Cybersecurity Information Sharing: Analysing an Email Corpus of Coordinated Vulnerability Disclosure

Kiran Sridhar, Allen Householder, Jonathan Spring, Daniel W. Woods

Abstract

Information sharing is widely held to improve cybersecurity outcomes whether its driven by market forces or by cooperation among firms and individuals. Formal institutions may be established to facilitate cooperative information sharing. This paper presents a case-study of such an institution, the CERT Coordination Center (CERT/CC), and provides quantitative insights based on the meta data of 434K emails passing through CERT/CC since 1993. Our longitudinal results show how the volume and proportion of emails about different products and vendors has varied over time. We also analyse the distributions of information sharing volume, participation, and duration across 46K vulnerabilities. Finally, we run regressions to understand how the volume of information sharing and decision to coordinate vary based on properties of the vulnerability and the affected vendors. We discuss what has changed, the appropriateness of a competitive or cooperative framing, and limitations.

1 Introduction

The distribution of information about security vulnerabilities determines who compromises which systems. After a bug has been discovered, information can flow to actors in different ways. Benign information flows include a vendor being notified about a bug, releasing a patch, and it being applied before an exploit is created. Alternatively, a malicious actor could discover and exploit a vulnerability until compromise is detected.

Information sharing research seeks to understand how and why information flows. Attention has predominantly been focused on proprietary threat intelligence feeds [1, 2], bug bounty programs [3–8], vulnerability credits [9], and illegal markets [10–12] for which access and participation are driven by financial incentives. Voluntary information sharing also takes place without these incentives, such as via informal relationships between security professionals or in voluntary networks facilitated by institutions like Information Sharing and Analysis Centers (ISACs) [13].

Increased voluntary information sharing is widely viewed as desirable [14–16], but there is less consensus about what prevents it from happening. Barriers identified in the literature include free-riding [14, 17], lack of standards [18], reputation damage [19], and social norms [13]. Rather than trying to explain why a phenomena does not happen, we identify determinants of when it does by conducting a case-study of one of the longest standing cybersecurity information sharing institutions.

This paper explores factors that explain whether and how much cybersecurity information is shared. We make empirical observations based on a set of 434K emails passing through the CERT Coordination Center (CERT/CC), which is tasked with facilitating information sharing about vulnerabilities among vendors and consumers of affected products. Our descriptive statistics highlight how one institution and its constituents have adapted to developments since 1993 like the establishment of a vulnerability numbering system, and the proliferation of bug-bounty programs and IoT products. We then show how the volume of information shared and decision to coordinate can be explained by factors like the risk associated with the vulnerability as well as the number and maturity of affected vendors.

CERT/CC should not be confused with US-CERT. US-CERT was an element of the US Department of Homeland Security (DHS) from 2003 to 2018. In 2018, US-CERT's functions were subsumed under the Cybersecurity and Infrastructure Security Agency (CISA). CERT/CC formed in 1988 and has been operated by Carnegie Mellon University from 1988 to the present. While the funding for CERT/CC is US-centric and includes DHS, CERT/CC does not limit vulnerability coordination by geographic region. CERT/CC primarily coordinates in English; the extent to which language limits information exchange in this context is unclear.

The paper is organized as follows. Section 2 identifies related work. Section 3 describes our research design. Section 4 provides longitudinal descriptive analysis of the emails, and also an analysis of the distribution of three proxies for information sharing per vulnerability. Section 5 seeks to identify factors that explain the variance in the volume of messages about specific vulnerabilities and the decision of which vulnerabilities to share information about. Section 6 interprets the results and highlights areas for future work. Section 7 concludes the paper.

2 Related Work

Research into software vulnerabilities can be structured according to a timeline of events [20]. Putting aside the metaphysical question of whether an undiscovered vulnerability exists, timelines begin when the first individual discovers a vulnerability. Employers exert influence over vulnerabilities found by their employees, such as those discovered as part of secure software engineering [21] or via the vulnerabilities equity process [22, 23]. Miller [24] provides a useful taxonomy of possible next steps for independent researchers: publicly announcing the vulnerability, privately reporting it to the vendor, selling the information to a legitimate buyer, and selling the information on the black market.

One should distinguish between legitimate markets in which vendors pay for vulnerability reports to be used internally (bug bounties) and intermediaries who pay for vulnerability reports to be re-sold (vulnerability brokers) [25]. Kannan and Telang [26] introduced a theoretical model showing that private vulnerability brokers always lead to worse social outcomes than a public broker would. The private brokers of 2005 have morphed into exploit acquisition platforms like Zerodium. Public vulnerability brokers remain federated [27]: examples include JPCERT/CC in Japan and CISA (part of the Department of Homeland Security) for infrastructure in the US.

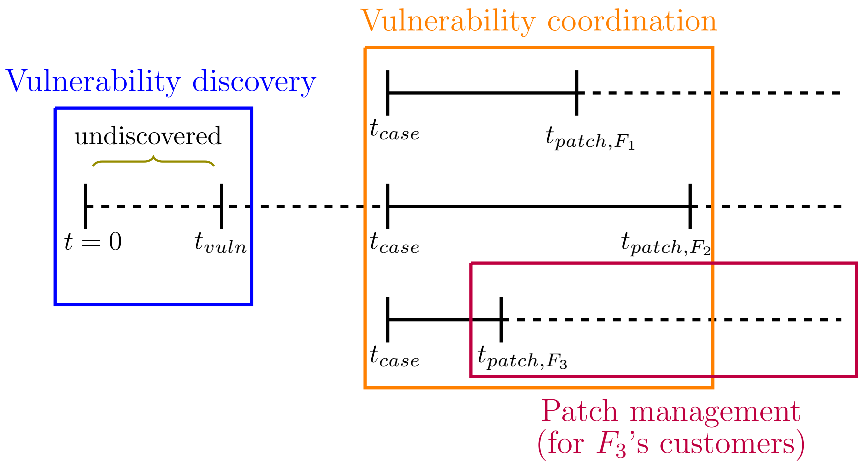
Bug bounties have proliferated. HackerOne manages programs for many companies [8], and firms including Facebook, Google, and Apple run independent schemes [28]. Research into bug bounty programs spans theoretical work into optimal design [28–30] and empirical work evaluating the effectiveness of existing programs [3–8].

Illegitimate markets for vulnerabilities also exist. Such markets are difficult to study because anything observable to academic researchers is even more likely to be observed by law enforcement [31][1]. Consequently, research focuses on the least sophisticated criminals who operate in public channels [32].

The other end of the vulnerability timeline concerns patch deployment. Cavusoglu [33] derived the optimal patching rate in a model assuming homogeneous vulnerabilities, an assumption that was relaxed in [34], and showed how cost sharing or liability can synchronise patch release and deployment. Inconsistent deployment of patches motivated research into dynamics [35–40], notification [41–44] and prioritisation [45–47]. However, deployment is premised on patch availability, which is the focus of this paper (see Figure 1).

[1]Unless researchers are better at infiltrating forums, using methods like those employed by Allodi et al. [11, 12].

Figure 1: Coordination takes place when one vulnerability affects multiple vendors. In this example, a vulnerability impacting F_1, F_2 and F_3 is discovered at t_{vuln}. The CERT/CC is unaware until coordination begins at t_{case} and it ends when all firms have developed a patch.



Patch release timing can be researched since patch availability is externally observable, whereas patching effort occurs privately (either totally within the firm or among multiple firms) and is thus harder to observe. Arora et al. [48] provide evidence that patch release is accelerated by public disclosure of the vulnerability and that vendors are more responsive to more severe vulnerabilities. Using a similar research design, Temizkan et al. [49] show that vulnerabilities affecting multiple vendors lead to patch release 1.64 times faster than with vulnerabilities affecting a single vendor. They argue competition between vendors drives vendors to prioritise release.

This paper describes the cross-firm effort that goes into patch development. We build on prior work [48, 49] by testing similar hypotheses, which will be justified and explored in Section 5. We also build on Howard's seminal thesis covering CERT from 1989-1995 [50] and recent qualitative work [27].

3 Research Design

We derive our hypotheses about coordinated vulnerability disclosure in Section 3.1. We then describe data generation in Section 3.2.

3.1 Hypothesis Generation

The first set of hypotheses concern *how much* information is shared, and the second set concern *whether* CERT/CC decide to coordinate a vulnerability. The second question is asked for the policy regime in which CERT/CC stopped coordinating every vulnerability.

How Much Information Sharing Just as Arora et al. [48] found that severe vulnerabilities were patched more quickly than less severe vulnerabilities, we would expect that more severe vulnerabilities are the subject of higher volumes of information sharing than less severe vulnerabilities.

H1: More severe vulnerabilities receive a higher volume of information sharing.

Previous studies [49, 51] have found that patches are developed more quickly for vulnerabilities affecting multiple vendors. Temizkan et al. [49] suggest that competition with peers leads vendors to prioritise releasing a patch. This zero-sum logic suggests that vendors are less likely to share information for multi-party vulnerabilities. A more cooperative framing, which was also invoked for **H1**, suggests more information will be shared for multi-party vulnerabilities.

H2a: Vulnerabilities that impact a greater number of vendors receive a *lower* volume of communications than vulnerabilities that impact fewer vendors.

H2b: Vulnerabilities that impact a greater number of vendors receive a *greater* volume of communications than vulnerabilities that impact fewer vendors.

Multi-vendor communications could also display different distributions over time. Past empirical results [48, 49] suggest that multi-vendor vulnerabilities are swiftly patched, in which case we would expect communications to resolve sooner. However, multi-party vulnerability exchanges could be held up by the least responsive vendor, in which case information sharing would continue for longer.

H3a: Communication volume decreases *quicker* for vulnerabilities that impact a greater number of vendors than those that impact a smaller number of vendors.

H3b: Communication decreases *slower* for vulnerabilities that impact a greater number of vendors than those that impact a smaller number of vendors.

In the first hypothesis (**H3a**), time to resolve resembles the min of involved participants, whereas the second is closer to the max.

