

$$k0d(a) = min(\{k0d(q \cap E_0, \phi) : q \in seq(a)\}) \quad (2)$$

$$k0d(q \cap E_0, \phi) = max(\{\ |F|\ : F \subseteq q \cap E_0,\ (\forall e_1, e_2 \in F)\ (e_1 \not\equiv_v e_2)\}) \quad (3)$$

$$seq(a) = \{e_1, e_2, \ldots, e_j : a \text{ is implied by } \cup_j\ post(e_j), (\forall i \in [1, j])\ (\forall c \in pre(e_i))\ (c \in C_I) \vee (\exists x \in [1, i-1]\ c \in post(e_x))\} \quad (4)$$

bilities may serve as shortcuts for bypassing zero day exploits, patching them will likely yield longer attack sequences and a larger k.

- *Prioritizing Hardening Options* The hardening options maybe prioritized based on the asset values in Equation (1) and shortest attack sequences in Equation (2) such that an option is given higher priority if it can lead to more significant reduction in k.

The above hardening options closely match current practices, such as the so-called *layered defense*, *defense in depth*, *security through virtualization*, and *security through diversity* approaches, and so on. This confirms the practical relevance of the proposed metric. Note that none of those hardening options can always guarantee improved security (that is, a hardening option does not always increase the value of k, as will be illustrated in Section VI). With the proposed metric, the relative effectiveness of potential network hardening options can now be directly compared in a simple, intuitive manner. Their cost can also be more easily justified, not based upon speculation or good will, but simply with a larger k.

B. Sub-Metrics

In addition to suggesting and evaluating hardening options, we now show that the proposed metric model can also be applied to modeling and quantifying a network's security-related properties, such as the degree of diversity among services and the level of patching with respect to known vulnerabilities. Such properties may serve as a starting point for security analysts in understanding the current state of security in a network. They can also indicate the potential of each network hardening option, and provide guidelines for choosing or prioritizing different options in designing a hardening solution (in contrast to evaluating a given one, as described in the previous section).

1) Effective Diversity: First, to quantify the diversity of services in a network (the level of isolation around services can be similarly quantified and hence omitted here), we define the sub-metric *effective diversity* of a network as the ratio between the network's current security, represented by k, and the range of security that can be achieved while varying the degree of diversity in services, or more precisely, the number of zero day vulnerabilities related by the relation $\equiv_v$ between $\mid E_0 \mid$ (that is, all services correspond to the same zero day vulnerabilities) and zero (that is, all services correspond to different vulnerabilities). Definition 4 formalizes this notion where k_{min} and k_{max} correspond to the two extreme cases where all, and none, zero day vulnerabilities required for compromising an asset are distinct, respectively.

Definition 4 (Effective Diversity): The level of effective diversity of a network is defined as $\frac{k-k_{min}}{k_{max}-k_{min}}$, where k is given in Equation (1), k_{min} and k_{max} are calculated using Equations (1) through (4) but with $\equiv_v$ replaced by $E_0 \times E_0$ and ϕ, respectively.

The effective diversity of a network indicates in a percentage how much diversity in services is present in terms of its effect on security. It also indicates how much more security can potentially be achieved by further diversifying the services, that is, the potential of this hardening option. Note that our definition focuses on diversity that is effective in improving security, rather than diversity in general.

Example 5.1: For our running example, we can see that, although the three ssh services may be further diversified, this hardening effort will not increase k (since the conjunctive clauses in Table III all include a single ssh service). Correspondingly, the effective diversity sub-metric will be equal to 1, meaning increasing diversity will not achieve any more security. On the other hand, if we increase the degree of isolation around the ssh service on host 2, such that compromising the service will not directly lead to the root privilege on host 2, then we can increase k by 1; correspondingly, the sub-metric for isolation (which can be defined similarly as in Definition 4) will be less than 1, indicating the potential of this hardening option.

2) Effective Patching: We define the next sub-metric, the level of effective patching, as the ratio between the network's current security, represented by k, and the range of k by varying the amount of known vulnerabilities from zero (that is, all are patched) and maximum possible (that is, known vulnerabilities are sufficient for compromising any asset, so $k = 0$). This sub-metric indicates the current state of known vulnerabilities in terms of their overall effect on security, and the potential in improving the overall security by patching those known vulnerabilities. Note that a high effective patching rate does not necessarily mean a small number of known vulnerabilities and conversely patching only a few vulnerabilities may significantly increase the k.

Definition 5 (Effective Patching): The level of effective patching of a network is defined as k/k_{max}, where k_{max} is calculated using Equations (1-4) but with $E_1 = \phi$.

3) Conditional Safety: For some of the hardening options, the sub-metric may need to be defined differently. For example, when we consider disabling initial conditions (which may imply changing network configuration, removing services, enforcing stricter access control policies, and so on), it may be more reasonable to define the sub-metric over each condition separately. This is partly due to the fact that, at least in theory, disabling conditions can always achieve $k = \infty$, for example, by isolating all the assets from networks. However, such a result is of little practical value. In contrast, the conditional safety defined below indicates the potential effect of disabling each condition, which will provide actionable information.

Definition 6 (Conditional Safety): The conditional safety of $c \in C$ with respect to $S_c \subseteq C$ (or simply that of c if

$S_c = \phi$) is defined as $k_{c|S_c} - k_{S_c}$, where k_{S_c} and $k_{c|S_c}$ is calculated using Equations (1-4) but with all conditions in S_c, and those together with c, respectively, set as $FALSE$.

