The Effect of Common Vulnerability Scoring System Metrics on Vulnerability Exploit Delay

Andrew Feutrill*‡, Dinesha Ranathunga*, Yuval Yarom†, Matthew Roughan*
*ARC Centre of Excellence for Mathematical and Statistical Frontiers, School of Mathematical Sciences, University of Adelaide
{dinesha.ranathunga, matthew.roughan}@adelaide.edu.au
†School of Computer Science, University of Adelaide
yval@cs.adelaide.edu.au
‡Data to Decisions CRC
andrew.feutrill@d2dcrc.com.au

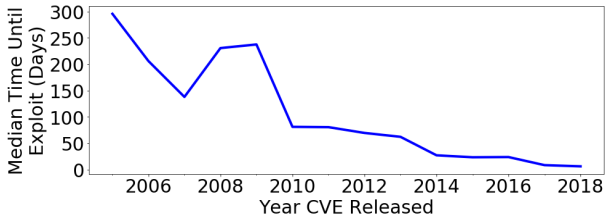
***Abstract*—Modern system administrators need to monitor disclosed software vulnerabilities and address applicable vulnerabilities via patching, reconfiguration and other measures. In 2017, over 14,000 new vulnerabilities were disclosed, so, a key question for administrators is which vulnerabilities to prioritise. The Common Vulnerability Scoring System (CVSS) is often used to decide which vulnerabilities pose the greatest risk and hence inform patching policy. A CVSS score is indicative of a vulnerability severity, but it doesn't predict the time to exploit for a vulnerability. A prediction of exploit delay would greatly assist vendors in prioritising their patch releases and system administrators in prioritising the installation of these patches.**

In this paper, we study the effect of CVSS metrics on the time until a proof of concept exploit is developed. We use the National Vulnerability Database (NVD) and the Exploit Database, which represent two of the largest listings of vulnerabilities and exploit data, to show how CVSS metrics can provide better insight into exploit delay. We also investigate the time lag associated with populating CVSS metrics and find that the median delay has increased rapidly from a single day prior to 2017 to 19 days in 2018. This is an alarming trend, given the rapid decline in median vulnerability exploit time from 296 days in 2005 to six days in 2018.

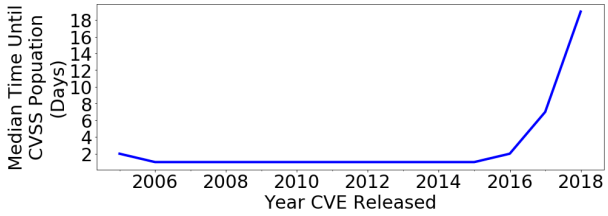
I. Introduction

In 2017 there were 14,714 cyber security vulnerabilities publicly disclosed as Common Vulnerabilities and Exposures (CVE) entries [11]. This volume of vulnerabilities can hinder information-security professionals from understanding the risk they pose to their organisations. Risk scoring mechanisms such as the Common Vulnerability Scoring System (CVSS) have been developed to improve understanding. CVSS assigns a severity score to each disclosed CVE entry. A CVSS score is often used by organisations to determine which vulnerabilities pose the greatest risk and hence inform patching policy. For example, in the Payment Card Industry Data Security Standard (PCI-DSS) v3.2.1 any vulnerability with a base score of higher than 4.0 results in instant non-compliance [8].

The CVSS score is calculated by combining various vulnerability characteristics which are referred to as CVSS metrics [10]. When a CVE entry (or a CVE for brevity) is publicly disclosed, its CVSS metrics are not populated until the analysis of the proposed vulnerability is completed by the National Institute of Standards and Technology (NIST). Until this occurs, the vulnerability details page in the National Vulnerability Database (NVD) [17] has the status "Awaiting Analysis".



(a) Median Time until Exploit by Year of CVE disclosure



(b) Median CVSS Metric Population delay by Year of CVE disclosure

Fig. 1: *Comparison of median Time until Exploit and CVSS Metric Population delay.*

A CVSS score is indicative of vulnerability severity but does not help predict exploit delay. [1, 4]. Better insight into time-to-exploit for a vulnerability could greatly assist vendors when prioritising their patch releases and systems administrators when prioritising their installation of these patches.

Past works have investigated how well a CVSS score describes real world vulnerability exploits [1], and how time to exploit can be modelled using probability models [12]. But, these works have not investigated the delay in obtaining CVSS metrics or whether CVSS metrics can help predict exploit delay more usefully. We ask the question

Can we predict the time until a proof of concept exploit is developed based on the CVSS metrics?

However, the metrics are not populated until a vulnerability is analysed, so a second question is

Are CVSS metrics populated in time to be used meaningfully for exploit delay prediction of CVEs?

In this paper we answer the first question in the positive and the second in the negative. We begin by studying the CVSS metric population delay. Our analysis finds that the median time it takes to populate CVSS metrics for a CVE has increased rapidly from a single day prior to 2017 to 19 days in 2018 (Figure 1(b)). This is an alarming trend, given the rapid decline in median vulnerability exploit time from 296 days in 2005 to six days in 2018 (Figure 1(a)).

We use statistical methods to determine how Time to Exploit is affected by conditioning on CVSS metrics, to understand the lead time assuming CVSS metrics were available. In particular, we show how certain combinations of CVSS metrics can provide better insight into exploit delay. For instance, we see that the median exploit delay for CVEs with User Interaction (UI) = Required and Attack Complexity (AC) = Low is seven days. This figure is more useful in predicting exploit delay than the general figure of 47 days we obtain when delay is not classified by CVSS metrics.

