Measuring and Modeling Software Vulnerability Security Advisory Platforms

Lucas Miranda[1], Daniel Vieira[1], Mateus Nogueira[1], Leonardo Ventura[1], Miguel Bicudo[1], Matheus Martins[1,2], Lucas Senos[1], Leandro P. de Aguiar[2], Enrico Lovat[2], and Daniel Menasche[1(✉)]

[1] Federal University of Rio de Janeiro, Rio de Janeiro, Brazil
sadoc@dcc.ufrj.br

[2] Siemens Corporate Research, Princeton, NJ, USA

Abstract. In this paper, we report results on a large scale measurement campaign to collect temporal information about events associated with software vulnerabilities. The data is curated so as to extract dates from each of the analyzed security advisories. The resulting time series are our object of study. From our measurements we were able to identify which role was assumed by different platforms (such as websites and forums) in the security landscape, including sources and aggregators of information about vulnerabilities. Then, we propose an analytical model to express the flow of information through security advisories across multiple platforms. The model is based on a queueing network, where each platform corresponds to a queue which adds a delay in the information propagation. Such delays, in turn, have an impact on the visibility of the information at different platforms. Leveraging the proposed model and the collected data, we assess how different system parameters, such as the delays incurred by each platform to propagate its messages, impact the overall flow of information across platforms.

1 Introduction

Background. Software vulnerabilities which threaten modern computer systems are disclosed in a daily basis. Platforms such as the National Vulnerability Database (NVD) and vendor websites are responsible for informing users about such vulnerabilities, through security advisories. Each vulnerability, identified through its Common Vulnerabilities and Exposures identifier (CVE id), is typically associated with many security advisories, which reflect, for instance, the availability of exploits and patches [13].

Timely information about early disclosure of vulnerabilities is key. Accurate information about the impact of vulnerabilities and about which systems are exposed helps to guide decisions, e.g., related to risk aware patch management [19]. Although security platforms play a fundamental role in the security landscape, the flow of information among them is still poorly understood. *In this paper, our goal is to report measurements, and present models and methods to provide insight into how information about vulnerabilities flows across security platforms.*

J. Garcia-Alfaro et al. (Eds.): CRiSIS 2020, LNCS 12528, pp. 31–48, 2021.
https://doi.org/10.1007/978-3-030-68887-5_2

Challenges. There are several challenges pertaining a characterization of the flow of information across security platforms. First, each security platform uses its own format to disseminate data about vulnerabilities. The information retrieved from those platforms must be curated, e.g., to extract publication dates by the sources. Second, there are no models to capture the dynamics of how information flows in the ecosystem. Such models can be instrumental to assess what would occur to the security landscape given events such as important platforms being attacked, e.g., by a DDoS attack, or two platforms merging together. As an example, relevant platforms, such as Security Focus and Security Tracker, have not been recently updated, and the impact is still unclear. Third, some security advisories may be shared in private platforms or in black hat forums, making them unreachable to the general public.

Gaps in Prior Art. The quality and timeliness of security advisories have been considered, for instance, in the context of threat information feeds [12]. However, most of those results are derived from data collected under restrictive non-disclosure agreements, and may not be reproducible in a non-industrial setting. One of our goals is to analyze security advisories made available through public platforms, referred by NVD as authoritative sources for information about vulnerabilities. In addition, there are works considering the flow of vulnerabilities across *software modules* [8]. Nonetheless, we are not aware of previous work on measurements and models of flow of advisories across *software platforms*.

Methods. To tackle the above challenges, we report results on a large scale measurement campaign to collect temporal information about events associated with software vulnerabilities. The data is curated so as to extract dates from each of the analyzed security advisories. The resulting time series of security advisories associated with vulnerabilities are our objects of study. From our measurements we are able to identify which role was assumed by different platforms (such as websites and forums) in the security landscape, including sources and aggregators of information about vulnerabilities. Then, we propose an analytical model to express the flow of information through security advisories across multiple platforms. The model is based on a network of queues, where each platform corresponds to a queue which adds a delay in the information propagation.

Contributions. In summary, our contributions are threefold.

Measurement Campaign and Insights: we retrieved security advisories from the top platforms referred to by NVD. By analyzing the time series of events associated with each vulnerability, we identified and quantified the extent at which platforms are used either as the ultimate sources of information about CVEs or as aggregators to forward advisories from its sources to other platforms.

Analytical Model: we propose and parametrize a queueing network comprised of $M/G/\infty$ queues to express the flow of information through security advisories across multiple platforms. The model is inspired by Sankey diagrams, which are derived from the collected data.

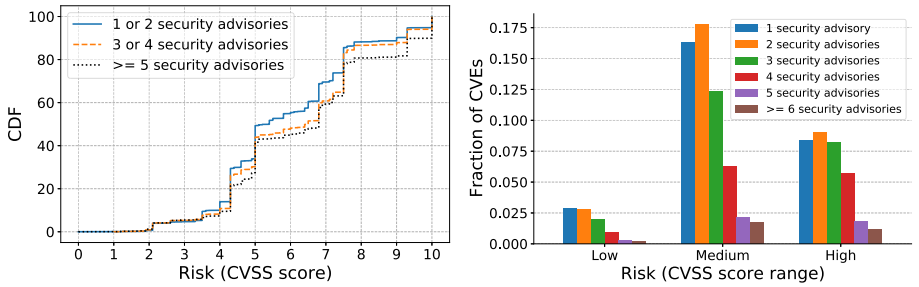

Fig. 1. CDF of risk given number of vulnerability advisories (left) and fraction of CVEs with given number of advisories (right): more advisories correlate with higher risk.

Model-Based Analysis: in light of the proposed analytical model, we extract additional insights about the security advisory ecosystem. Our model allows us to study how different system parameters, such as the order at which platforms issue advisories and the delays incurred by each platform to propagate its advisories, impact the mean number of days it takes for an authority such as NVD to report an advisory.

Paper Organization. The following section sets the ground in the realm of risk aware patch management and reports related work. Measurements and the analytical model are introduced in Sects. 3 and 4. The proposed model is evaluated using the collected measurements at Sect. 5 and Sect. 6 concludes.