Next, we turn our attention to the vulnerabilities for which CERT/CC had prior notice before publication. If CERT/CC begins its coordination activity before a vulnerability is broadly known, then it is likely a locus of communication. This may suggest that it receives higher volumes of communication for these vulnerabilities. Alternatively, Arora et al. [48] find that the specter of disclosure motivates vendors to more quickly resolve bugs. If vendors are more receptive to communications that take place before publication, this could obviate the need for high levels of communication.

H4a: There is *less* information sharing for vulnerabilities in which communications began before the vulnerability was published.

H4b: There is *more* information sharing for vulnerabilities in which communications began before the vulnerability was published.

Finally, we anticipate that vulnerabilities affecting products further up supply chains, such as hardware and operating systems, require more coordination effort than vulnerabilities in applications. CERT/CC may prioritize these vulnerabilities, since they require a greater number of stakeholders to resolve and because they affect many people. It also may be harder to exhume and remediate these vulnerabilities, leading to higher volumes of information sharing [52]. We thus expect operating system (OS) or hardware vulnerabilities to be the source of greater volumes of information sharing than application vulnerabilities.

H5: Vulnerabilities further up the software supply chain require higher volumes of communication to coordinate.

H6: Vulnerabilities further up the software supply chain require longer durations of time to coordinate.

We now turn to the decision to coordinate.

Whether to Coordinate CERT/CC states that it "prioritize[s] reports that affect multiple vendors" [53].

H7: Vulnerabilities which are associated with more vendors are more likely to be coordinated by CERT/CC.

Similarly, CERT/CC states that it prioritizes the vulnerabilities that are most impactful and which pose the most imminent danger to "critical or internet infrastructure" [53]. We can test whether this in fact aligns with CERT/CC coordination practice, or whether a threat-based metric better aligns with coordination.

H8: More severe vulnerabilities are more likely to be coordinated by CERT/CC.

H9: Vulnerabilities with publicly released exploit code are more likely to be coordinated by CERT/CC than those without one.

Some vendors have engaged in vulnerability remediation since the 1990s. They have developed processes, refined processes, and recruited internal talent. Moreover, many organizations who have processed high volumes of vulnerability reports for years have established their own product security incident response teams (PSIRTs) who handle vulnerability coordination in place of CERT/CC. Some industries have created groups for increasing the expertise of their constituent organizations, either by joining FIRST or a sector-specific ISAC, such as the Health ISAC or Financial Services ISAC. Those who are encountering IT vulnerabilities for the first time, such as Internet of Things (IoT) providers, are much more likely to have immature cybersecurity controls and processes [54]. CERT/CC prioritizes vulnerability management for the companies and industries which lack the capacity to handle vulnerability disclosure on their own.

H10: Vulnerabilities impacting vendors who have more experience dealing with bug remediation are less likely to receive CERT/CC coordination.

3.2 Data Generation

To test these hypotheses, we combine email metadata from CERT/CC's *Coordinated Vulnerability Disclosure* program with the information ecosystem surrounding the National Vulnerability Database. Table 1 provides an overview of variables for each of our units of analysis.

CERT/CC Internal Case We were granted access to anonymized metadata about emails passing through CERT/CC's *Coordinated Vulnerability Disclosure* program. Once a vulnerability has been identified, CERT/CC begin to facilitate communications between affected parties including the finder of the vulnerability, vendors of the vulnerable product, and system owners that should mitigate or remediate the vulnerability. The *disclosure* occurs with the publication of a vulnerability note or other public documentation, which is another method of communicating with system owners,

Table 1: The variables associated with each unit of analysis. Emails may be associated with multiple CVE IDs.

Symbol	**Description**
CERT/CC Internal Case	
nmsg	The number of case emails
ndays	The number of days between first and last message
nrecp	The number of unique addresses across all case emails
VU#	The internal case identifier
CVE IDs	All CVEs associated with the case
CVE ID	
$datepub_{cve}$	Date published
$type_{cve}$	Type of affected products in CPE ID (*os*, *app*, *hw* and *mix*)
$vendors_{cve}$	# of unique vendor strings in CPE ID
$impact_{cve}$	CVSS impact score
$exploit_{cve}$	Whether an exploit was discovered [55]
$prior_{cve}$	Whether first CERT/CC email came before *datepub*
CERT/CC email	
date	Date sent
age	Years since the first message was sent
VU#	The internal case identifier
CERT/CC email with non-empty *CVE IDs* label	
type	$type_{cve}$ across *CVE IDs*
vendors	Average of $vendors_{cve}$ across *CVE IDs*
impact	Average of $impact_{cve}$ across *CVE IDs*
exploit	$\max(exploit_{cve})$ across *CVE IDs*
vendorage	Average years since NVD debut across $vendors_{cve}$ in *CVE IDs.*

especially when the population of system owners is large and diverse. Participants may want to know about the existence of a new vulnerability; which systems or versions are affected; under what conditions the vulnerability can be exploited; methods for detecting or preventing exploitation; the existence of a software patch to remediate the vulnerability; and whether attackers are actively exploiting the vulnerability in the wild.

CERT/CC has a long standing policy of holding email contents as confidential to maintain trust and cooperation in the coordination process. Any content analysis has the potential to leak information. The email metadata contains important new information, such as message volume and timing.

Table 2: Summary Statistics

Statistic	Mean	St. Dev.	Min	Pctl(25)	Pctl(75)	Max
cert_n_rcpt	8.4	29.969	1	2	7	1,222
impact	6.36	2.09	1	5	7.5	10
exploit	0.046	0.209	0	0	0	1
prior	0.523	0.499	0	0	1	1
n_days	123.4	343.8	0	0	95	6,226
n_msg	1	2	4	22.4	12	2363

Howver, because a vulnerability disclosure publicizes a vulnerability and the stakeholders involved as a matter of course, we judge we can manage the small risk of disclosing case dynamics via anonymization and grouping results.

Our data set records the time each individual email was sent, the number of recipients and domains, and whether the email was sent (outbound) or received (inbound) by CERT/CC. The emails were hand-labelled with an internal vulnerability ID *VU#*. As recommended by Howard [50], the vulnerability ID is used to group emails into *cases*, which can be thought of as the collection of email threads related to a vulnerability. We use three proxies for information sharing per case. *nmsg* counts the number of emails, *ndays* is the time between first and last message, and *numrecp* counts the number of unique email addresses in a case. The final proxy was calculated internally and shared for each case[2] To link email meta data to information sharing, we assume the distribution of information content across emails is constant. For example, x emails share x times as much information as 1 email. We discuss this later in Section 6.

We then group emails by vulnerability. CERT/CC define a *vulnerability* to be "a set of conditions or behaviors that allows the violation of an explicit or implicit security policy. Vulnerabilities can be caused by software defects, configuration or design decisions, unexpected interactions between systems, or environmental changes" [53, §1.2]. Cases contain a field, *CVE IDs*, for associated Common Vulnerability Enumeration ID, but there is no one-to-one correspondence. Not all cases are associated with CVE IDs, and some cases are associated with multiple CVE IDs.

CERT/CC's case triage policy has changed during the data collection period. Until 2006, CERT/CC aimed to document all known vulnerabilities.

[2]This was impossible for us to calculate as we did not know which addresses (real or anonymised/hashed) sent/received each message.

With the recognition that the NVD had successfully assumed this role, in 2006 there was a shift to only open cases for valid vulnerabilities (as defined above) directly reported to CERT/CC or of particular interest to staff. The CERT/CC vulnerability reporting form states its new policy:

> "CERT/CC does not accept or respond to every report. We prioritize reports that affect multiple vendors or that impact safety, critical or internet infrastructure, or national security. We also prioritize reports that affect sectors that are new to vulnerability disclosure. We may be able to provide assistance for reports when the coordination process breaks down" [56].

We believe this text has been consistent since 2011, but the Internet Archive has it documented from 2018 [56]. More detailed guidance on why CERT/CC may decline a case was published in 2015 [57]. As part of SSVC version 2, CERT/CC published further details about how analysts make coordination triage decisions [47]. Although these details have been updated over time, the policy remains materially consistent from 2006 to 2020. Consistent analyst application of this policy is driven by training and peer review. Several CERT/CC staff members have been participating in vulnerability coordination for 15 to 20 years; this low turnover increases our confidence that the policy in practice remained consistent enough during the data collection period.

CVE ID The CVE labels allow us to pull from the information ecosystem surrounding the National Vulnerability Database (NVD) to classify emails according to vulnerability characteristics.

We extract three salient information items associated with each CVE ID. The Common Platform Enumeration (CPE) provides information about the products affected by the vulnerability. Each product is classified as either an application (*app*), operating system (*os*), or hardware (*hw*). We use this classification ($type_{cve}$) unless the vulnerability affects multiple products of different types, in which case we classify it as *mix*. In addition, we extract the name of the vendor for each affected product. $vendors_{cve}$ counts the number of unique vendor names. We use the impact score from the Common Vulnerability Scoring System as a proxy for the severity of the vulnerability ($vendors_{cve}$).

Two additional variables use supplemental information. As a proxy for whether CERT/CC had prior notice, $prior_{cve}$ is set to one if a CERT/CC email associated with a given CVE was received before $datepub_{cve}$. Finally, we use a data set [55] built by scraping ExploitDB and MetaSploit to identify whether a public exploit had been published for each CVE ID ($exploit_{cve}$).

CERT/CC email The emails are grouped into months or years using the *date* the email was sent. The variable *age* tracks the number of years since the first communication. We then use the internal case number (*VU#*) to link each email to associated CVE IDs (*CVE IDs*). To address the lack of one-to-one correspondence, we take averages across $vendors_{cve}$ and $impact_{cve}$. We label the case as *mix* if the CVE IDs have different types ($type_{cve}$), otherwise it takes the type of the associated *CVE IDs*. Finally, we set exploit to 1 if there is an exploit available for *any* of the associated CVE IDs.

To develop a proxy (*vendorage*) for the age of each vendor, we count the years between when the vendor first appeared in the NVD and when the email was sent. Each email may be associated with multiple CVEs and each CVE may be associated with multiple vendors, each of which has a unique *vendorage* value. We then take the arithmetic mean of all associated vendors so that each email has one *vendorage* value.