However, our research shows that, if continued, the current trend in median CVSS metric population delay will hinder their meaningful use in predicting the time to exploit for a CVE. Our study of CVE disclosures over time found that there is a noticeable increase in the number of disclosed CVEs from 2017 onwards. This increase could be leading to a significant backlog of CVEs. Thus, perhaps more resources need to be allocated to this task.

II. CVE History

The Common Vulnerabilities and Exposures Database (CVEDB) is a collection of cyber security vulnerabilities. Each vulnerability entry in this database is assigned a unique identifier to aid communication between parties and ensure there are no gaps in security coverage [25].

The CVE Database has been adopted as the industry standard for identifying vulnerabilities and exposures and has achieved wide acceptance by the security industry and a number of government organisations. Databases, such as the NVD, rely on the CVE Database to identify vulnerabilities and then supplement the identifier with information such as vendor reports, products and versions affected and vulnerability type.

The CVE Database was publicly launched by MITRE in September 1999 [25]. It was initialised with 321 entries from historical vulnerabilities collected from multiple databases. Over the next few years, vendors gradually became participants. This was assisted by NIST recommending the usage of the CVE Database in 2002 [13], the requirement of CVE Database compliance by the United States Defence Information Systems Agency (DISA) in 2004 [25], and adoption by the NVD in 2005 [17]. Figure 2 shows the number of CVE entries disclosed per month. The initiation of MITRE's CVE database in 1999 can be seen, with low numbers of CVE entries disclosed, as well as the increase in rate of disclosures in the shaded region during 2005. From the establishment of the NVD Database in August 2005 the process was approximately stationary until 2017.

Figure 2 also shows that from the beginning of 2017, the rate of CVE disclosures has sharply increased. Chris Coffin from MITRE's Common Vulnerabilities and Exposures Team informed us that this increase was due to the following changes:

- an increase in the number of Common Vulnerabilities and Exposures Numbering Authorities (CNAs), which are independent organisations who are authorised to assign CVE identifiers;
- simplification of the request and submission procedures for a CVE;
- changes to the counting rules of CVEs; and
- new technologies utilising the CVE Database.

Hence, we analyse the disclosure process in three distinct eras. As shown in Figure 2, the first era represents the establishment of the process, from the launch in 1999 until end of July 2005. The second era covers the establishment of NVD in August 2005 until the end of 2016. The third era begins at the beginning of 2017, when the number of CVEs jumped drastically. Our study focuses on the second and third eras as the first era was prior to MITRE's CVE database being an industry standard. Hence, the CVE arrivals in this era are highly variable and not indicative of the rest of the process.

Software vendors have proposed many proprietary schemes for scoring software vulnerabilities [16, 23, 26], each with its own merits. For instance, CERT/CC considers factors such as whether the Internet infrastructure is at risk [26]. Microsoft employs a scoring system which reflects the exploitation difficulty and overall impact of a vulnerability [16]. A key drawback with these scoring systems is that they assume that the vulnerability impact is constant across every individual and organisation. Moreover, these schemes do not provide visibility into how a vulnerability score is calculated.

The Common Vulnerability Scoring System (CVSS) was developed (and is currently maintained) by the CVSS Special Interest Group (CVSS-SIG) to address these shortfalls. CVSS standardises the information used by organisations to prioritise vulnerability mitigation [14]. CVSS is the only open specification that is also designed to be quantitative, removing the need for qualitative evaluation of vulnerability severity. The specification aims to allow multiple vulnerability analysts to produce identical CVSS scores for the same vulnerability. Most importantly, CVSS provides visibility into how a vulnerability score was computed.

The CVSS specification has been widely adopted by vendors, service providers, vulnerability tools and bulletins. For instance, software and hardware manufacturers such as IBM, Cisco and HP use it as a reporting metric. CVSS is also mandated for use in evaluating the security of payment card systems globally. The U.S. Federal government also uses it in the National Vulnerability Database (NVD) [17] which is the main repository of known vulnerabilities worldwide.

There are other publicly available sources which provide information on vulnerability exploits. The Exploit Database (*i.e.,* ExploitDB) is an archive of public exploits and corresponding vulnerable software, developed for use by penetration testers