Example 5.2: In Table IV, we have added corresponding initial conditions necessary to achieve the result shown in Table III. Recall that the first two conjunctions indicate attack sequences leading to the metric value of three, while the last leading to the metric value of two. We can see that disabling $\langle ssh, 2\rangle$ (that is, host 2 now has no remote service at all) can render all three sequences invalid. Therefore, the conditional safety of $\langle ssh, 2\rangle$ is ∞. On the other hand, disabling $\langle 0, F\rangle$ or $\langle firewall, F\rangle$ will only prevent the last sequence, increasing k by one; those two conditions thus have a conditional safety of one. Similarly, we can see that, after $\langle 0, F\rangle$ is already disabled, further disabling $\langle 1, 2\rangle$ or $\langle 0, 1\rangle$ will increase k to ∞, but disabling $\langle http, 1\rangle$ or $\langle iptables, 1\rangle$ will not change k at all.

$$\begin{aligned}\langle root, 2\rangle \equiv\ & (\langle v_{root}, 2, 2\rangle \wedge \langle v_{ssh}, 1, 2\rangle \wedge \langle v_{http}, 0, 1\rangle \wedge \langle 1, 2\rangle \wedge \langle ssh, 2\rangle \\ & \wedge \langle http, 1\rangle \wedge \langle 0, 1\rangle) \\ \vee\ & (\langle v_{root}, 2, 2\rangle \wedge \langle v_{ssh}, 1, 2\rangle \wedge \langle v_{iptables}, 0, 1\rangle \wedge \langle 1, 2\rangle \wedge \\ & \langle ssh, 2\rangle \wedge \langle iptables, 1\rangle \wedge \langle 0, 1\rangle) \\ \vee\ & (\langle v_{root}, 2, 2\rangle \wedge \langle v_{ssh}, 0, 2\rangle \wedge \langle v_{firewall}, 0, F\rangle \wedge \langle ssh, 2\rangle \\ & \wedge \langle 0, F\rangle \wedge \langle firewall, F\rangle)\end{aligned}$$

TABLE II
CONDITIONAL SAFETY

C. Instantiating the Model

The proposed metric and algorithms are based on an abstract model of networks and assumptions about zero-day attacks. How to instantiate such a model for a given network is an equally important issue. This section describes relevant input information that needs to be collected and possible ways for collecting it, and discusses the practicality and scalability.

1) The Network Model: To instantiate the network model (Section III), we need to collect information about

- hosts (e.g., computers, routers, switches, firewalls, etc.),
- connectivity between hosts, and
- for each host, its remotely accessible services, security mechanisms and services, and privileges.

Such information is typically already available to administrators in the form of a network map or configuration database. A network scanning will assist in collecting or verifying information about hosts, connectivity, and services. Nonetheless, a close examination of host configurations (including firewall rules) is still necessary since network maps and network scanning will usually not reveal hidden or disabled services or connectivity (which may be re-enabled through zero day attacks and thus must be correctly modeled), and privileges are often best identified by examining the host configuration.

Collecting and maintaining such information for a large network certainly involves substantial time and efforts. However, we note that a key advantage of our model is its exclusion of local applications and services (modeling which would be infeasible for most networks). Focusing on remote services allows our model to stay manageable and scalable, considering the fact that most hosts typically only have a few open ports (but many more local applications).

2) Zero Day Attack Model: To instantiate the zero day attack graph model, we need to collect both

- zero day exploits, and
- exploits of known vulnerabilities.

The former can be directly composed based on the network model, with no additional information needed, since, unlike known vulnerabilities, all zero day exploits have hard-coded conditions (Section III). On the other hand, exploits of known vulnerabilities must be identified, together with their pre- and post-conditions (which are specific to each exploit). Known vulnerabilities may be discovered through various vulnerability scanners, and their pre- and post-conditions may be obtained from public vulnerability databases. These may also be directly available from existing attack graphs of known vulnerabilities. One subtlety here is that the exploits not reachable from the asset can no longer be omitted since they may now be reachable from the asset with the help of zero day exploits.

Traditional attack graphs are practical for realistic applications, with efficient implementations (e.g., the MulVAL project [27]) and commercial tools (e.g., the CAULDRON tool [15]) available. A zero day attack graph would have comparable complexity as traditional attack graphs, because the number of added zero day exploits (which depends on the number of remote services and privileges) on each host should be comparable to the number of known vulnerabilities.

3) k-Zero Day Safety Metric Model: To instantiate the k-zero day safety metric model, we need to collect

- initial conditions (initially satisified conditions),
- an asset condition (or, in a more general form, logic clauses of multiple conditions [41]), and
- the equivalence relation $\equiv_v$ (Section III).

In our model, the notion of initial condition may refer to either a fact (e.g., existence of a service or connectivity) or an assumption (e.g., attackers' existing privilege on a host due to insider attack or user mistakes). In the former case, initial conditions are already part of the network model. In the latter case, determining initial conditions will require examining facts (e.g., access control policies and users' relative experiences) and then estimating potential risk (e.g., attackers are less likely to have initial privilege on a well guarded server than on a desktop shared by many inexperienced users). The asset condition(s) needs to be determined base on the relative value or importance of hosts. Finally, instantiating the equivalence relation between zero day exploits of two remote services requires examining the similarity between such services (and underlying OS and applications), and instantiating the relation between zero day exploits of a remote service and a privilege requires examining the existence and strength of isolation techniques around that service.