For the duration of the data collection, the technology stack used by CERT/CC was Thunderbird, Lotus Notes, GnuPG, and a small number of custom tools for doing things like managing encrypted mailing lists (that tool is SRmail). In June 2020, CERT/CC began to transition to a fully customized coordination tool, the Vulnerability Information and Coordination Environment [58]. However, that transition does not impact our data collection. The basic analyst workflow during the collection period is captured by [59].

There is little automation in the process of vulnerability coordination. Although automated processes such as fuzzing [60] and program verification [61] can automate initial vulnerability discovery, these tools cannot yet provide a human-intelligible report on what the vulnerability is and why it is important. Much of this coordination work must be done manually. Automated tools prepared reports for VU#528719 and VU#582497 because these vulnerabilities were protocol implementation errors (for SNMP and TLS, respectively) which automated tools could diagnose.

4 Descriptive Statistics

Section 4.1 provides a longitudinal perspective on information sharing coordinated by CERT/CC and also the CVE labelling system. Section 4.2 presents univariate analysis of three proxies for information sharing per vulnerability.

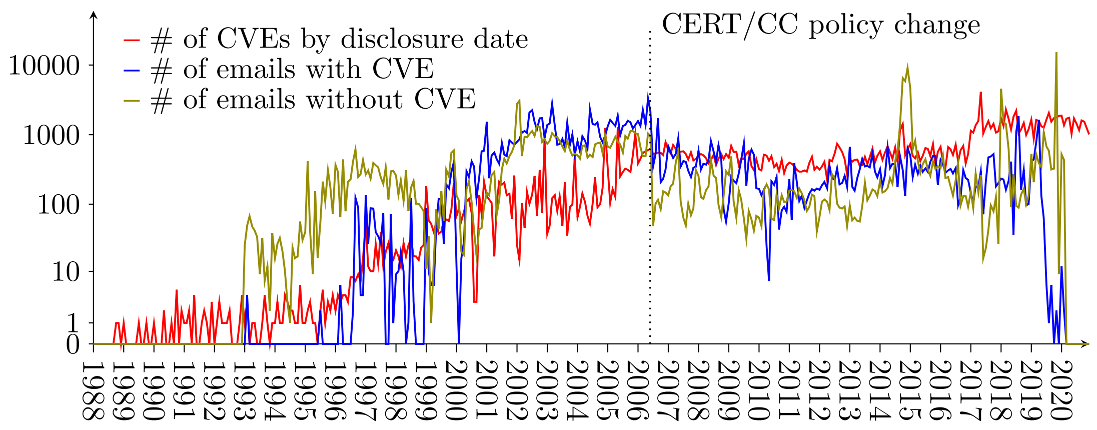


Figure 2: Emails passing through CERT/CC have remained relatively constant since the early 2000s despite high month to month variation. The number of CVE IDs gradually increased since its inception until 2017 when the yearly number spiked and remained high. Note log-scale y-axis.

4.1 Longitudinal

Taking a high-level perspective, Figure 2 shows the number of emails passing through CERT/CC separated according to whether the email has been labelled with a CVE ID. The volume of emails grew until mid-2006, at which time CERT/CC made a policy shift away from trying to document all known vulnerabilities. Since 2008, email volume has stayed relatively level. We also show the monthly number of CVE IDs according to the date the vulnerability was disclosed to the public, which may be different than the date the CVE ID entry was published[3]. CVE IDs have jumped almost an order of magnitude since 2017, to the extent that some months have more CVE IDs published than there are emails. We try to shed light on this puzzle, but first we need to contextualise the data.

The high-level view should give way to a perspective in which CERT/CC simultaneously coordinate multiple cases, each of which corresponds to one vulnerability. For each of the 46.7K cases, we can track three proxies for information sharing: the number of messages sent (*nmsg*), the number of unique recipients (*numrecipients*), and the days between first and last message (*ndays*). Calculating the Spearman's rank correlation coefficient shows intuitive results: the number of messages is positively correlated with the case length ($\rho(\text{nmsg}, \text{ndays}) = 0.76$)) and the number of unique recipients

[3]For example, MITRE created CVE IDs for a number of historic vulnerabilities and the date published field is earlier than the CVE system in some cases.

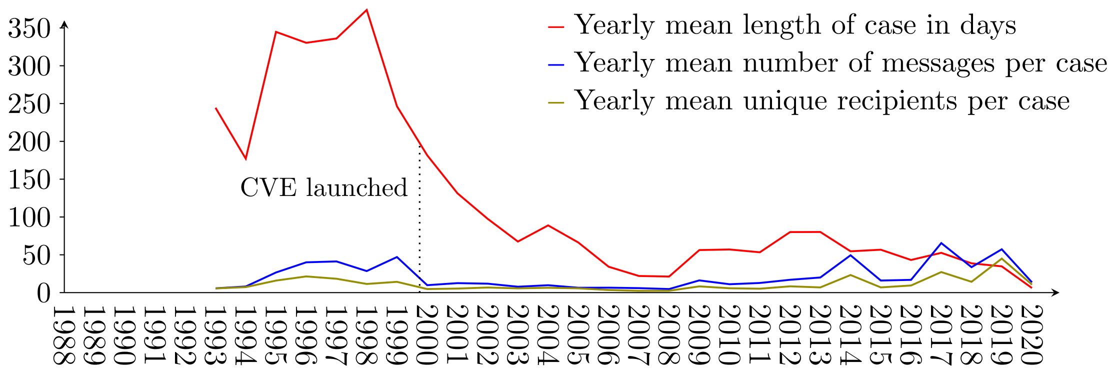


Figure 3: How three proxies for coordination effort vary across three distinct reporting periods; pre-CVE (until around 2000), including every CVE (until 2008), and selective coordination (2008 and onward).

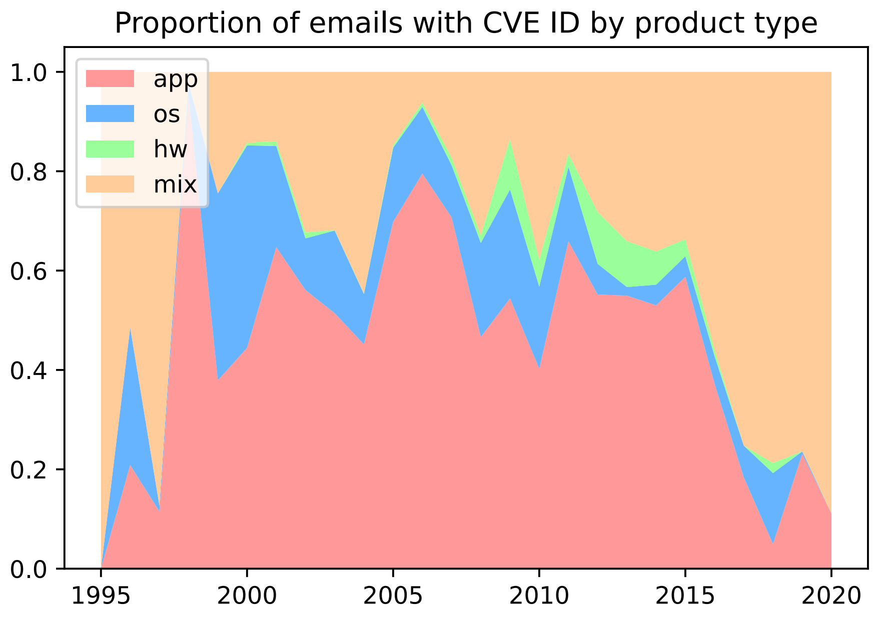
($\rho(\text{nmsg}, \text{numrecipients}) = 0.7$). The relationship between the length of the case and the number of recipients is weaker ($\rho(\text{numrecipients}, \text{ndays}) = 0.59$)).

The different reporting periods can be seen in Figure 3, which shows the yearly mean for each proxy. Before the CVE labelling system was created, the typical CERT/CC vulnerability required about 50 messages and almost a year before information sharing ceased. Then MITRE launched the CVE dictionary in 1999 and CERT/CC began a policy by which they would send at least one email about each CVE ID. MITRE selectively included historic vulnerabilities that had already been discovered.

Figure 2 shows how CVE IDs proliferated and this drove down the average number of messages per case in Figure 3 to below 5 in 2008. All three proxies began to rise when CERT/CC became selective over which vulnerabilities they coordinated. Interestingly, after the policy change, mean case length never reached 1990s levels unlike messages per case, which suggests cases are now resolved in less time. The three proxies in Figure 3 are right censored because cases opened in recent years are still being coordinated.

The metadata provided by CERT/CC quantifies properties of communications rather than the properties of the vulnerabilities associated with each case. Such information can be drawn from the National Vulnerability Database for the subset of emails labelled with a CVE ID (the mustard coloured line in Figure 2). Although the preceding analysis throw away unlabelled cases, it allows us to ingest information about the vulnerabilities for which information is shared.

Returning to the aggregate view, we can track how information sharing



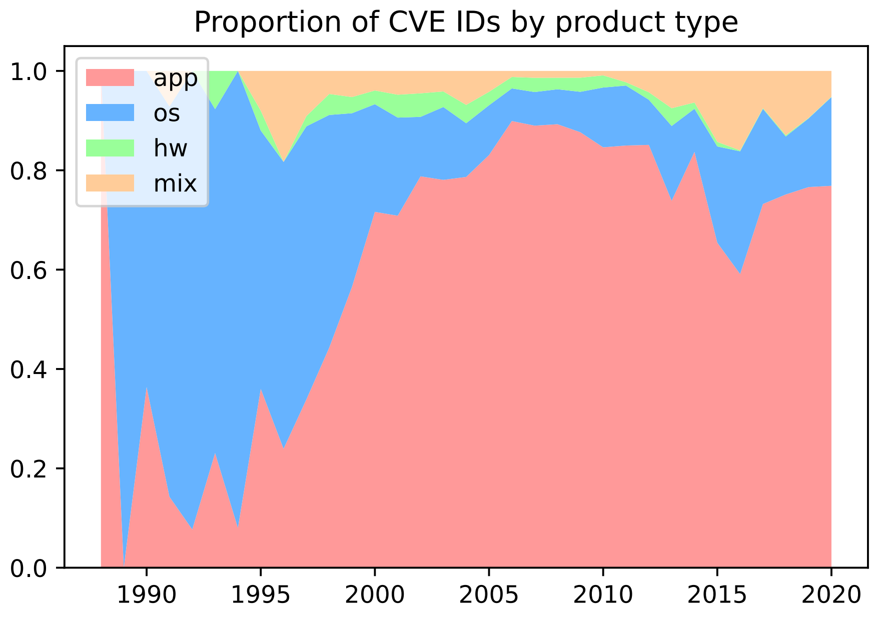


Figure 4: Proportion of information sharing and CVE IDs according to type of affected product. One email may be mapped to multiple CVE IDs. If these are labelled differently, then the email will be labelled as *mix*.

is distributed across the type of the affected product (e.g hardware, OS or application). Figure 4 shows that the majority of CVE IDs and information sharing is concerned with applications. Recent years seem to have seen a rise in vulnerabilities affecting multiple product types. But properties of recent data may also reflect the ongoing process of labeling cases with CVE IDs as the CVE ID may not be published when the case is opened and must later be labeled.