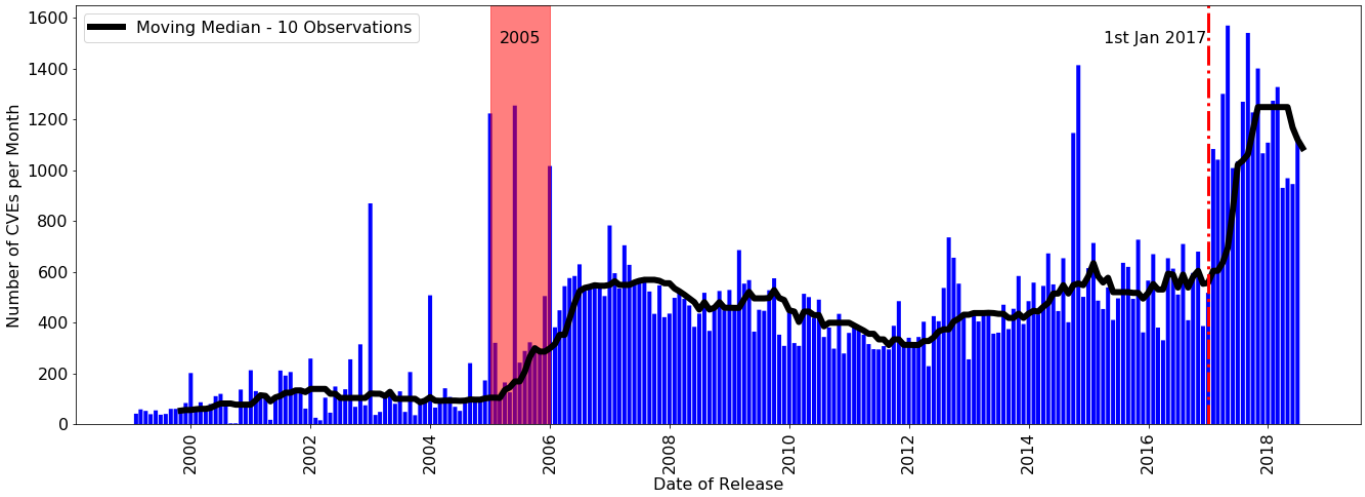


Fig. 2: *Number of CVE Entries Disclosed per Month from 1999 to 2018. The CVE arrivals can be split into three distinct eras: (1) from 1999 to 2005, (2) from 2005 to 2017 and (3) from 2017 to date.*

and vulnerability researchers [18]. This began as a public exploit archive in 2004 by a hacker named 'str0ke' and was taken over by Offensive Security in November 2009 [18]. ExploitDB contains nearly 40,000 exploits, with the earliest exploit date being 1st August 1988. The information about an exploit includes an assigned ExploitDB ID, a CVE identifier (if exists), author name, exploit type, platform and publication date.

Our study uses the NVD and ExploitDB as the primary data sources because they represent two of the largest and most reliable sources of vulnerability and exploit data.

III. Related Work

Statistical analysis of the vulnerability life cycle has been performed by Frei [12] and Allodi and Massacci [1, 2]. Frei produced a comprehensive analysis of the vulnerability life cycle, and in particular studied the dynamics of this process. He defined a life cycle composed of six stages and empirically modelled probability distributions of exploit availability and patch availability post vulnerability disclosure. Frei also showed that while 94% percent of exploited vulnerabilities had an exploit available within 30 days, only 72% of patches were available in this time-frame. Our research extends this work by classifying CVE exploit delay by CVSS metrics to yield better delay predictions.

Allodi and Massacci have used descriptive statistics to analyse how effective vulnerability databases are in estimating the risk of real world cyber attacks [1, 2]. They also used the NVD and the Exploit Database as a basis for their analysis. In [1] they developed another database — EKITS — which contains information about the exploits that are contained in known exploit kits. Their conclusion is that CVSS scores don't allow one to effectively discriminate between the likelihood of exploitation and non-exploitation. In contrast, our work focuses on the time for an exploit to become publicly available for a CVE, following its disclosure. Better understanding of this delay allows vendors to prioritise the patches they release and system admins to prioritise the installation of these patches and we show that improved predictions are possible.

Several authors have also studied the use of CVSS metrics in machine learning algorithms to predict whether vulnerabilities will be exploited and the length of time until exploit [4, 5, 9, 22]. All of these works have used the CVSS scores as features in machine learning algorithms. Bozorgi et al. [4] concluded that the CVSS scores did not have a noticeable impact on the prediction results. Edkrantz and Said [9] concluded that the CVSS metrics are redundant when a large number of text features are used. However, early exploit detectors built using Twitter information in [22] had better performance when supplemented with CVSS metrics.

Rajasooriya et al. [20] have also proposed an approach using stochastic modelling to generate predictions of exploit development which incorporate CVSS scores. Their approach uses Markov Chains with states representing the stages of the vulnerability life cycle. They developed three different models for the behaviour of vulnerabilities moving through the life cycle which are based on whether their base CVSS score is in the High (7.0-10), Medium (4.0-6.9) or Low (0.0-3.9) range. This revealed that different vulnerability dynamics occur at different risk levels and that High risk vulnerabilities have a higher probability of being exploited. Our work differs from this approach because we study exploit delay, not exploit probability and we also consider the effect of the various CVSS metrics not just the CVSS score. Doing so, allows us to identify more granular relationships between CVSS metrics and exploit delay.

Next, we describe the data sources used in our analysis and how we extracted the data.

IV. CVE and Exploit Data

We extracted ExploitDB data via web scraping using the Python library Scrapy [24]. Our web scraper iterates through

all of the exploits which are hosted on ExploitDB, then if a CVE identifier exists, it extracts the exploit name, link to exploit, date of release, author name and platform effected. We collected 28,088 exploits dated between 1st August 1988 and 9th July 2018 from ExploitDB with a CVE identifier. Of these exploits only 22,048 had unique CVE identifiers because some exploits applied to the same CVE. These form the gold standard records for when a vulnerability is considered to have been exploited.

To obtain the CVE and CVSS metrics data, we used the JSON data feed from NVD. An XML feed is also provided, however we found that the JSON feed is more compact, readable and easier to process. We extracted 106,764 unique CVE identifiers from these JSON files. The date when the CVE was analysed and the CVSS scores were added to the records was obtained by examining the *Change History* of the NVD page for the CVE.

Each CVE has information including a description of the vulnerability, CVSS scores, references to external pages related to the issue, vulnerability type and affected products. Using a JSON parser in Python, CVSS v2 and v3 information and date of disclosure were extracted. The CVSS base metrics that are contained in the JSON files and their possible values are shown in Table I. We discuss these metrics in detail in § V.

We merged the NVD and ExploitDB data, matching on the CVE identifiers, and then dropped all of the records for CVE identifiers that were not unique and had an earlier exploit released, to construct a single reference dataset.