2 Risk Aware Patch Management and Related Work

Patch management is key for product developers and asset owners. Patches imply outages explaining why industrial control systems (ICS) patches are typically deferred by 2–3 months [19]. Many enterprises help asset owners to manage risks associated to vulnerabilities, and risk aware vulnerability management products are available in the market; Siemens has the Industrial Vulnerability Manager (IVM) for ICS; Tenable has a complementary product for enterprise IT; the open source community has similar initiatives, e.g., the Exploit Prediction Scoring System (EPSS), a data-driven framework for assessing vulnerability threat.

Risk aware patch management (RAPM) involves tracking risk, e.g., CVSS or EPSS, ranging between 0 and 10, over time. It relies on data made available at security advisory platforms, e.g., to parameterize the temporal components of CVSS or EPSS. We illustrate the applicability of the results reported in the remainder of this paper through two examples in the realm of RAPM.

Assess Impact of Sources on Delays. The measurements and model in this paper support RAPM: if one plans patches based on information from a subset of platforms, how late may patches occur?

Estimate Confidence Levels of Risk. The proposed model supports what-if analysis for RAPM: what are the biases incurred when predicting risk? What is the confidence interval (CI) of risk estimates? How is CI expected to change?

Answering the above questions requires knowledge on how the number of data points associated with vulnerabilities varies over time, which is the main theme of this work. To illustrate that point, Fig. 1, obtained from our measurements (Sect. 3), shows the CDF of risk (CVSS) for three classes of vulnerabilities, varying according to the number of data points associated to each CVE. Vulnerabilities with 1 or 2 advisories tend to have lower risk when compared against vulnerabilities with 5 or more advisories, which attract more visibility. Figure 1(right) further supports that point, showing that vulnerabilities with high risk tend to have more advisories. Advisories are issued by multiple platforms, and confidence levels on risk build up during the period at which the advisories are made public. One of our aims is to measure and model the flow of advisories so as to understand how such confidence builds up across the lifecycle of vulnerabilities.

The lifecycle of vulnerabilities has been considered in previous works, accounting for weaponization and exploitation events [3,18], patching practices [19] and the role of CVSS [10,17] and TI feeds [12]. Our work considers the vulnerability lifecycle from a novel angle, accounting for platforms delays and leveraging Sankey diagrams and networks of queues. Whereas most works focus on predicting weaponization and exploitation, we focus on the flow of advisories across platforms, which is key for RAPM.

We rely on NVD as our reference to select the analyzed security platforms. Previous studies relied on NVD to cluster vulnerabilities [9] and to predict software vulnerabilities and corresponding risks [21]. None of these works leveraged the list of hyperlinks provided by NVD with advisories about each vulnerability, as considered in this paper.

Most of the literature on vulnerability disclosures tracks how vulnerabilities evolve within a software product [11] or across software modules [8]. In this paper, we take a different approach towards the evolution of software vulnerabilities, focusing on evolution across platforms. We believe that those approaches are complementary. In particular, one of the platforms considered in our study, Github [2,7,15], has within itself a rich history of software patches and upgrades whose timelines complement the time series considered in this paper.

3 Measurement Setup and Measurement Insights

Next, we describe our measurement setup, indicating how we selected the multiple sources of data and how we extracted temporal information from those. Then, we report insights obtained from the measurements.

Terminology. We begin by introducing some basic terminology.

CVE id is the Common Vulnerabilities and Exposures (CVE) identifier, which is a unique id used to refer to vulnerabilities.

CVE authority, also known as CVE numbering authority (CNA), is any entity that can issue CVE ids.

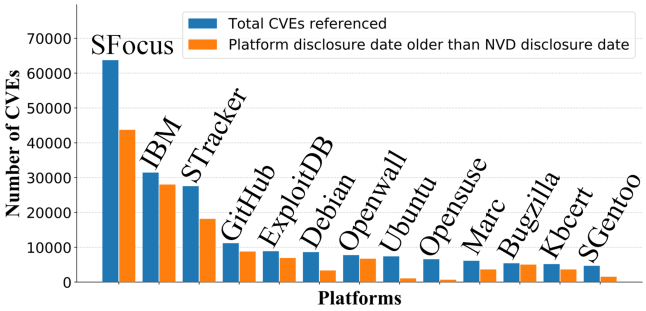

Fig. 2. CVEs per platform.

NVD CVE published date is the date at which the CVE was disclosed at the National Vulnerability Database (NVD). Both CVE and NVD are sponsored by the U.S. Department of Homeland Security (DHS). All vulnerabilities granted a CVE are eventually published by NVD [16].

Security advisory is any piece of information published about a given CVE, including patches, exploits and notes. *We encompass NVD disclosures as security advisories, whose publication dates equal NVD CVE published dates.*

Security advisory platform is a platform that issues security advisories. Examples include NVD and Security Focus.

Security advisory hyperlink is a hyperlink to a security advisory published at a given platform.

Security advisory date is the date at which the material contained in the security advisory was published by its author.

Platform CVE disclosure date is the earliest security advisory date for a given CVE among the collected advisories from a given platform. In particular, the NVD CVE disclosure date coincides with the NVD CVE published date.

CVE disclosure date is the earliest security advisory date for a given CVE among all collected advisories.

In the remainder of this paper, we contrast security advisory dates among platforms and against NVD CVE published dates.

Measurement Setup. We take NVD as our reference platform of security advisories. For each vulnerability, NVD provides its CVE, its NVD CVE published date, and hyperlinks to security advisories by other platforms. We select the top 13 platforms, in addition to NVD, ranked based on the number of encompassed CVEs (see Fig. 2).

For each vulnerability, NVD reports its CVE id together with a list of hyperlinks to security advisories. We download the content of those hyperlinks, and process the corresponding HTML files as described below. Note that for each of the considered platforms we find at least 1500 unique hyperlinks at NVD, and each advisory typically accounts for up to 2 CVEs.

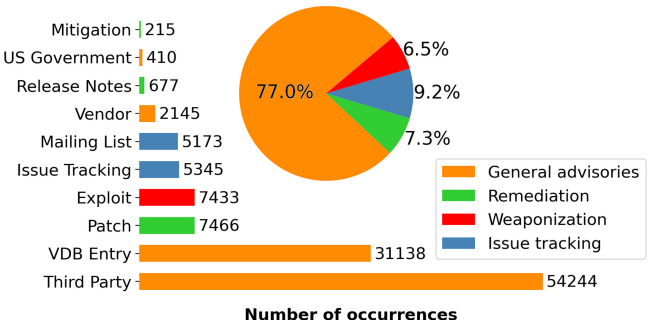

Fig. 3. Security advisory categories.