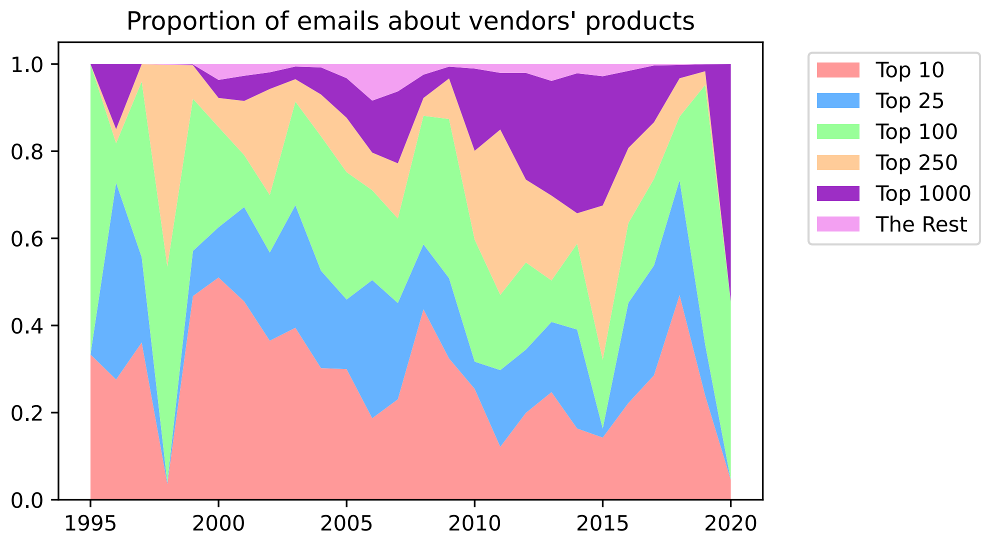
We can also link CVE IDs to the vendor of the affected products. There are over 23 000 unique vendors, which can be grouped into bins according to the frequency with which they are linked to emails with CVE IDs. Figure 5 uses such bins to show the proportion of emails and CVE IDs according to the volume related to each vendor. From 2000 to around 2008, an increasing fraction of CVE IDs affect new/unpopular vendors (e.g vendors outside the top 100) but then this trend changes direction culminating in 2017 when there was as many CVE IDs affecting the top 10 vendors as for the remaining 23 000.

Despite MITRE assigning CVE IDs to so many vendors, information sharing is more focused on those vendors whose products are regularly associated with CVE IDs. The proportional share of CERT/CC emails displays much more year on year variability likely because individual vulnerabilities are weighted by the number of associated emails, which means one vulnerability can have a large effect. However, we do see a general trend towards information sharing about the top 100 to 250 vendors. It is unclear what effect large vendors establishing product security teams (PSIRTs) and bug bounty programs has had on the number of reports about those vendors to CERT/CC.

4.2 Information Sharing per Vulnerability

The yearly averages presented in the previous section obscure the variance of information across vulnerabilities. For example, one case (VU#582497) involved over 27 000 emails and it can be seen as the 2014 spike in Figure 2. This motivates considering the distribution of the three proxies, namely the number of messages, recipients and length of each CERT/CC vulnerability.

We model this with the `powerlaw` package [62]. Figure 6 shows the empirical complementary cumulative distribution of *nmsg*, *nrecp*, and *ndays*, along with four distributions fitted by leaving the minimum value as a free parameter and only setting a maximum value for the case length (equal to the age in days of CERT/CC). The number of observations (n = 46.7K) far exceeds the rule of thumb ($n > 50$) for extracting reliable parameter estimates [63].



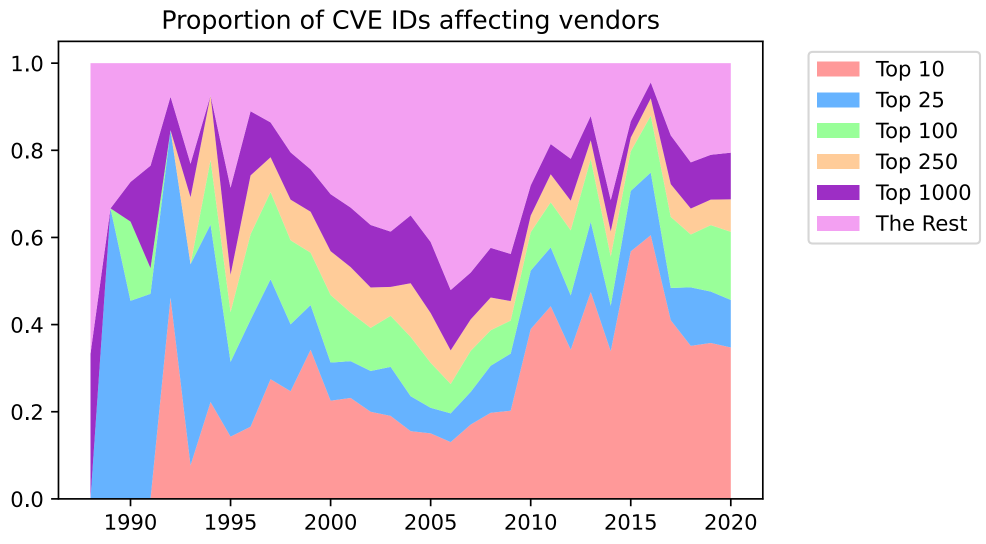


Figure 5: Information sharing coordinated by CERT/CC is concentrated among around 250 vendors, whereas CVE IDs affect over 23K vendors. From 2000–2008, the proportion of CVE IDs affecting the top 25 vendors declined before this trend reversed direction. This culminated in 2017 when the top 10 vendors were affected by more CVE IDs than all others combined.

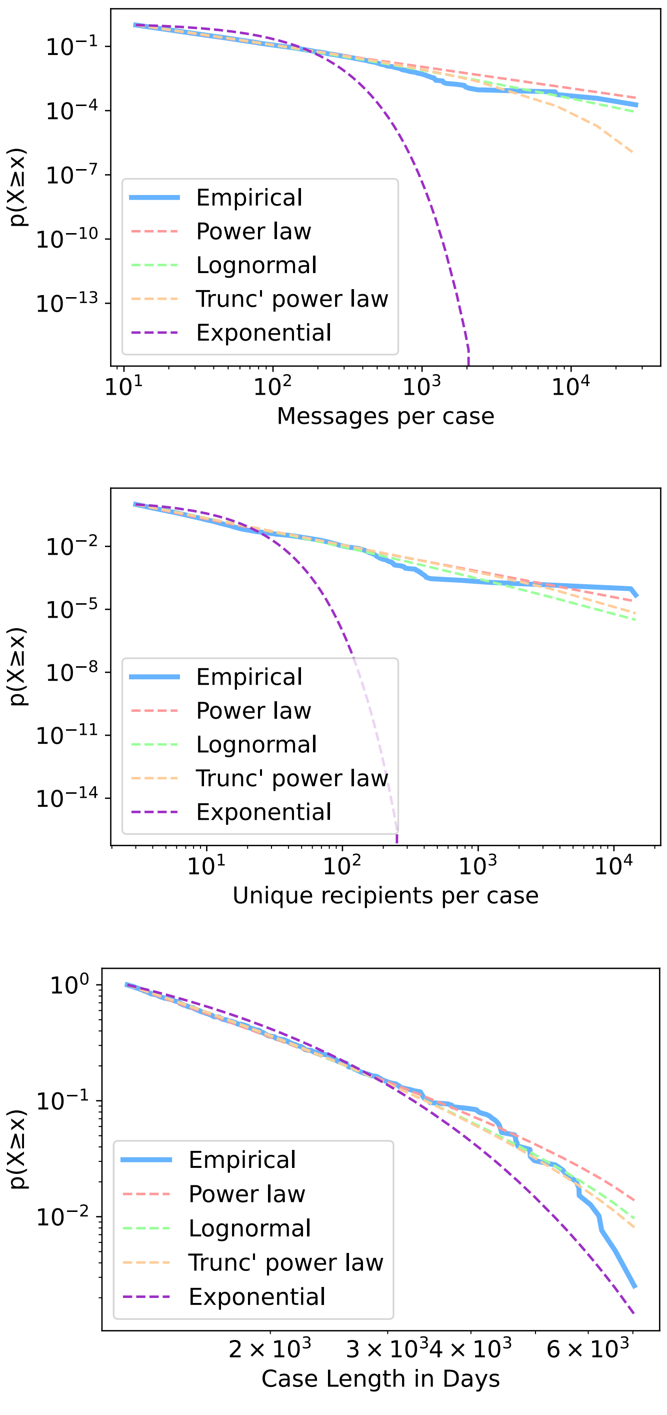


Figure 6: Number of messages and recipients follow a heavy-tailed distribution, whereas the case length in days is much closer to an exponential distribution.

The fitted minimum value for case length (1220 days) excludes a lot of data relative to the other minimum values (12 messages and 3 recipients). The difference can also be seen in the scaling parameter α of the power law, which provides information about whether the standard deviation and mean are defined via the test of $\alpha < 2$ and $\alpha < 3$ respectively [62]. The results ($\alpha_{nmsg} = 2.01$ and $\alpha_{nrecp} = 2.23$) suggest an undefined standard deviation for the number of messages and recipients per vulnerability, whereas it is defined for case length ($\alpha_{ndays} = 3.05 > 3$).