We describe the CVSS framework in detail next, to provide an understanding of the CVSS metrics, prior to ascertaining their effect on the Time to Exploit for a CVE.

V. CVSS FRAMEWORK

The CVSS vulnerability score is a decimal number in the range 0.0–10.0. This score is calculated using three metric groups; base metrics, temporal metrics and environmental metrics. Base metrics represent vulnerability attributes which remain constant over time and user environments. Temporal metrics represent vulnerability attributes which change with time but not between user environments (*e.g.,* due to changes in publicly available exploit code or a remediation technique). Environmental metrics represent vulnerability attributes which are implementation specific; *e.g.,* how prevalent a target is within an organisation. The temporal score of a vulnerability is calculated using the base score and the temporal metric values as parameters. Likewise, the environmental score is calculated using the temporal score and the environmental metrics values as parameters.

We filtered the data extracted from NVD to obtain CVSS metrics for 94,417 CVEs originating from August 2005 to June 2018. This is a reduction from the entire set of 106,764 CVEs as we exclude those that were released prior to era two in the data. These metrics cover CVSS v2 [15] and v3 [10]. We present a comparison, below, of the two versions.

TABLE I: *CVSS v3 Base Metrics used in our analysis. The exploitability metrics capture the difficulty in exploiting a vulnerability. The impact metrics capture the consequence of a vulnerability if successfully exploited.*

Metric Type	Metric	Possible Values
Exploitability Metrics	Attack Vector	Network Adjacent Local Physical
	Attack Complexity	Low High
	Privileges Required	None Low High
	User Interaction	None Required
Scope	Scope	Unchanged Changed
Impact Metrics	Confidentiality Impact	High Low None
	Integrity Impact	High Low None
	Availability Impact	High Low None

A. Converting CVSS v2 Metrics to CVSS v3

CVSS v1 – the first version of CVSS – was introduced in February 2005. This was superseded by CVSS v2 in June 2007. We need not convert CVSS v1 metrics to v3 because all vulnerabilities which contained v1 metrics have already been updated to v2 [11].

The CVSS v2 and CVSS v3 metrics are not compatible. Hence, we needed to define a mapping from CVSS v2 to CVSS v3 to be able to compare the attributes of CVEs. In CVSS v2 [15] there are six base metrics grouped into exploitability and impact metrics. The exploitability metrics consist of Access Vector (AV), Access Complexity (AC), and Authentication (AU). These capture the difficulty in exploiting a vulnerability. The impact metrics consist of Availability Impact (AI), Confidentiality Impact (CI), and Integrity Impact (II). These capture the consequence of a vulnerability if successfully exploited.

The exploitability and impact metrics allow one to compute subscores which in-turn can be used to compute a base CVSS score. This base CVSS score quantifies the overall severity of a vulnerability.

CVSS v3 [10] was introduced in June 2015 with several key changes to the scoring system to more accurately reflect vulnerabilities within the Web application domain. In this version, the three metric groups, the base score, the temporal score and the environmental score remain the same. But several new metrics such as Scope (S) and User Interaction (UI) have been added within the groups (see Table I). Several older metrics such as Authentication (AU) have also been modified to a newer metric, *e.g.,* Privileges Required (PR).

TABLE II: *Mapping of factors that changed from CVSS v2 to CVSS v3.* Scope *and* User Interaction *are new metrics introduced in v3. Privileges Required is named Authentication in v2.*

Category	CVSS v2	CVSS v3
User Interaction (UI)	Access Complexity - Low Access Complexity - High	None Required
Scope (S)	Confidentiality Impact - High Integrity Impact - High Availability Impact - High Otherwise	Changed Unchanged
Privileges Required (PR)	Single Multiple None	Low High None
Attack Vector (AV)	Adjacent Network Network Local	Adjacent Network Local
Attack Complexity (AC)	Access Complexity - Low Otherwise	Low High
Confidentiality Impact (CI)	Partial Complete None	Low High None
Availability Impact (AI)	Partial Complete None	Low High None
Integrity Impact (II)	Partial Complete None	Low High None

We use the more refined CVSS v3 metrics throughout our analysis. CVSS v2 metrics (corresponding to CVE entries prior to 2015) were converted to v3 metrics using Table II. We constructed this table considering CVSS v2 and v3 specifications. Most base vectors are directly transformable (*e.g.,* AU = Single in v2 is PR = Low in v3) , but complexities do arise regarding other vectors. For instance, Access Complexity in v2 is separated into Attack Complexity and User Interaction in v3. Also in v2, CI, II and AI collectively map to Scope in v3 (*i.e.,* in most cases CI = High, II = High and AI = High indicate Scope = Changed).

B. CVSS population delay

CVSS metrics provide valuable insight into the characteristics of a vulnerability and can potentially relate to the delay associated with exploiting the vulnerability. This ability to better understand CVE exploit delay when conditioned by CVSS metrics, makes the CVSS population delay an important consideration. The NVD consists of several vulnerability data feeds, each containing the CVE disclosure date t_{disc} but not the CVSS metric population date $t_{cvss\text{-}population}$. CVSS metric population is the result of analysis of CVEs by NIST by aggregating data points from the description, references supplied and any supplemental data that can be found publicly at the time. This analysis results in the assignment of values to the vulnerability attributes of base, temporal and environmental metrics.