Accounting for the specifics of the format of the HTML files provided by each platform, we use `XPath`, the XML Path Language, to extract temporal information from each of those files. Each platform corresponds to a given `XPath` parametrization to extract the HTML element corresponding to the publication dates of the advisories posted by that platform. After implementing such process, for each security advisory we have its (*i*) publication date, (*ii*) NVD hyperlink, (*iii*) list of CVEs at NVD that contain that hyperlink. We store the elements in a database from which we derive the analysis and insights that follow.[1]

Dataset. Next, we report findings and insights obtained from our measurement campaign. We considered the NVD bundle of vulnerabilities accounting for all vulnerabilities released up to March 26, 2020. This amounts to a total of 576,750 references (hyperlinks) to 336,559 distinct security advisories (unique hyperlinks), from 129,504 CVEs. Then, we crawled the security advisory platforms of interest between March 27, 2020 and March 30, 2020, with March 25, 2020 being the latest CVE publication date at our database. *In the analysis that follows we account for 232,873 references to more than 108,889 distinct advisories released by 13 platforms referred by NVD, targeting 98,407 CVEs.*

Flow of Information Across Security Advisory Platforms. We begin by considering the nature of the security advisories exchanged in the considered platforms. To that aim, Fig. 3 indicates the distribution of the categories of the security advisories, as reported by NVD. Note that some advisories pertain to multiple categories and others have not been classified. The security advisories are distributed across remediation, weaponization, issue tracking and general advisories. In particular, information about exploitation of vulnerabilities in the wild is not captured through those advisories.

The flow of advisories across platforms can be captured through Sankey diagrams, as illustrated in Fig. 4. To produce Fig. 4, we consider for each vulnerability the sequence of platforms that published advisories about the corresponding CVE, ordered by their security advisory dates. The longest sequence in our

[1] The data and scripts to produce the reported results are available by contacting the authors.

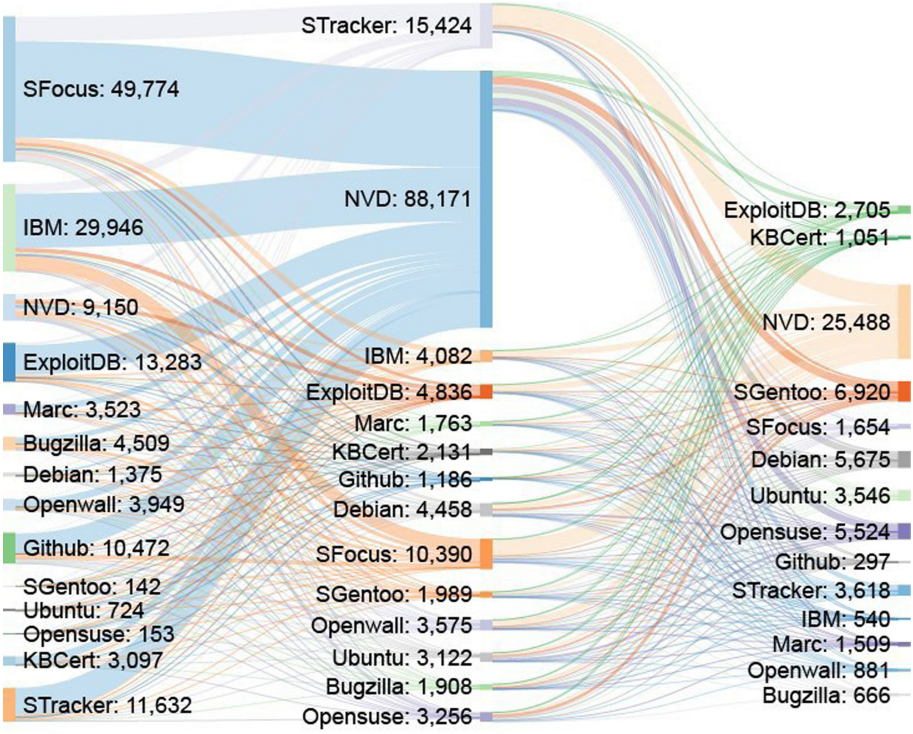

Fig. 4. Flow of information across security advisory platforms

dataset corresponds to CVE-2009-3555, and consists of 13 platforms. Heartbleed (CVE-2014-0196), Shellshock (CVE-2014-6271) and Meltdown (CVE-2017-5754) are vulnerabilities whose sequences comprise at least 6 platforms.

Given the sequences of platforms associated to each CVE, we extract the ordered pairs of consecutive platforms appearing in those sequences. Let $\mathcal{P}$ be the multiset of obtained ordered pairs. If two platforms A and B produce advisories for a given CVE at the same day, no entry is added to $\mathcal{P}$. If they are followed by C at an upcoming day, we add both (A, C) and (B, C) into $\mathcal{P}$. The Sankey diagram in Fig. 4 is generated from $\mathcal{P}$. The width of lines is proportional to the *CVE flow* between pairs of platforms, i.e., the width of the line between (A, B) is proportional to the frequency of ordered pair (A, B) in $\mathcal{P}$. The lines between layers 1 and 2 correspond to ordered pairs (A, B) wherein platform A is the first to publish an advisory about the corresponding CVE. The lines between layers 2 and 3 account for the additional ordered pairs. The analysis in this section accounts for CVEs published by NVD and by at least one additional platform, at a distinct day, noting that those correspond to more than 90% of the vulnerabilities in our dataset.

Figure 4 indicates that Security Focus and Security Tracker platforms play a relevant role in the security advisory ecosystem. Interestingly, they have not

been updated since the end of 2019. We envision that in a couple of years some of the platforms shown in Fig. 4 will replace Security Focus and Security Tracker, and one of our purposes is to present a methodology to track how the ecosystem of security advisories evolves over time.