We note that the above subtleties in determining initial conditions and equivalence relations arise mainly because those concepts are designed as a means for handling uncertain information (e.g., the human factor). There exists an inherent trade-off between the effort required for collecting and estimating such information, and the accuracy and quality of the resultant model. While the model can still be applied even

when drastic approaches are taken toward such information (e.g., simply assuming insider attack or user mistakes to be absent), the instantiated model will not be as accurate as it can be with more refined and accurate input information (which also demands more effort). In an extreme case, an overly conservative assumption may lead to a trivial result (e.g., no network is 1-zero day safe, if every host is considered to have insider attacks). While such an assumption may be the safest and easiest choice, it is also the least helpful in terms of improving the security (since nothing can be done).

VI. Case Study

In this section, we illustrate through a series of case studies that our metric can reveal interesting and sometimes surprising results, which are not always obvious even for a small network; for larger and more realistic networks, the systematic approach to security evaluation using the metric and algorithms will thus become even more important.

A. Diversity

It is a common belief that greater diversity in software and services may help to improve networks' security. However, there lacks a precise approach to actually determining when, and how, diversity will help security. In this case study, we show that diversity does not always mean more security through applying the proposed metric.

The upper half of Figure 7 shows a small network in which services running on each host are marked beside that host and firewall rules are depicted below each firewall. Unless explicitly stated otherwise, we will assume different services or firewalls involve different zero day vulnerabilities. We also assume that none of the services, except $iptables$ and $tcpwrapper$, are protected by sufficient isolation. No known vulnerabilities are assumed in the services. Finally, suppose our main security concern is over host 4's root privilege.

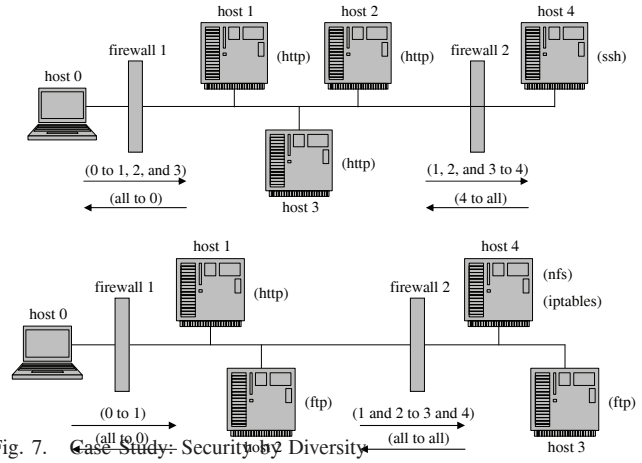

Fig. 7. Case Study: Security by Diversity

Now, suppose the three Web servers (host 1 through 3) are providing the $http$ service using the same software such that their corresponding zero day vulnerabilities are related by the $\equiv_v$ relation. This lack of diversity seems to result in poor security since one zero day vulnerability will compromise all three servers. However, by applying the k-zero day safety metric, we can see that k would remain the same regardless of the degree of diversity in these $http$ services, because any shortest attack sequence will only involve one of these three services (e.g., $\langle v_{http}, 0, 1\rangle, \langle v_{ssh}, 1, 4\rangle$). Therefore, increasing diversity will not increase k in this case.

In the above case, one may argue that the reason diversity does not help security is that the three Web servers are, intuitively speaking, in parallel to the asset (host 4). However, such informal observations will not lead to a general solution, as illustrated by the lower half of Figure 7. In this network (with the same assumptions as above), we are concerned with the diversity in the ftp services on host 2 and 3, which are clearly not in parallel to the asset (host 4), so the above observation will not apply to this second case.

Assume the $iptables$ services on host 4 only accept requests from host 2 and 3. Given that host 2 and 3 are directly accessible from each other, compromising host 2 through a zero day vulnerability will also compromise host 3. It thus seems tempting to prevent this situation by diversifying the ftp services on host 2 and 3. However, by applying the k-zero day safety metric, we will find that such a hardening option actually does not help.

Suppose we use ftp_x and ftp_y to indicate two different ways for providing the ftp service on host 2 and 3 such that their corresponding zero day vulnerabilities are not related by $\equiv_v$. We can then find that the shortest attack sequences of the original network (before diversifying the ftp services) are $\langle v_{http}, 0, 1\rangle, \langle v_{ftp_x}, 1, 2\rangle, \langle v_{nfs}, 2, 4\rangle$ and $\langle v_{http}, 0, 1\rangle, \langle v_{ftp_y}, 1, 3\rangle, \langle v_{nfs}, 3, 4\rangle$; the shortest attack sequences after diversifying the ftp services become $\langle v_{http}, 0, 1\rangle, \langle v_{ftp_x}, 1, 2\rangle, \langle v_{nfs}, 2, 4\rangle$ and $\langle v_{http}, 0, 1\rangle, \langle v_{ftp_y}, 1, 3\rangle, \langle v_{nfs}, 3, 4\rangle$. That is, diversifying the ftp services does not help increasing k.

This case study indicates that increasing diversity in hosts and services does not always help improving a network's security. More importantly, the way diversity affects security is not always straightforward even for a small network as depicted above, and intuitive observations or estimations may easily lead to incorrect and misleading results, which will certainly be exasperated in larger and more complex networks. On the other hand, the proposed k-zero day safety model and algorithms will automate such a daunting task and provide a meaningful evaluation about how diversity affects security in any reasonably large networks.

B. Known Vulnerability and Unnecessary Service

In this case study, we show how the existence of known vulnerabilities and unnecessary services, which may seem innocent enough at first glance, may actually affect the k-zero day safety of a network. The case study will also demonstrate that patching known vulnerabilities does not always improve the network's resistance to zero day attacks; a formal approach thus becomes necessary to evaluate the effectiveness of, and to prioritize, such patching tasks.