The CVSS metric population date is not part of the record itself but it's captured in the *Change History* field of each CVE's web page. In particular, when CVSS (v2 or v3) metric data is populated by NIST, a record is added to the *Change History* field stating the fact and the datetime it was added. We parse such records to extract the CVSS metric population date for each CVE. We scraped this information for CVEs from August 2005 to June 2018 in the NVD (*i.e.,* phase two of the data). The CVSS metric population delay was calculated using the two date time stamps above as

$$t_{cvss\text{-}delay} = t_{cvss\text{-}population} - t_{disc}. \tag{1}$$

Simple analysis of the population delay data shows that it should be modelled using a heavy-tailed probability distribution. This is due to the large number of extreme inter-event times that have occurred over the life of the process. An important class of heavy-tailed distributions is the power-law distribution which has the property that the probability density function $p(x)$ is proportional to a power of the delay x, *i.e.,*

$$p(x) \propto x^{-\alpha}. \tag{2}$$

Power-law distributions have emerged in a variety of fields in recent years. They have been used extensively in internet traffic modelling [3, 7, 21]. We used a power-law distributed random variable with the addition of an exponential cutoff, to model the observed truncation of the probability distribution. At high values of x the exponential function will decay to zero faster than the power law which results in lower probability of extreme events occuring. The probability density function of a power-law distributed random variable with exponential cutoff is given by,

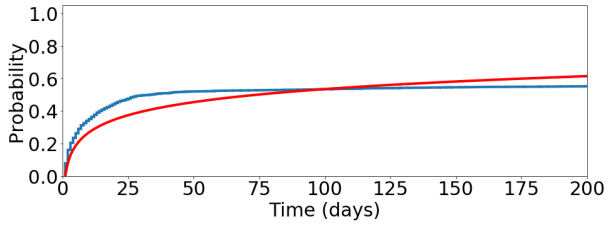
$$p(x) \propto x^{-\alpha} e^{-\beta x}. \tag{3}$$

The blue line in Figure 3(a) shows the Empirical Cumulative Distribution Function (ECDF) plotted for the distribution of $t_{cvss\text{-}delay}$ for data since 2005, and in Figure 3(b) shows the ECDF for $t_{time\text{-}to\text{-}exp}$ since 2005. The Powerlaw library in Python was used to fit a power-law distribution with exponential cutoff. The red line describes the fitted distribution. The Kolmogorov Smirnoff (KS) statistic was used to understand the goodness of fit for the distribution. The KS statistic measures the maximum difference between the ECDF and the fitted Cumulative Distribution Function (CDF) for all values of the functions. The value of the KS statistic for the fitted distribution is 0.16. As shown in Figures 4(a) and 4(b), the model fits reasonably throughout the body and drops off at the end as it approaches 4000 days, which is nearly 11 years and is at the extent of our observation period, which explains the abrupt truncation in extreme values.

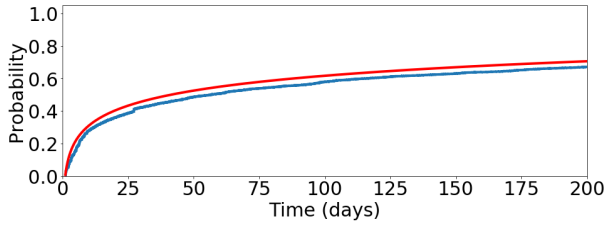
VI. Exploit Delay

In this section we analyse the historical CVE data and determine the impact of the CVSS metrics on the distributions of time until a proof of concept exploit is developed.

Our first step is to merge the data from ExploitDB and NVD, matching on the CVE identifier. This provides information of the day of CVE disclosure and the day of proof

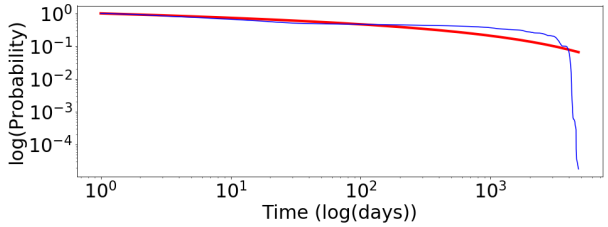


(a) CVSS Metric Population delay

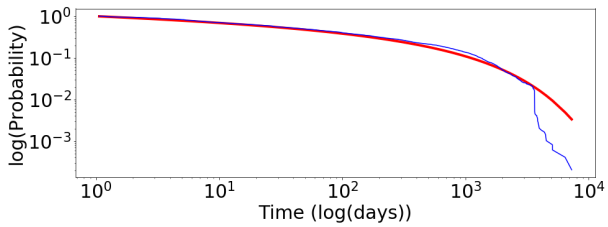


(b) Time to Exploit

Fig. 3: *Empirical Cumulative Distribution Function (ECDF) of CVSS metric population delay and Time to Exploit. Actual distributions are shown by the blue lines while the fitted distributions are shown by the red lines.*



(a) CVSS Metric Population delay



(b) Time to Exploit

Fig. 4: *Complementary Cumulative Distribution Function of CVSS Metric Population delay and Time to Exploit on Log-Log scale. Actual distributions are shown by the blue lines while the fitted distributions are shown by the red lines.*

of concept release for all vulnerabilities that have had an exploit developed. There were 22,048 unique disclosed CVE vulnerabilities that had a proof of concept exploit developed. Some exploited vulnerabilities had multiple proof of concept exploits developed, in this case the earliest date of proof of concept development was used.