Figure 4 allows us to further assess the contribution of different platforms with respect to the flow of information, accounting for the ordering at which advisories for CVEs flow across platforms. Some platforms behave as sources of information, while others as aggregators. Security Focus, for instance, is the source of advisories for 80% (49774/(10390 + 49774 + 1654)) of its CVE flows. NVD, in contrast, is the aggregator of advisories for 92% ((88171 + 25488)/(88171 + 9150 + 25488)) of its CVE flows in Fig. 4. Other platforms are more symmetric in the flow of CVEs. For example, KBCert is source of advisories for 49% (3097/(3097 + 2131 + 1051)) of its CVE flows.

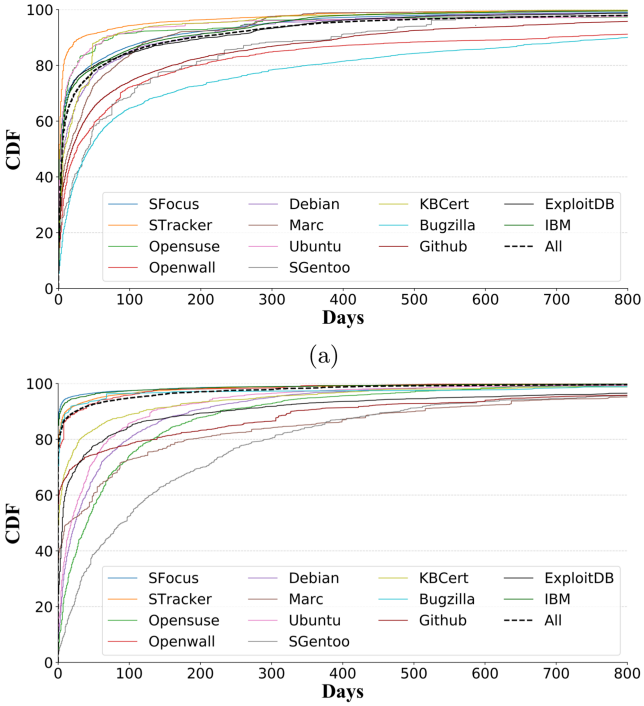

Fig. 5. NVD delay against platforms: (*a*) accounting for CVEs whose disclosure at platforms other than NVD occurred first and (*b*) accounting for CVEs whose disclosure at NVD occurred first.

NVD, as an aggregator, typically publishes CVEs after advisories have been announced at other platforms (see Fig. 2). Nonetheless, note that even though NVD is typically not the first platform to report information about vulnerabilities, in many instances it is the second to do so. As we move from the second to

the third layer, the height of the NVD bar decreases, indicating its diminishing relevance for advisories already published by more than one platform.

Figure 4 shows that security advisory platforms complement each other with respect to the timing at which they share information. It provides a birds eye of view of the flow of information across platforms, which we detail, analyze and model in the remainder of this work.

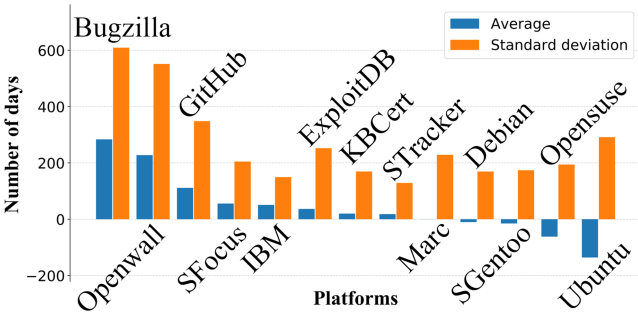

Fig. 6. NVD CVE published date minus source disclosure date.

NVD CVE Published Dates. Next, we aim at contrasting the date at which NVD publishes CVE information, as reported by NVD itself, in its NVD CVE published date field, against the date at which we first found information about the corresponding vulnerabilities in our dataset. Let f be the day at which we first find information about a given vulnerability in a platform other than NVD, and let n be the NVD CVE published date. Figures 5(a) and 5(b) show the cumulative distribution function (CDF) of $n - f$ and $f - n$ for the cases wherein $f \leq n$ and $f > n$, respectively, measured in days. Each solid line corresponds to a different platform. The dashed lines account for data across all platforms, ignoring platform identities to produce the plots.

Figure 5(a) indicates that NVD publishes advisories for up to 60% of the vulnerabilities in less than one month after the disclosure of the corresponding CVE. However, for the other 40% of the vulnerabilities, it may take more than one year for NVD to publish CVE information. The worst case scenario occurs when contrasting NVD against Bugzilla, for which we discover that after two years CVEs were not published at NVD for up to 15% of the vulnerabilities disclosed by Bugzilla.

Figure 5(b) indicates that when NVD is the first to release information about vulnerabilities, at least one of the other platforms quickly catches up. This is represented by the dashed line in Fig. 5(b), with a sharp increase close to zero. The corresponding dashed line in Fig. 5(a), in contrast, indicates that when NVD is not the first to release information, for roughly 20% of the vulnerabilities it may take up to three months for NVD to disclose the corresponding CVE.

Figure 6 provides further insights on how NVD positions against other platforms. Figure 6 shows the mean and standard deviation of NVD CVE published dates minus disclosure dates at other platforms, accounting for vulnerabilities that appeared at the two considered platforms. First, note that Bugzilla

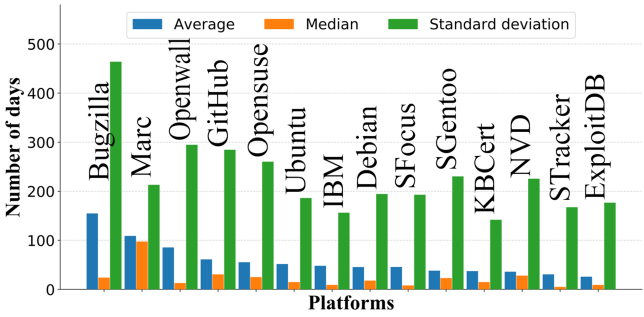

Fig. 7. Disclosure delays between consecutive advisories.

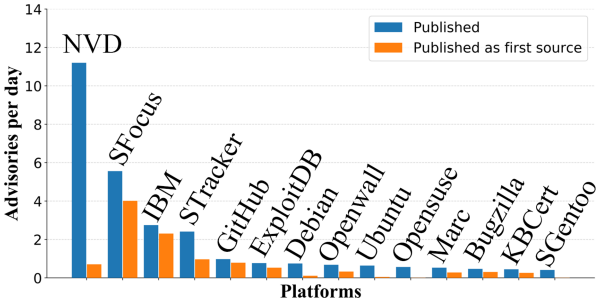

Fig. 8. Rate at which security advisories are published by platforms.