We explore whether each distribution has a heavy tail by fitting an exponential distribution [64, p. 412]. Both the distribution of sent messages and recipients are better explained by a power law than an exponential ($p < 10^{-8}$ for both), but we cannot rule this out for case length ($p = 0.08$). Visual confirmation for this can be found in Figure 6.

Both number of messages and recipients are better explained by a truncated power law ($p < 10^{-5}$ for both) and a lognormal ($p < 0.05$ for both) than an unconstrained power law. We found no statistical support for distinguishing between the truncated power law and the lognormal.

In summary, we found reasonable support for the number of messages sent and the number of recipients following a heavy-tailed distribution. However, comparisons with alternative distributions suggest the tail is less extreme than a pure power law. This finding is consistent with other distributions associated with vulnerability discovery [37] and the Internet [63, 65].

5 Regression Analysis

This section explores hypotheses about the determinants of information sharing. Section 5.1 considers the volume of emails associated with each CVE ID between 1995 and 2019. Section 5.2 explores which vulnerabilities CERT/CC coordinated following 2008, after the policy stabilized from the mid-2006 policy change to become more selective.

5.1 Volume of Information Sharing

We begin our empirical analysis by leveraging an unbalanced panel, comprising of 434,000 emails sent between the years 1995 and 2019. We seek to estimate the number of emails Y_{ij} in year j about the vulnerability with CVE ID i. This analysis assumes each email shares the same amount of information, a limitation we later return to.

Model specification We posit the following regression model:

$$log(Y_{ij}) = \beta_0 + \beta_1 \cdot impact_i + \beta_2 \cdot nvendors_i + \beta_3 \cdot prior_i + \beta_4 \cdot os_i + \beta_5 \cdot mix_i + \beta_6 \cdot hardware_i + \beta_7 \cdot age_{ij} + \beta_8 \cdot nvendors_i \cdot age_{ij} + \beta_9 \cdot os_i \cdot age_{ij} + \beta_{10} \cdot mix_i \cdot age_{ij} + \beta_{11} \cdot hardware_i \cdot age_{ij} + \beta_{12} \cdot log(numrecipients_i) \quad (1)$$

The following provides an intuitive interpretation of each variable, whereas Section 3.2 and Table 1 explain how each was calculated.

We control for the number of email recipients of communications about a vulnerability via $log(numrecipients_i)$ because we expect that emails sent to more people will receive a higher number of responses independent of substance. We take the natural log because the variable distribution has an extreme right skew (see Figure 6).

The $impact_i$ variable is the vulnerability's CVSS impact score and the $exploit_i$ variable is a dummy variable indicating whether exploit code for the vulnerability is ever made publicly available. A positive β_1 coefficient would validate **H1**

We run a further robustness test where we include a dummy variable, $exploit_i$, in our regression model. This variable takes on the value 1 if a public exploit is ever released. Unfortunately, the exploit variable dataset only starts in 2012; for the robustness check, we include a modified panel containing all emails from 2012 through 2019. The $nvendors_i$ variable is a numerical variable indicating how many vendors were impacted by a vulnerability. A negative β_2 coefficient would support **H2a** while a positive coefficient would support **H2b**.

The dummy variable $prior_i$ captures whether CERT/CC began communicating about a vulnerability before it was published in the NVD. A negative β_3 coefficient would provide evidence for **H4a** while a positive β_4 coefficient would support **H4b**.

The os_i, $hardware_i$, and mix_i variables are dummy variables indicating whether the vulnerability affected products classified as an application, operating system, hardware, or a mix. To avoid perfect multicolinearity, the regression is normalized to the *app* classification. The coefficient values of β_5 through β_7 correspond to **H5**.

The age_i variable is the number of years since a vulnerability's first communication via CERT/CC; we know that over time, vulnerabilities are the subject of less communication. The regression has a series of interaction terms: the age_i variable is interacted with the category of vulnerability and with the number of vendors that are impacted by the vulnerability. A negative β_9 value on the vendor age interaction term would validate **3a**; a positive value would validate **3b**. The β_{10} through β_{12} coefficients are employed to test **H6**.

We ran an F-test to see whether to include fixed-effects, a common econometric practice. We resoundingly conclude (p-value ≈ 0.00) that we should include year fixed-effects, but not vulnerability fixed-effects.

Results Table 3 presents an expanded model as well as a parsimonious model without interaction terms. The expanded model has a slightly higher overall adjusted R^2 and a significantly higher "within" R^2, which means that it explains more of the variation in communication about individual vulnerabilities over time. Thus, we conclude that this expanded model is more appropriate.

We posited that more severe vulnerabilities lead to higher levels of information sharing. Contra **H1**, vulnerabilities with higher CVSS scores do not receive higher volumes of communication; our coefficient measures were statistically insignificant. Looking at our robustness check conveyed in Table 7, we would expect vulnerabilities with publicly available exploits to receive 4% fewer communications than those without public exploits; this measure is almost statistically significant at the 5% level. This suggests that bug severity and exploitability do not drive increased communications.

In the first year that CERT/CC coordinates a vulnerability, it sends 20% more emails for each additional vendor who is impacted by the vulnerability. This validates **H2b**. We also find that the interaction term between vendor and vulnerability age is negative and statistically significant, validating **H3b**. However, the effects of the vendor term are far larger than the effects of the vendor vulnerability age interaction term. Ninety percent of vulnerabilities receive no CERT/CC communications after they are three years old.

Our model suggests that CERT/CC will communicate 2% more about vulnerabilities that it started coordinating before they were published in the NVD than about vulnerabilities it began coordinating after they were published in the NVD. This statistically significant coefficient estimate supports **H4b**, but the effect is marginal.

We find more evidence against **H5** than for it. Looking at the parsimonious regression model, although mixed vulnerabilities have greater amounts of overall communication than application vulnerabilities, as we had hypothesized, there is no statistically significant difference between the communications of hardware and OS vulnerabilities relative to application vulnerabilities.

However, **H6** is supported. There are a handful of vulnerabilities that take greater than five years to coordinate; these are most likely to be mixed

Table 3: Panel Data Analysis: The dependent variable Y_{ij} is the number of emails in the month j about a CVE ID x. The explanatory variables are drawn from the NVD and the metadata (see Table 1).

	Parsimonious	Expanded
impact	0.00	0.00
	(0.00)	(0.00)
numvendors	0.02^{***}	0.20^{***}
	(0.00)	(0.00)
prior	0.04^{***}	0.02^{*}
	(0.01)	(0.01)
os	0.03^{**}	-0.06
	(0.01)	(0.04)
mix	0.13^{***}	-0.18^{**}
	(0.02)	(0.07)
hardware	-0.04^{**}	-0.01
	(0.02)	(0.09)
vulnage	-0.36^{***}	-0.22^{***}
	(0.00)	(0.00)
lrecp	0.25^{***}	0.28^{***}
	(0.01)	(0.01)
numvendors*age		-0.03^{***}
		(0.00)
os*age		0.01^{*}
		(0.01)
mix*age		0.05^{***}
		(0.01)
hardware*age		-0.01
		(0.01)
Adjusted Within R^2	0.12	0.15
Adjusted R^2	0.92	0.92

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

vulnerabilities. The longest mixed vulnerability coordination was thirteen years. When a mixed vulnerability is thirteen years old, it receives 47% more vulnerabilities than an application vulnerability of the same age, impacting the same vendors, and of the same severity.

We also examine the results of the time fixed effects. There is a major shift in the second half of the panel. From 2006 to 2016, the annual output of CVEs stayed relatively flat. However, in 2017, there was a surge in CVE IDs: 14,688 vulnerabilities were published that year, compared to just 6,449 in the prior year.

Despite the fact that more vulnerabilities were published later in the time series, information sharing volume does not decrease later in the time series for vulnerabilities coordinated by CERT/CC (see Figure 7 which captures time fixed effects). This suggests that CERT/CC has grown more selective as more vulnerabilities are published. Indeed, CERT/CC now coordinates a fraction of the vulnerabilities which are published in the National Vulnerability Database. While it coordinated 5.8% of all NVD vulnerabilities released in 2012, it coordinated 0.2% of the NVD vulnerabilities released in 2019. The annual drop from 2016 to 2017 was most pronounced: CERT/CC coordinated 2.04% of vulnerabilities in 2016 and 0.83% of vulnerabilities in 2017. We thus wanted to study the factors that predict whether CERT/CC coordinates vulnerabilities.

5.2 The Decision to Coordinate

To understand the types of vulnerabilities CERT/CC coordinates and how this has changed over time, we run cross-sectional logit regressions examining all vulnerabilities published by the NVD from the years 2012 to 2019. We examine the factors that increase and decrease the likelihood that an NVD entry (that is, a vulnerability with a CVE ID) is coordinated by CERT/CC.

Model specification We run the following regression to test hypotheses **H7** through **H10**:

$$Y = \beta_{13} + \beta_{14} \cdot \mathit{numvendors} + \beta_{15} \cdot \mathit{exploit} + \beta_{16} \cdot \mathit{impact} + \beta_{17} \cdot \mathit{vendorage} \tag{2}$$

In this regression, Y is a binary choice variable, which takes on the value of 1 if a CVE ID is coordinated by CERT/CC and a value of 0 if it is not coordinated by CERT/CC. The *numvendors* variable measures the number of vendors who are impacted by a vulnerability. We test **H7** by examining the β_{14} coefficient. A positive value would support our hypothesis. The

Table 4: Logit Regressions: Exploring the factors associated with whether CERT/CC coordinated a given CVE ID released in the year 201x. Odds ratios are reported.