From this data, we calculated the time until exploit from the time of disclosure t_{disc} and time of proof of concept release t_{poc} as

$$t_{exp} = t_{poc} - t_{disc}. \tag{4}$$

The ECDF was plotted for the distribution of t_{exp}. For this study only the data with positive time to exploit was considered. This was to replicate the situation of a CVE being disclosed and not having an exploit available at time of disclosure. These are the threats that have been disclosed, without a proposed exploit developed and hence unlikely to have had a patch developed to mitigate the threat due to the vulnerability. Obtaining as much information as early as possible for these types of vulnerabilities is crucial when trying to allocate resources to protect against these types of threats. The exploits that were not considered represent either Zero-Day exploits, those that were exploited before a vulnerability disclosure occurred, or exploits that were released on the same day as disclosure, usually through collaboration between the security researchers that discovered the vulnerability and the vendor, which is a positive outcome for vulnerability security. These types of exploits accounted for 80.4% of the total exploits.

We fit power-law distributions with exponential cutoff to the ECDFs of the t_{exp} data, these are shown by the red lines in Figure 3. The median and 80th percentile values from the fitted distribution are shown in Table III. These show that there

TABLE III: *Fitted distributions of Time to PoC Exploit. The median and the tail of the all-time distribution are larger than those of the time restricted ranges.*

Time Range	Median (Days)	80th Percentile	KS Statistic
All Time	47	431	0.05
Since 2010	25	156	0.08
Since 2017	8	48	0.08

are many extreme values in the dataset, which are an order of magnitude higher than the median. Hence, a power law distribution is a suitable model for this process.

A. Conditioning on CVSS Metrics

We studied the effect of different CVSS metrics on the time until exploit, given the CVSS framework is used extensively by NVD in analysing a vulnerability. Conditional probability distributions were calculated by filtering the data to only include a fixed value from one CVSS metric (*e.g.,* all entries with Attack Complexity = High). Power-law distributions with exponential cutoff were fitted to these subsets of data.

Fitting the probability distributions and analysing the medians of the empirical data shows that there are several CVSS metrics which have values that reduce the expected time until exploit by more than half. These are displayed in Table IV.

The number of CVEs disclosed by year has been increasing, particularly with the drastic increase in the third era as shown in Figure 2. We plotted the evolution of the medians of both the time to exploit and CVSS population delay in Figure 1. This shows that in general the median time to exploit is decreasing,

TABLE IV: *Summary statistics for Time to PoC Exploit Data conditioned on CVSS Metrics. Several metric values (*e.g., *Attack Vector = Local) yield a much lower median exploit delay than the all-time median delay of 47 days in Table III.*

CVSS metrics	Values	Number of Entries	Median (Days)
Attack Complexity	Low	3349	45
	High	1489	58
Attack Vector	Local	820	18
	Network	3996	61
Privileges Required	None	4325	59
	Low	484	11
User Interaction	None	3061	59
	Required	1840	29
Confidentiality Impact	None	603	39
	Low	1474	114
	High	2757	37
Integrity Impact	None	765	16
	Low	1462	137
	High	2607	41
Availability Impact	None	748	21
	Low	1376	134
	High	2710	40

however since the start of the third era the CVSS population delay has increased.

We considered CVSS base metrics (which comprise of the eight metrics shown in Table II) and studied their effect on CVE exploit delay. We also conditioned exploit delay by combinations of these metrics, considering two up to four combinations. An ECDF was produced for data filtered using each metric combination and a power law distribution with exponential cutoff was fitted. We considered Attack Complexity, User Interaction, Attack Vector and Confidentiality Impact when conditioning exploit delay on three or four metrics.

The combinations of two CVSS metric values which reduces the median (relative to the all time median delay in Table III), by more than half are shown in Table V. These represent a significant amount of entries. We also studied the effect of conditioning on three and four CVSS metrics. The results are shown in the Appendix in Tables VII and VI. These show that by conditioning on more metrics the median time until exploit can be reduced down as low as five days.

VII. Discussion

CVSS metrics play a key role in informing security professionals on what vulnerabilities to prioritise. Hence, the CVSS population date stamp should be readily included in the NVD date feeds, to help better understand the effort required to analyse a CVE and identify its vulnerability characteristics.

Figure 1(b) also shows how the median CVSS population delay has spiked over the recent two years. This noticeable increase is a cause for alarm given the rapidly decreasing time to exploit for a vulnerability. The increasing CVSS delay not only hinders exploit delay prediction but also gives rise to a bigger problem: a disclosed vulnerability could be exploited well before its CVSS metrics are populated. The trend could render the CVSS information obsolete; vendors would not be able to rely on the CVSS information to prioritise patch development and systems administrators won't be able to use the information to prioritise their patch installations on time.

TABLE V: *Summary statistics for the fitted distribution of Time to PoC Exploit conditioned on two CVSS Metrics. Several metric combinations (*e.g., *Attack Vector = Low and User Interaction = Required) yield a much lower median exploit delay than the all-time median delay of 47 days in Table III.*

CVSS metrics	Values	Number of Entries	Median (Days)
Attack Complexity User Interaction	Low Required	388	7
Attack Complexity User Interaction	High None	100	9
Attack Vector User Interaction	Local Required	246	13
Attack Vector Confidentiality Impact	Local Low	95	13
Attack Vector Attack Complexity	Local Low	669	18
Attack Vector Attack Complexity	Local High	151	19
Attack Vector Confidentiality Impact	Local High	636	20
Attack Vector User Interaction	Local None	574	20
Confidentiality Impact User Interaction	High Required	1193	28
Confidentiality Impact User Interaction	Low Required	359	31
Confidentiality Impact Attack Complexity	High Low	1735	33
Attack Vector User Interaction	Network Required	1520	38
Confidentiality Impact User Interaction	High None	1564	41