(respectively, Ubuntu) is the platform that provides the most (resp., less) complementary information, concerning timeliness, when contrasted against NVD. Second, note that the standard deviation of the number of days is very large, which might pose challenges for predictive models. Advisories for some vulnerabilities are issued during a long period of time, some of them on the order of years.

Platform Delays. Next, we consider the delays incurred while different security advisories are disclosed at the considered platforms. Figure 7 shows the delays between advisories about a given vulnerability being disclosed in a given platform and the succeeding advisory for the same vulnerability being disclosed by the next platform. The mean delays are much higher than the corresponding medians, indicating that outliers can significantly impact the average. This finding is similar in essence to the observations made on Fig. 6.

Publishing Rates. Figure 8 shows the rate at which security advisories are published by each platform, in units of advisories per day. The corresponding normalized rates, reported in Fig. 9, will be used to validate the model in Sect. 5. As expected, NVD plays a key role in the security advisory ecosystem, publishing and average of 11.2 advisories per day. To account for the order at which information flows across platforms, Fig. 8 also shows the rate at which security

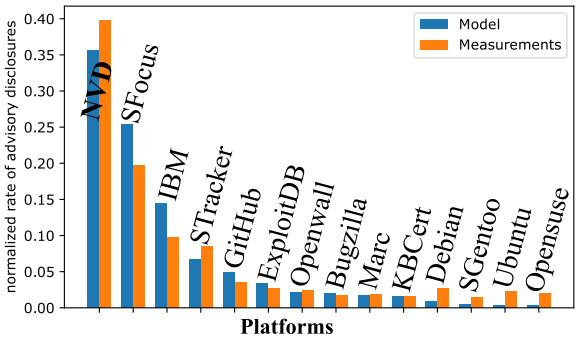

Fig. 9. Normalized rate of security advisory disclosures and model validation.

advisories are first published by each platform. Figure 8, in agreement with the Sankey diagram in Fig. 4, indicates that Security Focus is the platform which most commonly ranks as first to publish security advisories.

4 Analytical Model

Next, we propose and describe an analytical model to assess how local properties of security advisory platforms, such as the probability that an advisory at a platform yields a new advisory at another platform, impact global aspects of the system, such as the mean number of platforms wherein advisories for a given vulnerability are reported. Our model is inspired by the Sankey diagram presented in Fig. 4. While Sankey diagrams are instrumental to provide a visual perspective on the flow of information, the analytical model quantifies the role of each platform by augmenting the Sankey diagrams with temporal information about how many days each platform took to relay advisories to other platforms, as well as probabilistic information about the routing of vulnerabilities across platforms.

Parameters and Metrics of Interest. The model is introduced in Sect. 4.1. Basic output metrics obtained from the model appear in Sect. 4.2. The parameterization of model inputs is described at the end of Sect. 4.2 and the parameterized model is evaluated in Sect. 5.

4.1 Model Description

We model the flow of advisories across platforms through a "network of queues" [6, Chapter 17]. In the simplest setting, each queue corresponds to a security advisory platform. Platforms receive advisories from other platforms in the network or from outside the network of considered platforms, and propagate them after a certain delay. To capture delays, we leverage $M/G/\infty$ queues [6]. We denote by d_i the mean delay incurred between an advisory being published at platform i and at its succeeding platform, measured in days.

In what follows, we refer to an advisory as being relayed or routed from platform i to platform j when platform j publishes an advisory about the vulnerability of interest after i has done so. In particular, after an advisory is relayed from i to j, the advisory at platform j may leverage the advisory published at i. The advisory published by platform j may explicitly refer to previous advisories or may implicitly rely on their content, e.g., to propose a countermeasure.

Key Simplifying Assumption. The key simplifying assumption considered in our model consists of characterizing the flow of information between security platforms inspired by the Sankey diagram in Fig. 4 accounting for two layers while still capturing the fact that information about each CVE may be published by advisories across more than two platforms. Let L be the number of layers in the diagram, with $L = 3$ in Fig. 4. The proposed analytical model requires the estimation of roughly $(L-1)P^2$ parameters corresponding to the routing of advisories between platforms, where P denotes the number of platforms, and the notion of routing is made precise in what follows. As in any modeling exercise, we trade off between simplicity and accuracy, and find that the simplest model with $L = 2$ suffices for our purposes (see Sect. 5). Nonetheless, the model can be easily extended to account for $L > 2$, at the cost of additional complexity.

Routing of Advisories. We consider a set of P security advisory platforms. In addition to those platforms, we introduce a virtual platform, that is always the last platform to publish an advisory about a given vulnerability. The virtual platform, whose sole goal is to simplify model presentation, is denoted as platform ∞, and its index is $P + 1$.

The *routing* of advisories between security advisory platforms is captured through a *routing matrix*. Let R denote the routing matrix. Element r_{ij} denotes the probability that platform i routes an advisory to j, where $\sum_{j=1}^{P+1} r_{ij} = 1$. Note that as the virtual platform $P + 1$ is a sink, $r_{P+1,P+1} = 1$.

The platforms can be grouped into two blocks. The platforms in the first (resp., second) block are transient (resp., absorbing) platforms. Let R_0 (resp., I_u) be the intra-block transition matrix corresponding to elements in the first (resp., second) block, and let R_1 be the inter-block transition matrix. Note that we assume there are u absorbing platforms, and I_u is an $u \times u$ identity matrix. In the simplest case, which constitutes our reference setup, we have a single absorbing platform corresponding to the virtual sink, namely ∞, and $u = 1$. Whenever the dimension of the identity matrix is clear from context, we drop subscript u. Then,

$$R = \begin{pmatrix} R_0 & R_1 \\ \mathbf{0} & I \end{pmatrix}. \tag{1}$$

A published vulnerability is said to be at platform i if platform i was the last platform to publish an advisory about it. Then, the state of the queueing network is given by $(\sigma_1, \ldots, \sigma_P)$ where σ_i is the number of vulnerabilities at platform i. The queueing network state dynamics are given by a continuous time Markov chain, whose infinitesimal generator and steady state probabilities readily follow from the above description and are found in [6].