	2012	2013	2014	2015	2016	2017	2018	2019
exploit	2.78^{**}	2.66^{***}	2.39^{**}	2.31^{*}	3.94^{**}	8.08^{***}	1.25	5.52^{**}
	(0.91)	(0.73)	(0.75)	(0.78)	(1.81)	(3.73)	(0.46)	(3.37)
vendorage	0.94^{*}	0.95	0.97	0.89^{***}	0.84^{***}	1.01	0.93^{***}	0.93^{**}
	(0.02)	(0.03)	(0.01)	(0.02)	(0.02)	(0.02)	(0.02)	(0.02)
impact	1.06^{*}	1.17^{***}	1.12^{*}	1.16^{***}	0.99	1.12^{*}	1.15^{**}	1.15^{*}
	(0.03)	(0.05)	(0.06)	(0.05)	(0.06)	(0.05)	(0.05)	(0.08)
n_vendors	0.53^{*}	0.96	0.96	0.56^{**}	1.21	1.76^{**}	1.61^{***}	1.26^{**}
	(0.16)	(0.26)	(0.22)	(0.12)	(0.17)	(0.38)	(0.14)	(0.09)
Num.obs.	5287	5186	7926	6493	6449	14638	16511	17296
ℓ	-1132.38	-848.74	-971.45	-700.84	-538.67	-759.25	-839.76	-262.35
Deviance	2264.75	1697.48	1942.89	1401.68	1077.33	1518.49	1679.52	524.69

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$; ℓ: Log Likelihood

exploit variable is a dummy variable which takes on the value 1 if there is publicly available exploit code for a given CVE ID. The *impact* variable captures the CVSS impact score. We test **H8** and **H9** by examining the β_{15} and β_{16} coefficients respectively. Positive coefficient values would support our hypothesis.

The *vendorage* variable measures the number of years since a vulnerability first associated with a particular vendor appears on the NVD. In the case of multi-vendor vulnerabilities, we captured the mean vendor age. We test **H10** by examining the β_{17} coefficient. A negative coefficient value would support our hypothesis.

Results We run the logit regression for every year from 2012 to 2019. The coefficient values in Table 4 represent the odds ratio: for a one unit increase in each variable in question, the coefficient indicates the factor by which the odds of CERT/CC coordination change. For instance, a vulnerability with public exploit code in 2012 would be 2.78 times more likely to be coordinated by CERT/CC than a vulnerability without public exploit code.

Over time, more vulnerabilities have been discovered. Consistent with its remit and **H7**, as the number of vulnerabilities in the NVD increased, CERT/CC became increasingly focused on the vulnerabilities affecting more vendors. Vulnerabilities with fewer vendors were more likely to be coordinated by CERT/CC than vulnerabilities with more vendors earlier in the time series. By 2019, adding one vendor to a vulnerability increased the probability of CERT/CC coordinating the vulnerability by 25%.

CERT/CC became more selective about the vulnerabilities that it coordinated as the number of CVE IDs increased. Some of the particularly low values of the *n_vendors* coefficient, such as 2012 and 2015 in Table 4, could have been due to outliers. The vulnerabilities with the largest numbers of vendors in those years may not have been coordinated by CERT/CC, deflating the coefficient. Our robustness check, described in Table 6 (in the Appendix) did not significantly change the coefficient on the number of vendors.

CERT/CC is much more likely to coordinate a vulnerability for which there is a public exploit (**H9**). However, the odds ratio for this variable fluctuates. In 2017, CERT/CC was over eight times more likely to coordinate a vulnerability with public exploit code; in 2018, it was just 1.25 times more likely. Vulnerabilities with a higher CVSS impact score were more likely to be coordinated by CERT/CC in every year except 2016 when the effect was insignificant, which supports **H8**. Nonetheless, CVSS impact scores have less bearing on CERT/CC coordination than the presence of a public exploit. In 2013, when CVSS impact scores had their most significant effect on the odds of CERT/CC communication, a vulnerability in the third quartile of impact scores (at 7.90) was two times as likely to receive CERT/CC attention as a vulnerability in the first quartile of impact scores (at 2.90). The diminished impact of CVSS scores likely reflects the fact that CVSS scores are imperfect measures of vulnerability severity [66].

Finally, companies with more experience handling vulnerabilities are less likely to have CERT/CC coordination, which validates **H10**. In 2019, a vulnerability in the third quartile of mean vendor age (18.5 years) is 43% as likely to receive CERT/CC coordination as a vulnerability in the first quartile (7 years).

6 Discussion

This section discusses what has changed over time, the role of cooperation in cybersecurity, and methodological issues.

Developments The three proxies for the amount of information sharing are the most reliable longitudinal indicators given they are independent of CVE issuance or labelling. The strongest trend in these proxies is that cases are now resolved faster despite involving more emails and recipients (see

Figure 3), which suggests increased responsiveness among CERT/CC and its constituents.

Section 4.2 suggests information sharing volume and participation are heavy tailed, which means the majority of information is shared about a minority of vulnerabilities. This is unlikely down to intrinsic properties of the vulnerabilities, such as those captured by CVSS, but rather because of how the software products are deployed in the world, specifically the winner takes all dynamics of software markets [67]. Tuverson and Ruffle [68] note that certain IT vendors are "systemically important technology entities" for whom a security bug could impact thousands of businesses.

Indeed this can be seen in comparing the effect of proxies for severity on information sharing volume (Table 3) with the effect on CERT/CC's decision to coordinate (Table 4). While vulnerabilities with higher CVSS impact scores and publicly available exploit codes are more likely to become the focus of CERT/CC attention, they do not lead to more information sharing volume. In contrast, upstream supply chain vulnerabilities do seem more difficult to coordinate. Communications about these bugs appear to be more protracted than communications about other vulnerabilities, ceteris paribus, because it takes longer to understand their full scope and all of the end-users they afflict. Indeed, this is consistent with multiple noted supply chain attacks.

For instance, vulnerability coordinators had to scour the LinkedIn profiles of software engineers to figure out which companies may have been impacted by the Treck IP Ripple20 vulnerability; many security engineers stated on their resumes that they had experience "integrating" the open source library with their company's product [69]. But such ad-hoc methods are very inefficient, particularly for widespread supply chain vulnerabilities. Some open-source security flaws, such as the Amnesia:33 vulnerabilities, impact millions of devices. A software bill of materials (SBOM) [70] that clearly states the open source libraries and third-party vendors that are included in a product would reduce vulnerability managers' reliance on ad-hoc coordination methods and expedite the coordination and remediation process for upstream vulnerabilities. The Biden administration's recent executive order, which requires all federal software vendors to maintain publicly available SBOMs, is a step in the right direction.

Turning to the NVD, the increasing number of CVE IDs over time seen in Figure 2 creates significant work in terms of validating and processing new vulnerabilities. MITRE delegated some of this work to CVE Numbering Authorities (CNAs) who are authorised to issue CVE IDs. The number of CNAs grew from 22 in 2016 to over 100 in 2019. However, this does not explain how so many more new vulnerabilities are being discovered. Figure 5 shows an increasing share of CVE IDs now impact the largest vendors. This could be due to bug bounties offered by dominant vendors [7, 8].

This raises the question of how institutions tasked with coordinating information sharing should respond. We suggest three approaches: prioritisation, delegation, and automation. Delegation involves supporting external coordination capacity and transferring responsibility, which **H10** suggests has been successful. This is most appropriate for those vulnerabilities affecting dominant vendors who have the resources to establish PSIRTs. Automation is more appropriate when the lack of a single dominant actor means the same misconfiguration is made by many developers, such as when thousands of Android applications misconfigured SSL certificates in VU#582497. Automation allowed CERT/CC to coordinate a case involving over 27 000 emails. Prioritisation is necessary when the number of vulnerabilities remaining after automation and delegation exceeds the coordinator's resources. Targeting resources in light of this reality remains an open problem [47].

Competition vs cooperation Evidence of increased information sharing associated with multi-vendor vulnerabilities (**H3**) opens up the question of whether competition or coordination drives the faster patch release times observed by Temizkan et al. [49]. Their attribution of this observation to competition may be premature given the research designs only observes outcomes (patch release timing) and not the mechanisms driving it. Equally, we should not place too much emphasis on our own evidence given we do not know the content of this information. However, the mechanism of Temizkan et al. [49] relies on consumer-driven incentives for vendors to produce timely patches. Such a mechanism is inconsistent with consumer practices in which patches are inconsistently applied even when they are available [35, 36, 38, 71] and would also represent a rare instance in which markets reward security [72].

More broadly, the present case-study shows that cooperative cybersecurity information sharing is a viable alternative to market incentives, at least under certain circumstances. This observation is consistent with some older models of the vulnerability ecosystem [26]. Deriving generalised success factors is difficult, but it is worth emphasising one finding—establishing a publicly funded cybersecurity institution (CERT) did not crowd out private institutions in the long term. This can be seen in the evidence in support of **H10**, which suggests CERT prioritises younger vendors, and also the establishment of PSIRTs by companies like Microsoft and Apple. Qualitative work emphasises how CERT scaled internationally by building trusted infrastructure [27] and human capital in other countries [73]. This casts doubt over tropes like public services breeding dependency and foreign nations free-riding on the investments of the United States, rather initial public investments can bootstrap norms that lead to self-perpetuating positive outcomes.

Limitations Analysing email meta-data provides aggregate insights at the cost of nuance. We have no way of identifying which purpose a given email serves, which leads to the simplifying assumption that all emails contain an equal amount of information. In actuality, emails are highly heterogeneous: some contain novel findings like the existence of a vulnerability while others are mere formalities or pleasantries. We grouped messages by vulnerability and assumed the distribution of information content is relatively constant.

A separate issue is data quality. The meta-data is based on reliable information about when and to whom emails were sent. In contrast, CVE information is based on assessments that depend on the "assessor's knowledge and skills" [74]. Other researchers have demonstrated inaccuracies in the National Vulnerability Database [75]. We are exposed to all such issues because we did not independently validate information pulled from the NVD.

A third problem results from conducting longitudinal research [76]. Figures 2–5 all display flux but its difficult to disentangle changes in the world (e.g. the proliferation of bug bounties or IoT products) from changes in reporting procedures (e.g. the proliferation of CNAs or changes in CERT/CC policy). Statistical designs can help mitigate this, such as including time-based fixed effects in Section 5.1 or running a separate regression for each year in Section 5.2. However, the residual distortions will be reflected in our results. For example, the coefficients in Table 4 are partly determined by CERT/CC decisions but also by differences in the population of published CVE IDs across years. We could have more confidence in research results if a statistical agency aimed to control for these differences, as the US's Cyber Solarium Commission proposes.

