

Confusing Value with Enumeration: Studying the Use of CVEs in Academia

Moritz Schloegel¹, Daniel Klischies², Simon Koch³, David Klein³, Lukas Gerlach¹, Malte Wessels³, Leon Trampert¹, Martin Johns³, Mathy Vanhoef⁴, Michael Schwarz¹, Thorsten Holz¹, Jo Van Bulck⁴

¹ CISPA Helmholtz Center for Information Security (first.last@cispa.de)

² Ruhr University Bochum (first.last@ruhr-uni-bochum.de)

³ TU Braunschweig (first.last@tu-braunschweig.de)

⁴ DistriNet, KU Leuven (first.last@kuleuven.be)

Abstract

Common Vulnerabilities and Exposures (CVE) IDs serve as unique identifiers for security-relevant bugs, facilitating clear communication and tracking of affected products. Originally intended solely for identification, the CVE system has faced increasing criticism due to the misconception that assigning a CVE implies a serious security issue. Notably, academic works on security vulnerabilities often claim CVEs, presumably to demonstrate the practical impact of their methods.

We systematically study the use of CVEs in academic papers to better understand the correlation of academic CVEs with real-world implications. To this end, we present the trends we identified through quantitative analysis, qualitative review of published papers, and a user survey. We observe a clear shift towards more frequent use of CVEs in academic papers over the last 25 years, especially in certain research areas. Our qualitative review of 1,803 CVEs claimed in papers published in the past five years reveals that 34 % have not been publicly confirmed or were disputed by the maintainers of the affected software, challenging the notion of real-world effects. Our survey of 103 academic reviewers and authors reveals widespread misconceptions about the CVE system and an explicit preference for reporting CVE numbers, but without indicating any implicit bias in the review process. We advise caution on using CVEs as a proxy for real-world impact and provide actionable recommendations for the academic security community and practitioners.

1 Introduction

The *Common Vulnerabilities and Exposures* (CVE) program is ran by MITRE [31] with the explicit goal of assigning unique identifiers to vulnerabilities in order to collect and catalog them in a standardized way. Today, CVEs are the single most important and rapidly growing catalog of vulnerabilities, with more than 250,000 CVE records (often abbreviated as CVEs) assigned between 1999 and 2024 [33] and an average of 164 new CVEs added per day in 2024 [16]. To coordinate

these efforts, MITRE allows companies and organizations to register as *CVE Numbering Authorities* (CNAs) for vulnerabilities in their products. If a product has no designated CNA, a *CVE Numbering Authority of Last Resort* (CNA-LR), such as MITRE, serves as a catch-all solution.

Although the CVE ecosystem has been in use for over 25 years, it has several inherent shortcomings and has faced increasing criticism in recent years [8, 14, 18, 43, 46, 53]. First, a single CVE does not always correspond to a single vulnerability [8]. This is especially worrisome for recent microarchitectural CPU vulnerabilities like Spectre, which are often assigned a single CVE. However, they require numerous software patches across the entire software stack, including CPU firmware, operating systems, and applications. Second, whether or not a bug receives a CVE is at the sole discretion of the respective CNA, without unified procedures strictly adhered to by all CNAs. For companies acting as CNA for their own products, it may be beneficial not to assign CVEs to create a false illusion of security [9]. Conversely, CNA-LRs without expert knowledge of the affected system typically maintain low standards for CVE assignments [18]. Third, the apparent ease of obtaining CVEs is exacerbated by the inherent difficulty of invalidating disputed CVEs [8, 14, 53]. The frustration with this situation has recently prompted several major open-source projects, including Linux [19] and Python [27], to register as CNAs for their own projects. Finally, CVEs have arguably outlived their original role as vulnerability identifiers and have become de-facto proxies for real-world impact and prestigious indicators of competence in the field. Indeed, prominent developers have observed that CVEs are “abused by security developers looking to pad their resumes” [8] and “used in ways that elevate their importance well beyond the level that makes sense given the amount of scrutiny that is apparently applied to them” [14], prompting to “ignore them” [8] and “not blindly trust the CVE system. It is full of cracks and bogus reports” [54].