4.2 Security Advisory Platforms Metrics of Interest

Platform Publishing Rates. Let λ_i be the rate at which advisories for unheard vulnerabilities are first published by platform i, i.e., λ_i is the rate at which platform i discloses advisories whose corresponding CVE disclosure date equals platform i CVE disclosure date. Accordingly, let p_i be the probability that platform i is the first to publish an advisory for a given vulnerability, i.e., p_i is the fraction of vulnerabilities for which advisories are first published by platform i. Then, $\lambda_i = p_i \Lambda$, where Λ is the rate at which new vulnerabilities (and the corresponding security advisories) are reported online.

Let m_{ij} be the mean number of advisories for a given vulnerability, initially published at platform i, to appear at platform j. Matrix M is known as the *fundamental matrix* [6], and is given by

$$M = \sum_{k=0}^{\infty} R_0^k = (I - R_0)^{-1} \tag{2}$$

where I is the identity matrix and R_0 is the routing matrix R with the row and column corresponding to ∞ being removed as described above. In what follows, we indicate how the fundamental matrix yields our metrics of interest.

Let γ_i be the rate at which advisories are published by platform i. Vector $\boldsymbol{\gamma}$ is the vector of publishing rates,

$$\boldsymbol{\gamma} = \boldsymbol{\lambda} M = \boldsymbol{\lambda}(I - R_0)^{-1}. \tag{3}$$

Note that whereas $\boldsymbol{\lambda}$ is the vector of rates at which advisories for unheard vulnerabilities are published by each platform, vector $\boldsymbol{\gamma}$ accounts for all advisories.

Platform Hitting and Absorption Probabilities. Next, we consider the problem of quantifying the relative freshness of advisories published across a set of platforms. To that aim, consider a set of S competing platforms. Let b_{ij} be the probability that a given platform $j \in S$ is the first, among the platforms in S, to publish about a vulnerability originally issued by platform $i \notin S$. To compute b_{ij}, we include all platforms in S into a group of absorbing platforms, together with the virtual platform ∞. Following the terminology introduced in Sect. 4.2, let R_1 be a $(P - |S|) \times (|S| + 1)$ matrix, whose elements characterize the routing probabilities between platforms outside S to platforms in S. Similarly, M is the fundamental matrix accounting for platforms outside S. Then, b_{ij} is given by element (i, j) in matrix B

$$B = M R_1. \tag{4}$$

The derivation of the above equation can be found in [6]. In Sect. 5 we evaluate the above expression to assess how early four of the most popular platforms publish their advisories, when contrasted against the others.

Mean Time Between Platforms. Next, we consider the mean time between platforms, in days. Let column vector $\boldsymbol{t}^{(i)}$ characterize the mean time for a

vulnerability to be forwarded by each platform, except i, i.e., $\boldsymbol{t}^{(i)}$ equals $\boldsymbol{d}$, after excluding the i-th element from the latter. Then, the mean time to reach either platform i or the virtual platform ∞, whichever occurs first, is given by

$$\boldsymbol{T}^{(i)} = \left(I - R_0^{(i)}\right)^{-1} \boldsymbol{t}^{(i)} \tag{5}$$

where $R_0^{(i)}$ is obtained from R_0 after removing the column and row corresponding to platform i. Note that if $\boldsymbol{t}^{(i)}$ is a column vector with all its elements equal to 1, the resulting vector obtained from the above equation is the mean number of advisories issued until reaching either platform i or the virtual platform ∞, whichever occurs first.

To obtain matrix T, whose element $t_{i,j}$ corresponds to the mean time to reach platforms j or ∞, given that an advisory is first published at i, it suffices to concatenate column vectors $\tilde{\boldsymbol{T}}^{(i)}$, $i = 1, \ldots, P$, where $\tilde{\boldsymbol{T}}^{(i)}$ is obtained from $\boldsymbol{T}^{(i)}$ adding an entry equal to zero at the i-th position, to capture the fact that the mean time to reach platform i starting from i is zero [6]. In Sect. 5 we evaluate the above expression to assess the time it takes for advisories to flow across platforms.

How to Parametrize Model from Data. The model is parametrized through three sets of parameters: the arrival rates of advisories for new vulnerabilities from outside the considered network of platforms, the delays and the routing matrix.

Let n_i be the number of advisories published by platform i, as observed in the measurements. Similarly, let f_i be the number of advisories published by i, when no other advisory for the corresponding vulnerability has been issued, $f_i \leq n_i$. Also, let τ be the measurement duration, i.e., the time between the publication of the first and last advisories present in the dataset. We denote by $\hat{\lambda}_i$ and $\hat{p}_i$ the estimators of λ_i and p_i, respectively. Then, $\hat{\lambda}_i = f_i/\tau$ and $\hat{p}_i = f_i \Big/ \sum_{k=1}^{P} f_k$. We also have $\hat{\Lambda} = \sum_{k=1}^{P} \hat{\lambda}_k$. To parametrize the delays and the routing matrix, we begin by generating the multiset $\mathcal{P}$ of ordered pairs of platforms, as described in Sect. 3. Each ordered pair $(i, j) \in \mathcal{P}$ corresponds to an instance wherein platform j published an advisory about a given vulnerability after an advisory for the same vulnerability appeared at i. Let $\tilde{r}_{ij}$ be the number of times that the ordered pair (i, j) appears at $\mathcal{P}$. Then, denoting by $\hat{r}_{ij}$ the estimator of r_{ij},

$$\hat{r}_{ij} = \tilde{r}_{ij} \Big/ \sum_{k=1}^{P+1} \tilde{r}_{ik}. \tag{6}$$

To determine $\hat{d}_i$, the estimator of d_i, we also rely on $\mathcal{P}$. To each element $(i, j) \in \mathcal{P}$ we associate its corresponding delay in days, d, and denote the resulting triple by (i, j, d). The mean delay estimate $\hat{d}_i$ is given by the average of the third component of the triples wherein the first component equals i and the second component is different from ∞.

Model Extensions. The proposed model can be easily extended to account for additional layers, i.e., $L > 2$, providing more accurate estimates of the quantities of interest at the expense of a more complex model, involving more parameters to be estimated. Consider, for instance, the Sankey diagram presented in Fig. 4, wherein $L = 3$. To account for $L = 3$, we associate to each node in the first $L - 1$ layers in Fig. 4 its corresponding queue. The resulting queueing network comprises $(L - 1)P^2$ routing probabilities across queues.