In the upper half of Figure 8, assume no known vulnerabilities and we are mainly concerned by the root privilege on host 5. Assume host 4 is an administration client, and consider the effect of leaving an unnecessary rsh service

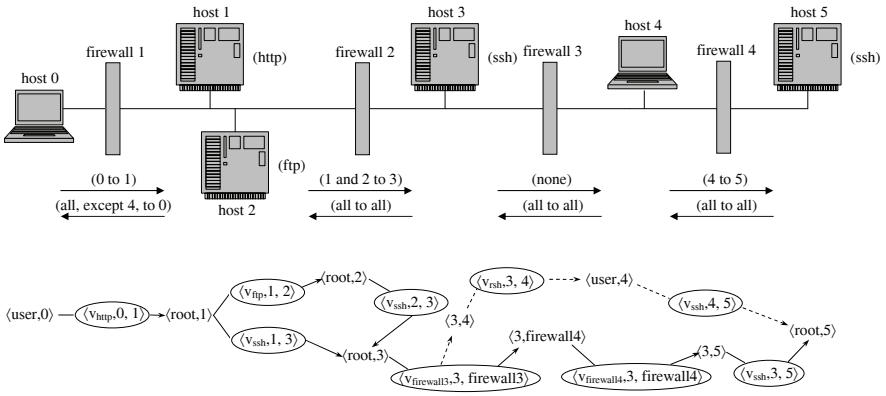

Fig. 8. Case Study: Removing Unnecessary Services and Known Vulnerabilities

running on host 4 and additionally the effect of introducing a known vulnerability v_{rsh} into that service. To existing techniques, such as an attack graph-based analysis, these may seem irrelevant to the security of host 5 since host 5 cannot be reached from host 0 anyway (due to firewall 3). However, by applying our metric, we will reach different conclusions.

The lower half of Figure 8 shows two attack sequences leading to the root privilege on host 5 (note that we have omitted other, longer attack sequences for simplicity). The edges in dashed lines correspond to attacks that become possible after introducing the rsh service and the corresponding known vulnerability mentioned above.

First, without the rsh service on host 4, as indicated by the lower attack sequence, the attacker would need to first exploit a zero day vulnerability v_{http} on host 1, v_{ssh} on host 3, and subsequently he/she will have to get around firewall 3 and 4 through $v_{firewall3}$ and $v_{firewall4}$ (assumed to be different), before he/she can attack host 5 from host 3 through exploiting v_{ssh} again. Therefore, totally four different zero day vulnerabilities will be needed in this case.

Now if service rsh is left running on host 4, but without any known vulnerability, then the upper attack sequence (part of which is in dashed lines) will become possible, with a new zero day vulnerability v_{rsh}. Although this does not actually change k in this case (with v_{rsh} replacing $v_{firewall4}$), it is easy to see that by further assuming v_{rsh} to be a known vulnerability, k will be reduced by 1.

Next, consider introducing a known vulnerability in the ftp service on host 2. From the attack sequences shown in the lower half of Figure 8, it is clear that such a known vulnerability does not give attackers any advantage in terms of reducing k, and therefore patching this vulnerability will not help to make the network more secure.

This case study illustrates that not every unnecessary service or known vulnerability will have the same effect on security. In practice, since removing a service or patching known vulnerabilities will usually incur a cost (e.g., administrative effort and cost for software patch or hardware upgrade), these activities should be prioritized based on their actual effect on security of the network.

C. Backup of Asset

In this case study, we will show that by placing an asset backup at different locations inside a network, the amount of security with respect to that asset may actually either increase, decrease, or remain the same.

In Figure 9, assume we are most concerned by the root privilege on host 4. We also assume that a known vulnerability exists in the $http$ service on both host 1 and 5, exploiting which provides root privilege on the host. Finally, assume we have chosen three candidate positions for placing a backup server for host 4, as indicated by the three dashed line circles.

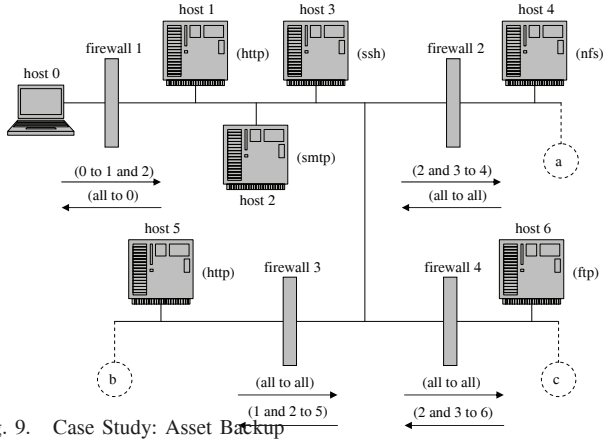

Fig. 9. Case Study: Asset Backup

First of all, without introducing any asset backup, we may find the three shortest attack sequences to be: $[\langle v_{http}, 0, 1\rangle, \langle v_{ssh}, 1, 3\rangle, \langle v_{nfs}, 3, 4\rangle]$, $[\langle v_{http}, 0, 1\rangle, \langle v_{smtp}, 1, 2\rangle, \langle v_{nfs}, 2, 4\rangle]$, and $[\langle v_{smtp}, 0, 2\rangle, \langle v_{nfs}, 2, 4\rangle]$. Note that v_{http} is a known vulnerability, and therefore, two different zero day vulnerabilities are needed to compromise host 4.