Despite the aforementioned concerns, we observe that the academic security community embraces CVEs with relatively few reservations. Especially papers that present new tech-

niques for the (automated) identification of vulnerabilities often report many CVEs as a metric for newly found bugs in real-world software [46]. In an analysis of publicly available meta-reviews, we found that CVEs were explicitly listed in the acceptance reasons for “identifying an impactful vulnerability” in at least five out of 19 papers at IEEE S&P 2024 that claim CVEs. The common practice of including CVEs in submitted papers has even prompted explicit instructions to “not include full CVE identifiers in order to preserve the anonymity of the submission” in the IEEE S&P 2025 call for papers [1]. Although this instruction to blind CVE identifiers is intended to preserve anonymity, it highlights their frequent use in papers and their role beyond mere identification.

In this work, we are the first to conduct both a quantitative and a qualitative study on academia’s use of CVEs in published scientific papers, as well as a user-centered survey to assess the perception of CVEs among reviewers and authors. First, we quantitatively analyze all papers published at the leading top 4 A* security conferences (i.e., IEEE S&P, USENIX Security, ACM CCS, and ISOC NDSS). Our results show that the relative proportion of papers mentioning CVEs has risen sharply over the last two decades. Following this trend, we perform a qualitative, in-depth analysis of CVEs claimed in papers published in these venues over the past five years. In particular, we track the underlying bug reports for all 1,803 CVEs and manually check how the maintainers of the affected projects have responded to them. We assess whether they have acknowledged and fixed the vulnerability, ignored the report, or outright rejected it. Notably, we cannot locate a bug confirmation or fix for 34 % of the CVEs and find that 52 were explicitly disputed by the developers, which raises the question of why a CVE has been issued in the first place. Finally, to better understand the broader perception of CVEs in the academic community, we conduct a survey of 103 individuals who serve as reviewers or authors of academic papers. Our results indicate that the perceived meaning of CVEs has been decoupled from their original purpose of being a standardized, unique identifier. Moving forward, we discuss the underlying root causes and provide actionable recommendations aimed at a more accurate understanding and use of CVEs in academia.

Contributions. Our main contributions are as follows:

- We conduct a large-scale, quantitative analysis of CVE usage in A* venues, showing increasing prevalence.
- We qualitatively study the outcome of all CVEs reported in A* security papers over the past five years, highlighting several reoccurring and worrisome trends.
- We survey 103 academic authors and reviewers to capture the perception of CVEs within our community, identifying widespread misconceptions.
- We provide actionable recommendations on how to better align CVE usage in academic papers and decrease the negative perception among developers.

2 Demystifying CVEs: A Short Primer

What is colloquially referred to as a CVE is technically a *CVE ID*, a standardized, unique identifier for a vulnerability, formatted as *CVE-Year-Number*. Alongside the identifier, MITRE stores additional information in a CVE record, including the current state, affected systems, and links to security advisories [31]. After assignment, NIST then analyzes the CVE and assigns a *Common Vulnerability Scoring System* (CVSS) score to quantify its impact. A CVE record can have one of the following status:

- **Reserved:** A CVE ID was reserved for a bug that was not publicly disclosed yet. Thus, the CVE record contains no information (not even the CNA).
- **Published:** Once the bug becomes public, the full data of the CVE record is published.
- **Rejected:** A previously published CVE is now considered invalid and should no longer be used [40].

Specific tags may carry additional information. For instance, the *unsupported-when-assigned* tag indicates that the faulty software has reached end of life. The *disputed* tag is used when software maintainers contest the CVE assignment, similar to the *rejected* state, though MITRE is not fully convinced of the CVE’s invalidity.

Assignment Process and Actors. The process from the discovery of a bug to the assignment of a CVE involves several actors and depends on the product in which the bug was found. Each company or project can become a CNA, responsible for assigning CVEs to vulnerabilities in their products. If a bug is found in a project without a dedicated CNA, as is often the case for smaller open-source projects, the reporter can contact any CNA-LR, such as MITRE or CISA. These entities can assign CVEs for *any* project not covered by another CNA. In practice, any research group, such as Talos, or even individuals can become a CNA-LR [34].

Upon discovering a vulnerability, researchers have two ways to obtain a CVE. The typical flow is to responsibly disclose the bug to the project, which then handles the CVE assignment, either internally (if a project or vendor has their own responsible CNA) or by communicating with a CNA-LR. If projects have no responsible CNA, researchers can also directly contact a CNA-LR to request a CVE, which will be assigned with little to no verification. This can be exploited to receive CVEs without properly coordinating or disclosing the bug to the maintainers, as observed in practice [52, 54, 60].