5 Evaluation

Next, we evaluate our model leveraging data collected from our measurement campaign. Our goals are to (a) validate the theory using real data, indicating that the estimated metrics of interest are accurate even when the assumptions of the model do not hold and (b) numerically analyze the metrics derived from the model.

The model is parameterized using the methodology described in Sect. 4.2. The routing matrix estimator, $\hat{R}$, is given by (6), the vector of delays, $\hat{\boldsymbol{d}}$, is obtained from Fig. 7, and the vector of disclosure rates of advisories for unheard CVEs, $\hat{\boldsymbol{\lambda}}$, is obtained from Fig. 8 ("published as first source" bars).

Model Validation. Next, we validate our model against measurements. In particular, we aim at verifying if the key simplifying assumption discussed in Sect. 4.1, according to which the flow of advisories across platforms can be approximated through a two layer model, with $L = 2$, already leads to accurate estimates.

To validate our model against measurements, we compare the normalized rate at which platforms disclose security advisories obtained from the model against measurements. We choose the normalized rate as our reference metric for validation purposes as most of the other quantities can be derived from it. Similar results hold for those other metrics (omitted due to space constraints).

According to the model (resp., measurements), the normalized disclosure rate is obtained from (3) (resp., Fig. 8), normalizing the resulting vector so that the sum of rates equals one. Figure 9 indicates that our model estimates are accurate when compared against measurements, with the largest absolute error being less than 0.05. If smaller errors are required, one can trade off between model complexity and accuracy, and parametrize an extended version of the proposed model (see Sect. 4.2).

Assessing Distances Between Platforms. From the routing matrix, we compute our metrics of interest. Figure 10 shows the mean time between platforms, measured in days. It corresponds to matrix T, whose derivation is described in Sect. 4.2 (see Eq. (5)). Each entry (i, j) in the matrix in Fig. 10 corresponds to the mean number of days to route an advisory from platform i to either platform j or to the virtual platform ∞, whatever occurs first. Figure 10 shows that all vulnerabilities take, on average, more than one month to reach a target.

Figure 10 shows that from some platforms, such as Bugzilla and Openwall, it takes a long time to reach all other platforms. This can be explained from

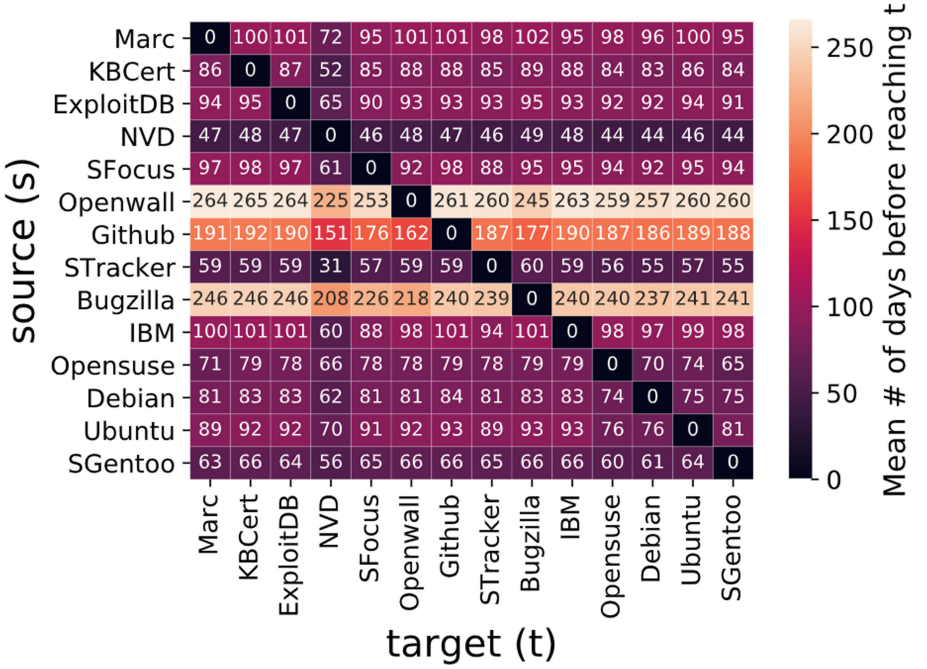

Fig. 10. Mean time between platforms in days.

Figs. 6 and 7. Figure 6 shows that Bugzilla and Openwall are the top two platforms ranked based on the delay until reaching NVD. As NVD is an important hub in the system to relay advisories across platforms, the advisories published on those two platforms tend to take longer to reach any other platform. Recall from Sect. 4.2 that we parametrize delays through the number of days between succeeding advisories are published at consecutive platforms, which for measurements is given by Fig. 7. Bugzilla and Openwall appear as the first and third elements in Fig. 7. This, in turn, indicates that the mean number of days to route an advisory from one of those two platforms to the virtual platform ∞ is the largest among all platforms.

Figure 11 shows the percentage of vulnerabilities for which an advisory first published at platform s will eventually reach platform t, before reaching the virtual absorbing platform ∞ or any other platform in the considered set of targets, comprising Security Tracker, Security Focus and NVD. The figure is obtained using Eq. (4) and the methodology described in Sect. 4.2, and complements Fig. 10. In particular, it indicates that with high probability the advisories initially posted in most of the platforms will reach NVD before absorption. Returning to the Bugzilla and Openwall platforms, this observation, together with the fact that the time to reach NVD from those platforms is large, explains why vulnerabilities initially generated at those platforms take longer to reach either one of the other platforms in the system or be absorbed.