Next, consider setting up a backup server for host 4:

- First, consider placing the backup server, host 7, at location a. We can see that k will not change, because the same zero day vulnerability of the nfs service can compromise both host 4 and 7.
- Second, consider placing host 7 at location b, and changing firewall rules such that host 4 is directly accessible from host 7 for backup purposes. We can now find that the shortest attack sequence becomes $[\langle v_{http}, 0, 1\rangle, \langle v_{http}, 1, 5\rangle, \langle v_{nfs}, 5, 7\rangle, \langle v_{nfs}, 7, 4\rangle]$. Now, only one zero day vulnerability (recall v_{http} is a known vulnerability) is required, and k actually decreases by 1.
- Third, if we place host 7 at location c, we can see that the shortest attack sequence to gain root privileges on both host 4 and 7 now becomes longer:

$[\langle v_{smtp}, attacker, 2\rangle, \langle v_{ftp}, 2, 6\rangle, \langle v_{nfs}, 6, 7\rangle, \langle v_{nfs}, 7, 4\rangle]$, which requires three different zero day vulnerabilities.

D. Firewall

In this case study, we apply the metric to evaluate the effectiveness of firewalls.

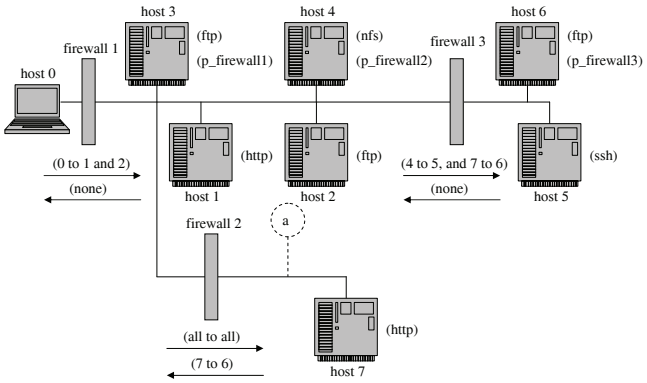

Fig. 10. Case Study: Firewall

In Figure 10, a personal firewall on host 3 allows inbound connection requests from host 1 and outbound requests to host 4 only. The firewall on host 4 allows inbound requests from host 3 and outbound requests to host 5. The firewall on host 6 allows inbound requests from 5 or 7. Moreover, we assume the personal firewall service on host C has a known vulnerability that may allow attackers to establish connections to the ftp service running on host 3. We are most concerned with the root privilege on host 6.

We can show that the shortest attack sequences are $[\langle v_{ftp}, 0, 2\rangle, \langle v_{p\_firewall1}, 2, 3\rangle, \langle v_{ftp}, 2, 3\rangle, \langle v_{nfs}, 3, 4\rangle, \langle v_{ssh}, 4, 5\rangle, \langle v_{ftp}, 5, 6\rangle]$ and $[\langle v_{ftp}, 0, 2\rangle, \langle v_{firewall2}, 2, firewall2\rangle, \langle v_{http}, 2, 7\rangle, \langle v_{ftp}, 7, 6\rangle]$. Since $v_{p\_firewall1}$ is known, both sequences require two different zero day vulnerabilities.

Suppose now, as a temporary workaround, the administrator decides to move host 3 to location a behind firewall 2, and remove its personal firewall $p\_firewall1$ but keep the same network access control by adding extra rules to firewall 2 to only allow connection requests from 1 to 3 and from 3 to 4.

On first glance, the above solution may seem a reasonable approach. However, by applying the metric, we can show that doing this will actually render the network less secure. Specifically, after moving host 3 to new location a, it can be shown that the shortest attack sequence becomes $[\langle v_{http}, 0, 1\rangle, \langle v_{ftp}, 1, 3\rangle, \langle v_{http}, 3, 7\rangle, \langle v_{ftp}, 7, 6\rangle]$, which requires only 2 different zero day vulnerabilities, and k decreases by 1 (this is mainly due to the new connectivity from 3 to 7).

E. Stuxnet and SCADA Security

The discovery of the high profile worm Stuxnet has drawn much attention to the security of supervisory control and data acquisition (SCADA) systems. This section presents a case study of Stuxnet and SCADA security in order to demonstrate

- why a network needs to be evaluated against the threat of (multiple) zero day attacks.
- how a threat such as Stuxnet may potentially be mitigated by applying our metric.
- how industry best practices on SCADA security are captured, and may be evaluated, by our metric.

First of all, one interesting fact about Stuxnet is that it employs four different zero day attacks for spreading itself [8]. This fact alone suffices to show that, in a mission critical system such as SCADA, the risk of zero day attacks is very real, and such risk may indeed come from more than one zero day vulnerabilities all at the same time. Therefore, it makes perfect sense for administrators to evaluate the security of such systems against such risk, and the k-zero day safety metric proposed in this paper provides one such solution.

Second, we examine the propagation methods of Stuxnet. It can distribute itself among Windows machines through a number of vulnerabilities involving USB flash drive, network share, peer-to-peer RPC, and Print Spooler [8]. Among those we can see that the last three will all be represented as remote services in our model, and hence is assigned with a zero day vulnerability. This will allow administrators to immediately identify potential threats if a machine with those services running is connected or close to a critical asset (e.g., PLC in this case). As to the vulnerability involving USB flash drive, it can certainly be modeled as a potential user mistake through an initial condition representing attackers' privilege, although such modeling is only helpful if appropriate policies about physical security are in place (e.g., policies preventing USB drives to be used on critical machines). In summary, applying our metric may help administrators to identify and hence mitigate such potential threats of zero day attacks.