7 Conclusion

CERT/CC represent a case-study demonstrating that voluntary cybersecurity information sharing *is* possible. Our longitudinal case-study of 430K emails sent across nearly 30 years show: (i) the number of vulnerabilities exceeds the number of emails in some months (Figure 2); (ii) the information sharing window per vulnerability has fallen from a mean of nearly a year in the 1990s to less than 50 days (Figure 3); and (iii) the most information is shared about vulnerabilities affecting application products and the most popular vendors (Figure 4 and 5). Further, the distribution of information sharing volume and recipients is heavy-tailed (Figure 6).

In exploring the factors associated with information sharing volume, we discovered that more information is shared via email about vulnerabilities for which multiple vendors are affected (**H3**) and when communications begin before the vulnerability is made public (**H5**). Finally, we analysed which vulnerabilities CERT/CC focus coordination efforts on in the years following 2012. We find that CERT/CC is more likely to coordinate vulnerabilities that affect more vendors (**H7**), affect less established vendors (**H10**), and have public exploits available (**H9**).

Acknowledgements

We are grateful to Rainer Böhme, Max Hils, and three anonymous reviewers for their feedback on our work. Daniel Woods is funded by the European Commission's call H2020-MSCA-IF-2019 under grant number 894700.

References

[1] Vector Guo Li, Matthew Dunn, Paul Pearce, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. Reading the tea leaves: A comparative

analysis of threat intelligence. In *28th USENIX Security Symposium (USENIX Security 19)*, pages 851–867, 2019.

[2] Xander Bouwman, Harm Griffioen, Jelle Egbers, Christian Doerr, Bram Klievink, and Michel van Eeten. A different cup of ti? the added value of commercial threat intelligence. In *29th USENIX Security Symposium*, 2020.

[3] Sam Ransbotham, Sabyasachi Mitra, and Jon Ramsey. Are markets for vulnerabilities effective? *ICIS 2008 Proceedings*, page 24, 2008.

[4] Matthew Finifter, Devdatta Akhawe, and David Wagner. An empirical study of vulnerability rewards programs. In *USENIX Security Symposium*, pages 273–288, 2013.

[5] Mingyi Zhao, Jens Grossklags, and Peng Liu. An empirical study of web vulnerability discovery ecosystems. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, pages 1105–1117. ACM, 2015.

[6] Thomas Maillart, Mingyi Zhao, Jens Grossklags, and John Chuang. Given enough eyeballs, all bugs are shallow? Revisiting Eric Raymond with bug bounty programs. *Journal of Cybersecurity*, 3(2):81–90, 2017.

[7] Thomas Walshe and Andrew Simpson. An empirical study of bug bounty programs. In *2020 IEEE 2nd International Workshop on Intelligent Bug Fixing (IBF)*, pages 35–44. IEEE, 2020.

[8] Kiran Sridhar and Ming Ng. Hacking for good: Leveraging HackerOne data to develop an economic model of bug bounties. *Journal of Cybersecurity*, 7(1):tyab007, 2021.

[9] L Jean Camp and Catherine Wolfram. Pricing security. In *Proceedings of the CERT Information Survivability Workshop*, pages 31–39, 2000.

[10] David McKinney. Vulnerability bazaar. *IEEE Security & Privacy*, 5(6):69–73, 2007.

[11] Luca Allodi and Fabio Massacci. Comparing vulnerability severity and exploits using case-control studies. *ACM Transactions on Information and System Security (TISSEC)*, 17(1):1, 2014.

[12] Luca Allodi, Marco Corradin, and Fabio Massacci. Then and now: On the maturity of the cybercrime markets the lesson that black-hat marketeers learned. *IEEE Transactions on Emerging Topics in Computing*, 4(1):35–46, 2015.

[13] Alain Mermoud, Marcus Matthias Keupp, Kévin Huguenin, Maximilian Palmié, and Dimitri Percia David. To share or not to share: a behavioral perspective on human participation in security information sharing. *Journal of Cybersecurity*, 5(1):tyz006, 2019.

[14] Lawrence A Gordon, Martin P Loeb, and William Lucyshyn. Sharing information on computer systems security: An economic analysis. *Journal of Accounting and Public Policy*, 22(6):461–485, 2003.

[15] Felix Antonio Barrio Juárez, Raul Riesco Granadino, Aurélio Blanquet, and et. al. Information sharing and analysis centres (ISACs). Technical report, ENISA, 2017.

[16] Adam Zibak and Andrew Simpson. Cyber threat information sharing: Perceived benefits and barriers. In *Proceedings of the 14th International Conference on Availability, Reliability and Security*, pages 1–9, 2019.

[17] Esther Gal-Or and Anindya Ghose. The economic incentives for sharing security information. *Information Systems Research*, 16(2):186–208, 2005.

[18] Vasileios Mavroeidis and Siri Bromander. Cyber threat intelligence model: An evaluation of taxonomies, sharing standards, and ontologies within cyber threat intelligence. In *2017 European Intelligence and Security Informatics Conference (EISIC)*, pages 91–98. IEEE, 2017.

[19] Roberto Garrido-Pelaz, Lorena González-Manzano, and Sergio Pastrana. Shall we collaborate? a model to analyse the benefits of information sharing. In *Proceedings of the 2016 ACM on Workshop on Information Sharing and Collaborative Security*, pages 15–24, 2016.

[20] Stefan Laube and Rainer Böhme. Strategic aspects of cyber risk information sharing. *ACM Computing Surveys (CSUR)*, 50(5):1–36, 2017.

[21] Chad Heitzenrater, Rainer Böhme, and Andrew Simpson. The days before zero day: Investment models for secure software engineering. In *Workshop on the Economics of Information Security*, 2016.

[22] Ari Schwartz, Rob Knake, Belfer Center for Science, and International Affairs. *Government's Role in Vulnerability Disclosure: Creating a Permanent and Accountable Vulnerability Equities Process.* Harvard Kennedy School, Belfer Center for Science and International Affairs, 2016.

[23] Tristan Caulfield, Christos Ioannidis, and David Pym. The US vulnerabilities equities process: an economic perspective. In *International Conference on Decision and Game Theory for Security*, pages 131–150. Springer, 2017.

[24] Charlie Miller. The legitimate vulnerability market: Inside the secretive world of 0-day exploit sales. In *Workshop on the Economics of Information Security*, 2007.

[25] Rainer Böhme. A comparison of market approaches to software vulnerability disclosure. In *International Conference on Emerging Trends in Information and Communication Security*, pages 298–311. Springer, 2006.

[26] Karthik Kannan and Rahul Telang. Market for software vulnerabilities? think again. *Management Science*, 51(5):726–740, 2005.

[27] Rebecca Slayton and Brian Clarke. Trusting infrastructure: The emergence of computer security incident response, 1989–2005. *Technology and Culture*, 61(1):173–206, 2020.

[28] Mingyi Zhao, Aron Laszka, and Jens Grossklags. Devising effective policies for bug-bounty platforms and security vulnerability discovery. *Journal of Information Policy*, 7:372–418, 2017.

[29] Andy Ozment. Bug auctions: Vulnerability markets reconsidered. In *Workshop on the Economics of Information Security*, 2004.

[30] Andreas Kuehn and Milton Mueller. Analyzing bug bounty programs: An institutional perspective on the economics of software vulnerabilities. In *42nd Research Conference on Communication, Information and Internet Policy*, 2014.

[31] Cormac Herley and Dinei Florêncio. Nobody sells gold for the price of silver: Dishonesty, uncertainty and the underground economy. In *Economics of information security and privacy*, pages 33–53. Springer, 2010.

[32] Jason Franklin, Adrian Perrig, Vern Paxson, and Stefan Savage. An inquiry into the nature and causes of the wealth of internet miscreants. In *ACM conference on Computer and communications security*, pages 375–388, 2007.

[33] Hasan Cavusoglu, Huseyin Cavusoglu, and Jun Zhang. Security patch management: Share the burden or share the damage? *Management Science*, 54(4):657–670, 2008.

[34] Hiroyuki Okamura, Masataka Tokuzane, and Tadashi Dohi. Optimal security patch release timing under non-homogeneous vulnerability-discovery processes. In *International Symposium on Software Reliability Engineering*, pages 120–128. IEEE, 2009.

[35] Stefan Frei, Martin May, Ulrich Fiedler, and Bernhard Plattner. Large-scale vulnerability analysis. In *Proceedings of the SIGCOMM workshop on Large-scale attack defense*, pages 131–138, 2006.

[36] Muhammad Shahzad, Muhammad Zubair Shafiq, and Alex X Liu. A large scale exploratory analysis of software vulnerability life cycles. In *2012 34th International Conference on Software Engineering (ICSE)*, pages 771–781. IEEE, 2012.

[37] Didier Sornette, Thomas Maillart, and Giacomo Ghezzi. How much is the whole really more than the sum of its parts? 1 1= 2.5: Superlinear productivity in collective group actions. *Plos one*, 9(8):e103023, 2014.

[38] Frank Li and Vern Paxson. A large-scale empirical study of security patches. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2201–2215, 2017.

[39] Valentina Piantadosi, Simone Scalabrino, and Rocco Oliveto. Fixing of security vulnerabilities in open source projects: A case study of Apache HTTP server and Apache tomcat. In *2019 12th IEEE Conference on Software Testing, Validation and Verification (ICST)*, pages 68–78. IEEE, 2019.

[40] Sadegh Farhang, Mehmet Bahadir Kirdan, Aron Laszka, and Jens Grossklags. An empirical study of Android security bulletins in different vendors. In *Proceedings of The Web Conference 2020*, pages 3063–3069, 2020.

[41] Marie Vasek and Tyler Moore. Do malware reports expedite cleanup? An experimental study. In *Workshop on Cyber Security Experimentation and Test*, Bellevue, WA, 2012. USENIX.

[42] Ben Stock, Giancarlo Pellegrino, Christian Rossow, Martin Johns, and Michael Backes. Hey, you have a problem: On the feasibility of large-scale web vulnerability notification. In *USENIX Security Symposium*, pages 1015–1032, 2016.