Related Work. CVEs have a long history of attracting academic interest. For example, they have been used as datasets for testing the bug-finding capabilities of tools [65] and as the basis of vulnerability patch datasets [5, 15, 36]. To enable this, various works attempt to map CVEs to fixing commits [12, 23, 39, 44, 58, 61, 64], underscoring the importance

of CVEs for the community. Alexopoulos et al. [2] have investigated how long the underlying vulnerabilities live on in affected projects, and Pauley et al. studied the vulnerability life cycle [38]. Rather than targeting the underlying bugs and vulnerabilities, we focus on the use of the CVE identifiers.

Close to our work, Schloegel et al. [46] have surveyed fuzzing evaluations and, as part of their larger efforts, categorized how software maintainers have reacted to vulnerabilities found by fuzzing. This is similar to our qualitative analysis discussed in Section 4, albeit restricted to a subset of CVEs found by fuzzing papers. The findings are alarming: Only around 43% of the bugs that a fuzzer found and that were assigned a CVE have been fixed. This motivates our efforts for a larger-scale analysis of academic CVE usage across more disciplines. More generally, various individuals have criticized shortcomings in the CVE assignment process [14, 43] or pondered its overall effectiveness [8, 18]. Compared to these works, we are the first to systematically review the use of CVEs in academia and survey the community’s opinion. Regarding CVSS, Wunder et al. [63] have recently conducted a large-scale user survey on how CVSS scores are evaluated, finding widespread inconsistencies. Similarly, Spring et al. [50] criticize CVSS for its unjustified scoring algorithm and its misuse as a risk score by compliance organizations. These works are orthogonal to ours, as we focus purely on CVEs without considering CVSS scores.

In the wider perspective of meta-science, our study on the use of CVEs as a metric contributes to the ongoing discourse on the *science of security* [22, 28, 51], especially regarding the persistent challenges of establishing reliable security metrics and assessing the research value of attack papers [22]. Tangentially relevant to our work, prior meta-science studies within the security research community have explored issues such as transparency [26], biases [20], and the overall reliability of the peer review process [49]. Questionable CVEs may be partly enabled by a bias in the community, a lack of transparency in the assignment, and limited details about individual vulnerabilities. Hence, reviewers are often unable to critically assess the value of a specific CVE and may instead accept CVEs as proxies for real-world impact. More broadly, our study on CVEs connects to the wider trend of *metrification* in academia, where researchers and their work are increasingly assessed using quantitative indicators, such as the number of CVEs reported. While such metrics are undeniably convenient for reviewers, any metric can and will be gamed (*Goodhart’s law*). The dynamics and consequences of metrification in science have been critically examined in other academic fields [10, 47, 62].

3 Studying CVE Use in Scientific Papers

Dataset. To facilitate a large-scale *quantitative* analysis, we constructed a dataset comprising papers from the top 4 A* security conferences in the 2024 CORE ranking [37], namely

IEEE S&P, USENIX Security, ACM CCS, and ISOC NDSS. For this, we manually extracted all publication titles from the respective programs or proceedings between 1999 and 2024, corresponding to a total of 7,785 papers. We excluded any publications that were not papers (e.g., posters, messages from the chair, or presentations) and unusual paper formats (e.g., short papers). Since the CVE program was publicly launched in September 1999 [32], this dataset covers all academic papers at these venues that could potentially contain CVE identifiers. We automatically processed all paper PDFs using `pdftotext` [24], employing three automated checks to validate our dataset. First, we check if the title of the paper is contained in the extracted text, to eliminate PDFs that do not match the paper. Next, we check if the text extraction was successful by manually validating short documents and documents in which less than 35% of the content consists of the six most frequent letters in the English language. If `pdftotext` failed to extract the text of the document, which only occurred for 32 documents, we fall back to the OCR tool *Nougat* [6], a visual transformer model focusing on scientific documents. Note that our analysis may contain inaccuracies due to the use of automated tools.

We extract the CVEs mentioned in a particular scientific paper using a regular expression `([cC][vV][eE]\s*[-]?[sS]*\d{4}[sS]*[-]?[sS]*\d{4,7})`, which accounts for white spaces in the source material and inconsistent character choices. Our automated approach may under-approximate the number of CVEs due to CVEs contained in images or papers that do not include the specific CVE identifiers [7] or reference ranges of CVEs [66]. Overall, our dataset contains 1,167 papers (15.0%) that mention a CVE. They reference a set of 6,506 unique identifiers. One CVE can be referenced in multiple papers, e.g., when used in benchmarks [35, 45, 67] or due to the prominence of the underlying vulnerability. For example, [CVE-2014-0160](#), which represents the famous Heartbleed bug [13], is mentioned 51 times in our dataset.