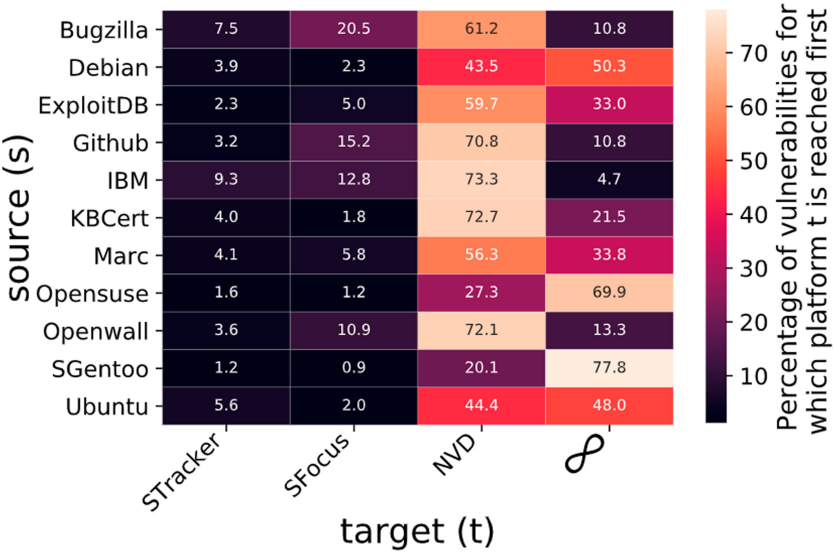

Fig. 11. Percentage of vulnerabilities issued from s for which t is reached first, considering Security Tracker, Security Focus and NVD as targets.

6 Conclusion

In the realm of security, information is the oxygen and security advisory platforms, the trees. Understanding how information flows across platforms is a key step in the full characterization of vulnerability lifecycles. After conducting a large scale measurement campaign, we use those measurements to parametrize a queueing model which allows us to estimate the impact of different parameters, such as the rate at which platforms relay information, on the mean time that vulnerability advisories take to reach a given platform.

Understanding the flow of information across platforms is instrumental for real world operations, such as data-driven risk aware patch management [19] and cybersecurity insurance [4,14,20]. In both cases, the confidence level on the estimates of risk is a function of the quality and the number of security advisories. While assessing how CVSS and EPSS evolve over time, the number of issued advisories serves to parameterize confidence intervals either in retrospect using measurements (Sect. 3) or in hindsight using the proposed model (Sect. 4).

Admins increasingly rely on automated data-driven approaches for decision-making, being utterly bound by the quality and freshness of the ingested information. This work is a step towards characterizing the latter, opening up a number of interesting avenues for research, accounting for advisories maturity and the modeling of topics and their evolution across security advisory platforms [1,5].

Acknowledgments. This project was partially sponsored by CAPES, CNPq and FAPERJ, through grants E-26/203.215/2017 and E-26/211.144/2019, as well as scholarships from Siemens Corporate Research.

References

1. de Boer, M.H., Bakker, B.J., et al.: Text mining in cybersecurity. Multimodal Technol. Interact. **3**(3), 62 (2019)
2. Decan, A., Mens, T., Constantinou, E.: On the impact of security vulnerabilities in the NPM package dependency network. In: Proceedings of the 15th International Conference on Mining Software Repositories, pp. 181–191 (2018)
3. Frei, S., May, M., Fiedler, U., Plattner, B.: Large-scale vulnerability analysis. In: SIGCOMM Workshop on Large-Scale Attack Defense, pp. 131–138 (2006)
4. Gai, K., et al.: A novel secure big data cyber incident analytics framework for cybersecurity insurance. In: Big Data Security on Cloud, pp. 171–176. IEEE (2016)
5. Georgescu, T.M.: Natural language processing model for automatic analysis of cybersecurity-related documents. Symmetry **12**(3), 354 (2020)
6. Harchol-Balter, M.: Performance Modeling and Design of Computer Systems: Queueing Theory in Action. Cambridge University Press, Cambridge (2013)
7. Horawalavithana, S., Bhattacharjee, A., et al.: Mentions of security vulnerabilities on Reddit, Twitter and Github. In: International Conference on Web Intelligence, pp. 200–207 (2019)
8. Hu, W., Wang, Y., Liu, X., Sun, J., Gao, Q., Huang, Y.: Open source software vulnerability propagation analysis algorithm based on knowledge graph. In: IEEE International Conference on Smart Cloud, pp. 121–127. IEEE (2019)
9. Huang, S., Tang, H., et al.: Text clustering on national vulnerability database. In: Computer Engineering and Applications, vol. 2, pp. 295–299. IEEE (2010)
10. Joh, H., Malaiya, Y.K.: A framework for software security risk evaluation using the vulnerability lifecycle and CVSS metrics. In: Proceedings of International Workshop on Risk and Trust in Extended Enterprises, pp. 430–434 (2010)
11. Johnson, P., Gorton, D., Lagerström, R., Ekstedt, M.: Time between vulnerability disclosures. Comput. Secur. **62**, 278–295 (2016)
12. Li, V.G., Dunn, M., Pearce, P., et al.: Reading the tea leaves: a comparative analysis of threat intelligence. In: USENIX Security 2019, pp. 851–867 (2019)
13. MITRE: Common vulnerabilities and exposures (2020). https://cve.mitre.org/
14. Rassam, M.A., Maarof, M., Zainal, A., et al.: Big data analytics adoption for cybersecurity. J. Inf. Assur. Secur. **12**(4) (2017)
15. Rosen, C., Shihab, E.: What are mobile developers asking about? A large scale study using stack overflow. Empirical Softw. Eng. **21**(3), 1192–1223 (2016)
16. Ruohonen, J.: A look at the time delays in CVSS vulnerability scoring. Appl. Comput. Inf. **15**(2), 129–135 (2019)
17. Ruohonen, J., Hyrynsalmi, S., Leppänen, V.: Modeling the delivery of security advisories and CVEs. Comput. Sci. Inf. Syst. **14**(2), 537–555 (2017)
18. Shahzad, M., Shafiq, M.Z., Liu, A.X.: A large scale exploratory analysis of software vulnerability life cycles. In: International Conference on Software Engineering, pp. 771–781 (2012)
19. Wang, B., Li, X., de Aguiar, L.P., Menasche, D.S., Shafiq, Z.: Characterizing and modeling patching practices of industrial control systems. Proc. ACM Meas. Anal. Comput. Syst. **1**(1), 1–23 (2017)
20. Woods, D., Moore, T.: Does insurance have a future in governing cybersecurity? IEEE Secur. Privacy Mag. **18**(1), 21–27 (2019)
21. Zhang, S., Ou, X., Caragea, D.: Predicting cyber risks through national vulnerability database. Inf. Secur. J. **24**(4–6), 194–206 (2015)