Next, we study the recommended practice on improving SCADA security by Homeland Security [38]. As illustrated in Figure 11, this recommended practice entails following main security strategies:

- The enterprise network is divided into different architectural zones, as illustrated by four different colored background, with the most critical zone (the control zone) being furthermost from external infrastructures.
- Firewalls are placed between different zones and besides the DMZs to regulate traffic flows.
- Multiple DMZs are created for separate functionalities and access privileges.
- IDS sensors, as illustrated by the blue dots, are placed at strategic locations in the network.

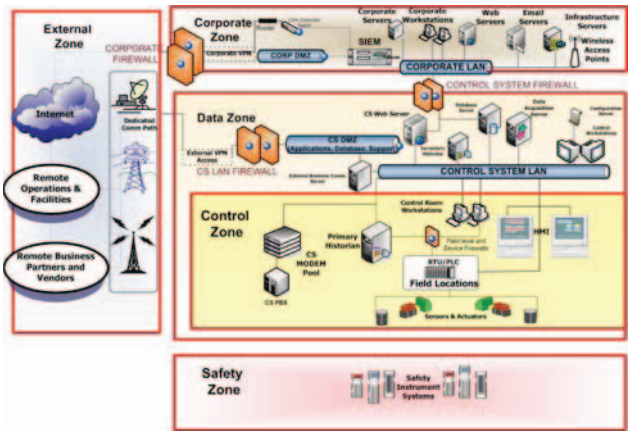

Fig. 11. Case Study: SCADA Security [38]

Clearly, those security strategies closely match the network hardening options described in Section V-A. Specifically, dividing the network into different zones, placing more critical zones further away from the network perimeter, and regulating network traffic using firewalls and DMZs all have the effect of increasing the length of shortest attack sequences, and thus may lead to better security. Introducing IDSs has the potential effect of forcing attackers to avoid certain hosts (to evade detection), which is captured by negation in the asset formula (details can be found in [41]). More importantly, the effectiveness of those recommended security strategies can now be more precisely evaluated using our metric.

In addition, we can easily see that all the network hardening options discussed in Section V-A will also apply in this case, and may help to prevent a threat like Stuxnet from propagating to the critical asset. Specifically, Stuxnet would need to first infect Windows computers inside either the corporate zone or data zone using one of the aforementioned vulnerabilities, and then it must spread itself into the control zone, and cover the final hop through removable drives (since field machines are typically never connected to an untrusted network) [8]. This will become much harder when the network has more diversity (e.g., smaller groups of Windows machines), stronger isolation (e.g., services running inside virtual machines), stricter access control and physical security policies (e.g., machines in the data and control zones are only accessible to experienced users, and removable media are prohibited or under more scrutinises in the control zone), up-to-date patching of vulnerabilities (e.g., Stuxnet also employs known vulnerabilities used by Conficker [8]), etc. It may be safely claimed that such a network, if sufficiently hardened using our metric, will be much less susceptible to a threat like Stuxnet.

VII. Conclusion

In this paper, we have proposed the k-zero day safety as a novel network security metric, discussed its computation and application, and demonstrated its power in practical scenarios. Specifically, we formally defined the k-zero day safety model and showed that the metric satisfied the required algebraic properties of a metric function. We then studied the complexity of computing the metric and proposed efficient algorithms for determining the metric value. Next, we applied the proposed metric to the practical issue of network hardening and extended the metric to characterize various hardening options; we also discussed in details how the abstract model may be instantiated for given networks in practice. Finally, we demonstrated how applying the proposed metric may lead to interesting and sometimes surprising results through a series of case studies; we also discussed how the metric may potentially be applicable to SCADA security.

A. Limitations and Future Work

We discuss several aspects of the proposed metric in which further improvements and evaluations are still needed.

1) Ranking Zero Day Vulnerabilities: We have regarded all zero day vulnerabilities as equally likely due to their commonly perceived unmeasurability. However, in some cases certain assumptions can be safely made about the relative likelihood of different zero day vulnerabilities (e.g., some OSs are generally considered more secure than others). Assigning different weights or probabilities to different (types of) zero day vulnerabilities would be a natural extension to our model.

2) Handling Uncertain Inputs: As discussed above, instantiating the metric model may involve uncertain input information (e.g., the possibility of insider attacks). Since our model is deterministic in nature, the only way to handle such uncertainty is through making conservative assumptions which leads to a lower metric value k (e.g., modeling the possibility of insider attack as initial conditions). An important future direction would be to develop a more refined model (e.g., a probabilistic approach) to model such uncertain information.

3) Known Vulnerabilities: In a zero-day attack graph, known vulnerabilities only affect the metric value through serving as a shortcut for attackers to bypass zero day exploits. The relative severity of different known vulnerabilities is not taken into consideration in the metric. An interesting topic for future research is to integrate the k-zero day safety metric with existing metrics of known vulnerabilities (an obvious solution, such as a weighted sum of the two metrics, may not make sense due to the metrics' different semantics).

4) Scope of Application: The scope of our metric is limited by the three basic assumptions about zero day vulnerabilities (the existence of network connectivity, vulnerable services on destination host, and initial privilege on source host). The model will be more suitable for application to the evaluation of penetration attacks launched by human attackers or network propagation of worms or bots in mission critical networks. An important future work is to broaden the scope by accommodating other types of attacks (e.g., a time bomb which requires no network connection).