Hence, a review of the current CVE analysis process employed by NVD is essential to understand areas of improvement within it and mitigate the observed trend in CVSS delay. Incorporating the CVSS population date stamp in the NVD data feeds can expedite this task: *e.g.,* if CVSS delay is relatively high for CVEs with a particular metric value, processing of that vulnerability aspect in the analysis process could be improved. Table III in Section VI shows how from the beginning of the CVE process, the median and the tail of the exploit delay distribution are larger than the time restricted ranges. This is due to higher exploit delays in the early years of the process. This decreasing trend in exploit delay is to be expected; highly sophisticated vulnerability exploit tools are increasingly becoming available to the public. For instance, NSA's security tools leaked in 2016 contained many sophisticated exploits and backdoors to vendor systems [6]. At the same time fast vulnerability patching or updating is impractical in domains such as critical infrastructure networks due to their high availability demands. Such domains are hence becoming easy targets for even the moderately skilled hackers

TABLE VI: *Summary statistics for the fitted distribution of Time to PoC Exploit conditioned on three CVSS Metrics. Several metric combinations (*e.g., *Attack Complexity = Low, Confidentiality Impact = Low and User Interaction = Required) yield a much lower median exploit delay than the all-time median delay of 47 days in Table III.*

CVSS metrics	Values	Number of Entries	Median (Days)
Attack Complexity Confidentiality Impact User Interaction	Low Low Required	95	5
Attack Vector Attack Complexity User Interaction	Local High None	50	5
Attack Vector Attack Complexity User Interaction	Network Low Required	242	6
Attack Vector Attack Complexity Confidentiality Impact	Local Low Low	76	7
Attack Vector Confidentiality Impact User Interaction	Local Low None	77	7
Attack Vector Attack Complexity User Interaction	Local Low Required	145	7
Attack Complexity Confidentiality Impact User Interaction	Low High Required	264	8
User Interaction Attack Complexity Confidentiality Impact	None High High	93	9
Attack Vector Confidentiality Impact User Interaction	Local High Required	196	13
Attack Vector Attack Complexity Confidentiality Impact	Local High High	121	13
Attack Vector Attack Complexity Confidentiality Impact	Local Low High	515	21
Attack Vector Confidentiality Impact User Interaction	Local High None	440	22
Attack Vector Attack Complexity User Interaction	Local Low None	524	24

[19].

Table IV in Section VI shows how exploit delay is reduced by over 50% (relative to the all time median delay in Table III), when it is conditioned by CVSS base metrics such as AV = Local and PR = Low. Exploit delay is also reduced to some extent when it is conditioned by base metrics such as CI = High and AI = High. Table V and Table VI also describe how exploit delay is likewise reduced when its conditioned by base metric combinations such as (AC = Low, UI = Required) and (AC = Low, CI = High, UI = Required). These results agree with our expectations; the smaller the effort required to exploit a vulnerability (*e.g.,* PR = Low), the lower the exploit

TABLE VII: *Summary statistics for the fitted distribution of Time to PoC Exploit conditioned on four CVSS Metrics. Several metric combinations yield a much lower median exploit delay than the all-time median delay of 47 days in Table III.*

CVSS metrics	Values	Number of Entries	Median (Days)
Attack Complexity Attack Vector Confidentiality Impact User Interaction	Low Network Low Required	95	5
Attack Complexity Attack Vector Confidentiality Impact User Interaction	High Local High None	48	5
Attack Complexity Attack Vector Confidentiality Impact User Interaction	Low Local Low None	76	7
Attack Complexity Attack Vector Confidentiality Impact User Interaction	Low Local High Required	123	7
Attack Complexity Attack Vector Confidentiality Impact User Interaction	Low Network High Required	140	9

delay. Although some of the metrics that were conditioned on may have low impact for a vulnerability, these still may have serious implications for a network. For example, the first entry in Table VII, shows a low confidentiality impact but the remaining metrics may contain high impact values i.e. Availability Impact of High, which may have serious implications for a high availability service.

As Table V through Table VI show, there are a variety of CVSS metric value combinations which reduce the median time until exploit. In particular, there are classes of vulnerabilities with very short median time to exploit (as low as three days), when conditioned on more than two metrics. These vulnerability classes provide significant information for prioritising patching and mitigating vulnerabilities. Moreover, these classes can be leveraged to build more granular predictive models for time to exploit. However, CVSS information must be readily available to achieve this task; *i.e.,* the recent upward trend in CVSS delay needs to be addressed with priority. New risk metrics could be developed that incorporate information about a shorter time to exploit dependent upon vulnerability class. This information can assist in determining the likely time until mitigations need to be deployed.

We observed that from the beginning of 2017 onwards, there is a noticeable increase in the number of disclosed CVEs (Figure 2). This increase in disclosures in recent times, may be overwhelming NVD's CVE analysis process leaving a significant backlog of CVEs to be analysed. Such a backlog would increase the median time until CVSS population. The NVD program may need to increase the amount of staff analysing

disclosed CVEs to address the increase of disclosures

The validity of our approach in computing CVE exploit delay is limited by the availability of accurate data. We assume here, that the date and times reported in the ExploitDB and NVD Databases are accurate. In ExploitDB, this accuracy depends on the reporting of precise time of exploit. This time is based on when the exploit was submitted to the database and may differ to the actual exploit discovery time. However, there is an incentive for a security researcher, to be the first to produce an exploit. The accuracy of the CVE disclosure times is another potential limitation. Some companies have the ability to assign and disclose CVEs and that can cause inaccuracies in the reported CVE disclosure times. An increase in the accuracy of all these data sources and to standardise how they are reported will allow better analysis to develop mathematical models of this process.