[43] Frank Li, Zakir Durumeric, Jakub Czyz, Mohammad Karami, Michael Bailey, Damon McCoy, Stefan Savage, and Vern Paxson. You've got vulnerability: Exploring effective vulnerability notifications. In *USENIX Security Symposium*, pages 1033–1050, 2016.

[44] Orcun Cetin, Carlos Ganan, Maciej Korczynski, and Michel van Eeten. Make notifications great again: learning how to notify in the age of large-scale vulnerability scanning. In *Workshop on the Economics of Information Security*, 2017.

[45] Debabrata Dey, Atanu Lahiri, and Guoying Zhang. Optimal policies for security patch management. *INFORMS Journal on Computing*, 27(3):462–477, 2015.

[46] Jay Jacobs, Sasha Romanosky, Idris Adjerid, and Wade Baker. Improving vulnerability remediation through better exploit prediction. In *Workshop on the Economics of Information Security*, Boston, MA, June 2019.

[47] Jonathan M Spring, Eric Hatleback, Allen D. Householder, Art Manion, and Deana Shick. Prioritizing vulnerability response: A stakeholder-specific vulnerability categorization. In *Workshop on the Economics of Information Security*, Brussels, Belgium, December 2020.

[48] Ashish Arora, Ramayya Krishnan, Rahul Telang, and Yubao Yang. An empirical analysis of software vendors' patch release behavior: impact of vulnerability disclosure. *Information Systems Research*, 21(1):115–132, 2010.

[49] Orcun Temizkan, Ram L Kumar, Sungjune Park, and Chandrasekar Subramaniam. Patch release behaviors of software vendors in response to vulnerabilities: An empirical analysis. *Journal of Management Information Systems*, 28(4):305–338, 2012.

[50] John D Howard. An analysis of security incidents on the internet 1989-1995. Technical report, Carnegie-Mellon Univ Pittsburgh PA, 1997.

[51] Ashish Arora, Chris Forman, Anand Nandkumar, and Rahul Telang. Competitive and strategic effects in the timing of patch release. In *Workshop on the Economics of Information Security*, 2006.

[52] Sandor Boyson. Cyber supply chain risk management: Revolutionizing the strategic control of risk management systems. *Technovation*, 34:342–352, 2014.

[53] Allen D Householder, Garret Wassermann, Art Manion, and Chris King. The CERT guide to coordinated vulnerability disclosure. Technical Report CMU/SEI-2017-SR-022, Carnegie Mellon University, Pittsburgh, PA, United States, 2019.

[54] Richard Clarke and Rob Knake. *The fifth domain: defending our country, our companies, and ourselves in the age of cyber threats.* Penguin Press, 2020.

[55] Allen D Householder, Jeff Chrabaszcz, Trent Novelly, David Warren, and Jonathan M Spring. Historical analysis of exploit availability timelines. In *Cyber Security Experimentation and Test*, Virtual Conference, August 2020. USENIX.

[56] CERT/CC. Report a vulnerability. `https://web.archive.org/web/20181203184600/https://kb.cert.org/vuls/report/`, 2018. Accessed Jun 14, 2021.

[57] Garret Wassermann and Madison Oliver. Guidelines for requesting coordination assistance. `https://vuls.cert.org/confluence/display/Wiki/Guidelines+for+Requesting+Coordination+Assistance`, August 2015. Accessed Jun 14, 2021.

[58] Emily Sarneso and Art Manion. CERT/CC releases VINCE software vulnerability collaboration platform. `https://www.sei.cmu.edu/news-events/news/article.cfm?assetid=641759`, June 2020. Accessed Jun 14, 2021.

[59] Roman Danyliw. Data sharing: Lessons learned by the CERT/CC and the CERT/NetSA groups. In *FloCon*, Crystal City, VA, July 2004.

[60] Valentin Jean Marie Manès, HyungSeok Han, Choongwoo Han, Sang Kil Cha, Manuel Egele, Edward J Schwartz, and Maverick Woo. The art,

science, and engineering of fuzzing: A survey. *Transactions on Software Engineering*, 2019.

[61] Cristiano Calcagno, Dino Distefano, Jérémy Dubreil, Dominik Gabi, Pieter Hooimeijer, Martino Luca, Peter O'Hearn, Irene Papakonstantinou, Jim Purbrick, and Dulma Rodriguez. Moving fast with software verification. In *NASA Formal Methods*, number 9058 in LNCS, pages 3–11. Springer, 2015.

[62] Jeff Alstott, Ed Bullmore, and Dietmar Plenz. powerlaw: a python package for analysis of heavy-tailed distributions. *PLOS ONE*, 9(1):e85777, 2014.

[63] Aaron Clauset, Cosma Rohilla Shalizi, and Mark EJ Newman. Power-law distributions in empirical data. *SIAM review*, 51(4):661–703, 2009.

[64] Søren Asmussen. *Applied probability and queues*, volume 51. Springer Science & Business Media, 2008.

[65] Lada A Adamic, Bernardo A Huberman, AL Barabási, R Albert, H Jeong, and G Bianconi. Power-law distribution of the world wide web. *Science*, 287(5461):2115–2115, 2000.

[66] Jonathan Spring, Eric Hatleback, Allen Householder, Art Manion, and Deana Shick. Time to change the CVSS? *Security & Privacy*, 19(2):74–78, 2021.

[67] Carl Shapiro, Shapiro Carl, Hal R Varian, et al. *Information rules: A strategic guide to the network economy.* Harvard Business Press, 1998.

[68] Michelle Tuveson and Simon Ruffle. Diversity is the way to avoid cyber collapse. *Financial Times*, April 2014.

[69] Allan Friedman. Software bill of materials: Transparency in the software supply chain, Jan 2021.

[70] Michelle Jump and Art Manion. Framing software component transparency: Establishing a common software bill of material (SBOM). Technical report, National Telecommunications and Information Administration, Washington, DC, November 2019.

[71] William A Arbaugh, William L Fithen, and John McHugh. Windows of vulnerability: A case study analysis. *Computer*, 33(12):52–59, 2000.

[72] Ross Anderson and Tyler Moore. The economics of information security. *Science*, 314(5799):610–613, 2006.

[73] Leonie Maria Tanczer, Irina Brass, and Madeline Carr. CSIRTs and global cybersecurity: How technical experts support science diplomacy. *Global Policy*, 9:60–66, 2018.

[74] Luca Allodi, Marco Cremonini, Fabio Massacci, and Woohyun Shim. Measuring the accuracy of software vulnerability assessments: experiments with students and professionals. *Empirical Software Engineering*, 25(2):1063–1094, 2020.

[75] Afsah Anwar, Ahmed Abusnaina, Songqing Chen, Frank Li, and David Mohaisen. Cleaning the nvd: Comprehensive quality assessment, improvements, and analyses. *arXiv preprint arXiv:2006.15074*, 2020.

[76] Eric Jardine. Taking the growth of the internet seriously when measuring cybersecurity. *Researching Internet Governance: Methods, Frameworks, Futures*, 2020.

Appendix

Table 5: Robustness Test: Volume of emails for years 2012-2019 with an exploit variable

	Model 1
impact	-0.02^{***}
	(0.00)
exploit	-0.04
	(0.02)
vendors	-0.13^{***}
	(0.00)
prior	0.04^{***}
	(0.01)
os	0.23^{***}
	(0.04)
mix	1.89^{***}
	(0.07)
hardware	-0.24^{***}
	(0.07)
ageven	0.01^{***}
	(0.00)
osage	-0.02^{***}
	(0.00)
mixage	-0.16^{***}
	(0.01)
vulnage	-0.20^{***}
	(0.00)
hardwareage	0.02^{*}
	(0.01)
lrecp	0.27^{***}
	(0.00)

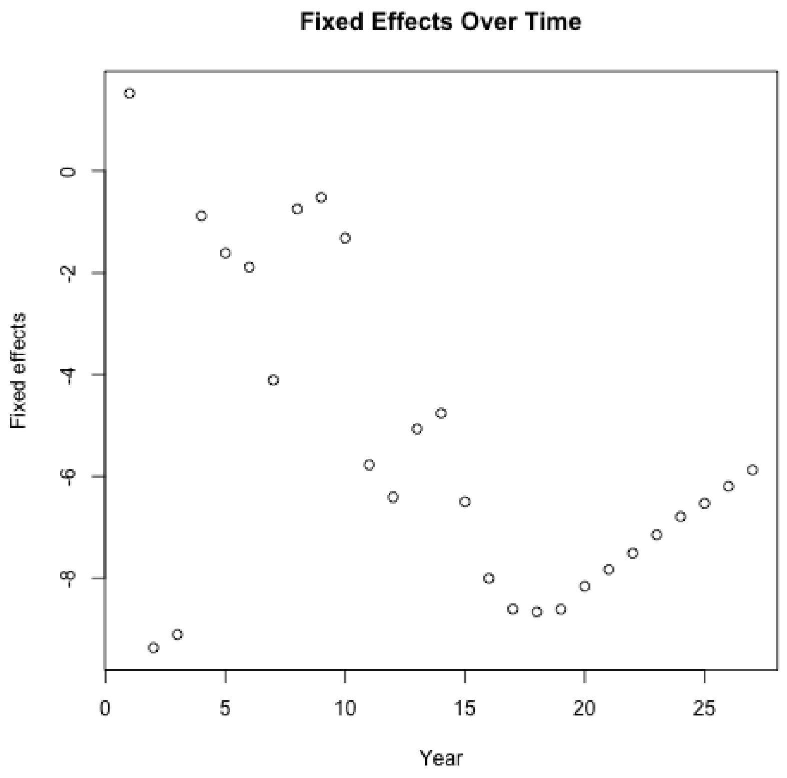
$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Table 6: Robustness check: Removing Right Vendor Outliers

	Year: 2015
exploit	2.61^{**}
	(0.88)
mean_age	0.94^{*}
	(0.02)
impact	1.07^{*}
	(0.04)
n_vendors	0.55^{*}
	(0.17)
Num. obs.	6470
Log Likelihood	-1127.04
Deviance	2254.08
AIC	2266.08
BIC	2305.50

$^{***}p < 0.001$; $^{**}p < 0.01$; $^{*}p < 0.05$

Figure 7: Time fixed effects.


Fixed Effects Over Time
Fixed effects
Year