5) Field Application and Evaluation: The field application and evaluation of the proposed metric is another important future work. The main difficulty in empirically evaluating a security metric lies in the lack of necessary benchmark data (such an evaluation would require both attack data and details of the networks, including detailed host configurations, firewall rules, user access control policies, etc., and the data must be representative enough in terms of both attacks and networks). One viable approach would be to integrate the proposed metric as an added feature to existing vulnerability analysis tools, such as CAULDRON [15], in order to evaluate its practical effectiveness and to fine-tune the model.

References

[1] P. Ammann, D. Wijesekera, and S. Kaushik. Scalable, graph-based network vulnerability analysis. In *Proceedings of ACM CCS'02*, 2002.
[2] D. Balzarotti, M. Monga, and S. Sicari. Assessing the risk of using vulnerable components. In *Proceedings of the 1st ACM QoP*, 2005.
[3] S. M. Bellovin. On the brittleness of software and the infeasibility of security metrics. *IEEE Security and Privacy*, 4:96–, July 2006.
[4] M. Dacier. Towards quantitative evaluation of computer security. Ph.D. Thesis, Institut National Polytechnique de Toulouse, 1994.
[5] E. W. Dijkstra. A note on two problems in connection with graphs. *Numerische Mathematik*, 1:269271, 1959.
[6] J. Doob. *Measure Theory*. Springer-Verlag, 1994.
[7] C. Dwork. Differential privacy. In *ICALP (2)*, pages 1–12, 2006.
[8] N. Falliere, L. O. Murchu, and E. Chien. W32.stuxnet dossier. Symantec Security Response, 2011.

[9] M. Frigault, L. Wang, A. Singhal, and S. Jajodia. Measuring network security using dynamic bayesian network. In *Proceedings of 4th ACM QoP*, 2008.
[10] A. Greenberg. Shopping for zero-days: A price list for hackers' secret software exploits. *Forbes*, 23 March 2012.
[11] H. Holm, M. Ekstedt, and D. Andersson. Empirical analysis of system-level vulnerability metrics through actual attacks. *IEEE Trans. Dependable Secur. Comput.*, 9(6):825–837, Nov. 2012.
[12] J. Homer, X. Ou, and D. Schmidt. A sound and practical approach to quantifying security risk in enterprise networks. Technical Report, 2009.
[13] N. Idika and B. Bhargava. Extending attack graph-based security metrics and aggregating their application. *IEEE Transactions on Dependable and Secure Computing*, 9:75–85, 2012.
[14] K. Ingols, M. Chu, R. Lippmann, S. Webster, and S. Boyer. Modeling modern network attacks and countermeasures using attack graphs. In *Proceedings of ACSAC'09*, pages 117–126, 2009.
[15] S. Jajodia, S. Noel, and B. O'Berry. Topological analysis of network attack vulnerability. In V. Kumar, J. Srivastava, and A. Lazarevic, editors, *Managing Cyber Threats: Issues, Approaches and Challenges*. Kluwer Academic Publisher, 2003.
[16] A. Jaquith. *Security Merics: Replacing Fear Uncertainity and Doubt*. Addison Wesley, 2007.
[17] S. Jha, O. Sheyner, and J. Wing. Two formal analysis of attack graph. In *Proceedings of the 15th Computer Security Foundation Workshop (CSFW'02)*, 2002.
[18] D. Leversage and E. Byres. Estimating a system's mean time-to-compromise. *IEEE Security and Privacy*, 6(1):52–60, 2008.
[19] W. Li and R. B. Vaughn. Cluster security research involving the modeling of network exploitations using exploitation graphs. In *Proceedings of the Sixth IEEE International Symposium on Cluster Computing and the Grid*, CCGRID '06, pages 26–, Washington, DC, USA, 2006. IEEE Computer Society.
[20] R. Lippmann, K. Ingols, C. Scott, K. Piwowarski, K. Kratkiewicz, M. Artz, and R. Cunningham. Validating and restoring defense in depth using attack graphs. In *Proceedings of the 2006 IEEE conference on Military communications*, MILCOM'06, pages 981–990, Piscataway, NJ, USA, 2006. IEEE Press.
[21] J. McHugh. Quality of protection: Measuring the unmeasurable? In *Proceedings of the 2nd ACM QoP*, pages 1–2, 2006.
[22] M. McQueen, T. McQueen, W. Boyer, and M. Chaffin. Empirical estimates and observations of 0day vulnerabilities. *Hawaii International Conference on System Sciences*, 0:1–12, 2009.
[23] V. Mehta, C. Bartzis, H. Zhu, E. Clarke, and J. Wing. Ranking attack graphs. In *Recent Advances in Intrusion Detection 2006*, 2006.
[24] P. Mell, K. Scarfone, and S. Romanosky. Common vulnerability scoring system. *IEEE Security & Privacy*, 4(6):85–89, 2006.
[25] National vulnerability database. available at: http://www.nvd.org, May 9, 2008.
[26] R. Ortalo, Y. Deswarte, and M. Kaaniche. Experimenting with quantitative evaluation tools for monitoring operational security. *IEEE Trans. Software Eng.*, 25(5):633–650, 1999.
[27] X. Ou, W. Boyer, and M. McQueen. A scalable approach to attack graph generation. In *Proceedings of the 13th ACM conference on Computer and communications security*, CCS'06, pages 336–345, New York, NY, USA, 2006. ACM.
[28] J. W. P. Manadhata. An attack surface metric. Technical Report CMU-CS-05-155, 2005.
[29] J. Pamula, S. Jajodia, P. Ammann, and V. Swarup. A weakest-adversary security metric for network configuration security analysis. In *Proceedings of the ACM QoP*, pages 31–38, 2006.
[30] C. Phillips and L. Swiler. A graph-based system for network-vulnerability analysis. In *Proceedings of the New Security Paradigms Workshop (NSPW'98)*, 1998.
[31] N. Poolsappasit, R. Dewri, and I. Ray. Dynamic security risk management using bayesian attack graphs. *IEEE Trans. Dependable Secur. Comput.*, 9(1):61–74, Jan. 2012.
[32] P. Samarati. Protecting respondents' identities in microdata release. In *IEEE Transactions on Knowledge and Data Engineering (TKDE)*, pages 1010–1027, 2001.
[33] R. Savola. Towards a taxonomy for information security metrics. In *Proceedings of the 3rd ACM QoP*, pages 28–30. ACM, 2007.
[34] M. Shahzad, M. Shafiq, and A. Liu. A large scale exploratory analysis of software vulnerability life cycles. In *Proceedings of the 34th International Conference on Software Engineering (ICSE)*, 2012.
[35] O. Sheyner, J. Haines, S. Jha, R. Lippmann, and J. Wing. Automated generation and analysis of attack graphs. In *Proceedings of the IEEE S&P'02*, 2002.
[36] T. Sommestad, H. Holm, and M. Ekstedt. Effort estimates for vulnerability discovery projects. In *Proceedings of the 2012 45th Hawaii International Conference on System Sciences*, HICSS '12, pages 5564–5573, Washington, DC, USA, 2012. IEEE Computer Society.
[37] The MITRE Corporation. Common weakness scoring system. http://cwe.mitre.org/cwss/, 2010.
[38] U.S. Department of Homeland Security. Recommended practice: Improving industrial control systems cybersecurity with defense-in-depth strategies. https://www.us-cert.gov/control_systems/practices/Recommended_Practices.html, 2009.
[39] V. Verendel. Quantified security is a weak hypothesis: a critical survey of results and assumptions. In *Proceedings of the 2009 NSPW*, pages 37–50. ACM, 2009.
[40] L. Wang, T. Islam, T. Long, A. Singhal, and S. Jajodia. An attack graph-based probabilistic security metric. In *Proceedings of the 22nd IFIP DBSec*, 2008.
[41] L. Wang, S. Jajodia, A. Singhal, and S. Noel. k-zero day safety: Measuring the security risk of networks against unknown attacks. In *Proceedings of the 15th ESORICS*, pages 573–587, 2010.
[42] L. Wang, S. Noel, and S. Jajodia. Minimum-cost network hardening using attack graphs. *Computer Communications*, 29(18):3812–3824, 11 2006.
[43] L. Wang, A. Singhal, and S. Jajodia. Measuring network security using attack graphs. In *Proceedings of the 3rd ACM QoP*, New York, NY, USA, 2007. ACM Press.