VIII. Conclusion and Future Work

In this paper, we use the National Vulnerability Database (NVD) and the Exploit Database to show how CVSS metrics combinations can provide better insight into exploit delay. In particular, we find that there are classes of vulnerabilities with very short median time to exploit (as low as three days), when conditioned on more than two metrics. These vulnerability classes provide significant information for prioritising patching and mitigating exploits. We hope to leverage these classes in future, to build predictive models for time to exploit.

We also show that the median time lag associated with populating CVSS metrics has increased rapidly from a single day prior to 2017 to 19 days in 2018. This is an alarming trend, given the rapid decline in median vulnerability exploit time from 296 days in 2005 to six days in 2018.

IX. Acknowledgement

This project was supported by the Data to Decisions CRC and the Centre of Excellence for Mathematical and Statistical Frontiers (ACEMS).

References

[1] L. Allodi and F. Massacci. A preliminary analysis of vulnerability scores for attacks in wild: the EKITS and SYM datasets. In *Proceedings of the 2012 ACM Workshop on Building analysis datasets and gathering experience returns for security*, pages 17–24. ACM, 2012.

[2] L. Allodi and F. Massacci. Comparing vulnerability severity and exploits using case-control studies. *ACM Transactions on Information and System Security*, 17(1):1–20, Aug. 2014.

[3] M. A. Arfeen, K. Pawlikowski, A. Willig, and D. McNickle. Fractal renewal process based analysis of emerging network traffic in access networks. In *2016 26th International Telecommunication Networks and Applications Conference (ITNAC)*, pages 265–270, Dec. 2016.

[4] M. Bozorgi, L. K. Saul, S. Savage, and G. M. Voelker. Beyond heuristics: learning to classify vulnerabilities and predict exploits. In *Proceedings of the 16th ACM international conference on Knowledge discovery and data mining*, pages 105–114. ACM, 2010.

[5] B. L. Bullough, A. K. Yanchenko, C. L. Smith, and J. R. Zipkin. Predicting exploitation of disclosed software vulnerabilities using open-source data. In *Proceedings of the 3rd ACM on International Workshop on Security And Privacy Analytics*, IWSPA '17, pages 45–53, New York, NY, USA, 2017. ACM.

[6] S. D. Carberry. Shadow Brokers leak trove of NSA hacking tools. [Online]. Available: https://fcw.com/articles/2017/04/14/shadow-brokers-swift-hack.aspx, 2017.

[7] A. B. Downey. Evidence for long-tailed distributions in the Internet. page 13.

[8] DSS, PCI. Payment card industry data security standard - requirements and security assessment procedures(version 3.2.1). *PCI Security Standards Council. Wakefield, MA. USA*, 2018.

[9] M. Edkrantz and A. Said. Predicting cyber vulnerability exploits with machine learning. In *SCAI*, 2015.

[10] Forum for Incident Response and Security Teams. Common vulnerability scoring system v3.0: Specification document. [Online]. Available: www.first.org/cvss/specification-document, 2015.

[11] Forum of Incident Response and Security Teams(FIRST). Common Vulnerability Scoring System SIG. [Online]. Available: https://www.first.org/cvss, 2018.

[12] S. Frei. *Security Econometrics: The Dynamics of (In)Security*. BookSurge Publishing, 2009.

[13] P. Mell and T. Grance. Use of the common vulnerabilities and exposures (CVE) vulnerability naming scheme. Technical Report NIST SP 800-51, National Institute of Standards and Technology, Gaithersburg, MD, 2002.

[14] P. Mell, K. Scarfone, and S. Romanosky. Common vulnerability scoring system. *IEEE Security & Privacy*, 4(6), 2006.

[15] P. Mell, K. Scarfone, and S. Romanosky. A complete guide to the common vulnerability scoring system version 2.0. [Online]. Available: www.first.org/cvss/v2/guide, 2007.

[16] Microsoft Corporation. Microsoft Security Response Center Security Bulletin Severity Rating System. [Online]. Available: http://www.microsoft.com/technet/security/bulletin/rating.mspx, 2002.

[17] National Institute of Standards and Technology. National Vulnerability Database. [Online]. Available: http://nvd.nist.gov/, 2018.

[18] Offensive Security. Exploits Database by Offensive Security. [Online]. Available: https://www.exploit-db.com/, 2018.

[19] N. Perlroth and D. E. Sanger. Hackers hit dozens of countries exploiting stolen NSA tool. *New York Times*, 2017.

[20] S. M. Rajasooriya, C. P. Tsokos, and P. K. Kaluarachchi. Cyber security: Nonlinear stochastic models for predicting the exploitability. *Journal of Information Security*, 08(02):125–140, 2017.

[21] B. Ryu and S. Lowen. Fractal traffic models for Internet simulation. In *Proceedings of Fifth IEEE Symposium on Computers and Communications*, pages 200–206, 2000.

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

[23] M. Schiffman, A. Wright, D. Ahmad, and G. Eschelbeck. The common vulnerability scoring system. *National Infrastructure Advisory Council, Vulnerability Disclosure Working Group, Vulnerability Scoring Subgroup*, 2004.

[24] Scrapinghub Ltd. Scrapy | A fast and powerful scraping and web crawling framework. [Online]. Available: https://scrapy.org/, 2018.

[25] The MITRE Corporation. Common vulnerabilities and exposures (CVE). [Online]. Available: http://cve.mitre.org/, 2018.

[26] United States Computer Emergency Readiness Team (US-CERT). US-CERT Vulnerability Note Field Descriptions. [Online]. Available: http://www.kb.cert.org/vuls/html/fieldhelp, 2006.