Lingyu Wang Lingyu Wang is an associate professor in the Concordia Institute for Information Systems Engineering (CIISE) at Concordia University, Montreal, Quebec, Canada. He received his Ph.D. degree in Information Technology from George Mason University. He holds a M.E. from Shanghai Jiao Tong University and a B.E. from Shenyang Aerospace University in China. His research interests include data privacy, network security, and security metrics. He has co-authored over 70 refereed conference and journal articles.

Sushil Jajodia Sushil Jajodia is University Professor, BDM International Professor, and the director of Center for Secure Information Systems in the Volgenau School of Engineering at the George Mason University, Fairfax, Virginia. He received his PhD from the University of Oregon, Eugene. The scope of his current research interests encompasses information secrecy, privacy, integrity, and availability. He has authored or coauthored six books, edited 39 books and conference proceedings, and published more than 400 technical papers in the refereed journals and conference proceedings. He is also a holder of ten patents and has several patent applications pending. He has supervised 26 doctoral dissertations. Nine of these graduates hold tenured positions at U.S. universities; four are NSF CAREER awardees and one is DoE Young Investigator awardee. Two additional students are tenured at foreign universities. Dr Jajodia received the 1996 IFIP TC 11 Kristian Beckman award, 2000 Volgenau School of Engineering Outstanding Research Faculty Award, 2008 ACM SIGSAC Outstanding Contributions Award, and 2011 IFIP WG 11.3 Outstanding Research Contributions Award. He was elected a fellow of IEEE in January, 2013. He was recognized for the most accepted papers at the 30th anniversary of the IEEE Symposium on Security and Privacy. His h-index is 72 and Erdos number is 2. The URL for his web page is http://csis.gmu.edu/jajodia.

Anoop Singhal Anoop Singhal is currently a Senior Computer Scientist in the Computer Security Division at NIST. His research interests are in network security, cloud computing security and data mining systems. He is a senior member of IEEE and he has published several papers in leading conferences and journals. He received his Ph.D. in Computer Science from Ohio State University, Columbus, Ohio.

Pengsu Cheng Pengsu Cheng is a security analyst at Gameloft, working on vulnerability analysis. He received his M.A.Sc. degree in Information Systems Security from Concordia University, Canada in 2011, and B.E. in Electronic Information Engineering from Wuhan University, China in 2009. His research interests include security metrics, vulnerability analysis and intrusion detection.

Steven Noel Steven Noel is Associate Director of the Center for Secure Information Systems at George Mason University. Dr. Noel has led a team of Mason scientists and engineers developing breakthrough patented technology for cyber attack modeling, analysis, and visualization (Cauldron). His research interests include network attack graph modeling, intrusion detection, and visualization for information security. He received his Ph.D. in Computer Science from the University of Louisiana at Lafayette in 2000, with dissertation work in data mining and visualization for information retrieval.