As an additional validation step, we cross-check that our automated pipeline identifies at least one CVE for all papers that were manually found to claim CVE identifiers. Out of the 304 papers analyzed in the qualitative analysis of Section 4, 6 papers only mention the exact CVE numbers in a linked artifact. For the remaining papers, our pipeline fails to identify any CVEs in just 3 of them (<1%). In these instances, the text does not include an identifier prefixed with CVE.

Prevalence of CVEs. To analyze how the presence of CVEs in our dataset has changed over time, we calculate the percentage of papers in each year that mention a CVE. The results of this analysis are illustrated in Figure 1. CVEs were mentioned for the first time at USENIX Security 2002. We find that the relative prevalence of CVEs has considerably increased over the years, with 18.7% of papers published between 2020 and 2024 mentioning a CVE, as opposed to only 10.7% of

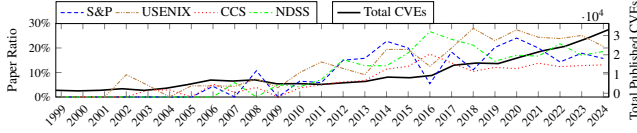


Figure 1: Percentage of papers per year that mention a CVE and the total number of CVEs published every year.

Table 1: Status of 6,506 academic vs. all 276,123 CVEs.

Status	Academic (%)		Overall (%)	
Published	6,143	(94.42%)	259,791	(94.09%)
Reserved	303	(4.66%)	—	—
Rejected	14	(<1%)	14,609	(5.29%)
Disputed/Unsupp.	45/1	(<1%)	1,280/443	(<1%)

the papers from 2010 to 2014. While we can observe this trend across all four venues, USENIX Security consistently has more papers mentioning CVEs over the entire period. Between 1999 and 2024, 20.3% of the papers published at USENIX Security mentioned a CVE. In the same period, 10.4% of CCS, 14.4% of IEEE S&P, and 15.2% of NDSS publications mentioned a CVE. We also analyzed where CVEs are mentioned within papers, as this may provide insights into their potential use. Our analysis reveals that the vast majority of CVEs are cited in the references or the appendix. Additional details can be found in Appendix B.

Status of CVEs. Table 1 compares the status (cf. Section 2) of the 6,506 CVEs in our dataset vs. all 276,123 CVEs published before January 9, 2025 (assigned to 1999–2024). Out of the analyzed CVEs used in academic papers, the vast majority (94.4%) are in the *published* state. A fraction of CVEs has been rejected or disputed, indicating explicit disagreement by the project maintainers. Notably, CVEs referenced by academic papers are less frequently rejected than CVEs overall. The remaining 4.7% of the mentioned CVEs are reserved, hinting at an uncompleted disclosure process. Due to the lack of publicly available information, we cannot compare this to the overall ecosystem. Note that this analysis does not distinguish whether a CVE is only cited in a paper or whether it is part of its contributions, as any such analysis ultimately requires manual review. In our qualitative analysis in Section 4, we manually study a subset of CVEs in detail.

Research Area. In addition to the CVE details inside papers, we analyze whether CVEs correlate with research areas or topics. While some papers contain a keyword section detailing their research area, this is only the case for a minority of the papers. Therefore, we train a keyword extraction model for security papers based on KeyBERT [17]. Our model performs unsupervised clustering to extract topics from the abstracts of all papers in our dataset. To improve accuracy, we additionally

train the model on all abstracts of arXiv papers labeled *Cryptography and Security*. Overall, this dataset contains 38,553 abstracts of papers published between 1999 and 2024. We then compare the number of CVEs in our A* conference dataset with the paper topics predicted by our model.

Our results show that the 104 papers classified as related to the topic of fuzzing mention an average of 7.6 CVEs per paper, which is among the highest ratios observed in our dataset. Similarly, microarchitectural security (59 papers) and browser security (100 papers) feature a relatively high ratio of 1.5 and 0.86 CVEs per paper, respectively. On the contrary, CVEs are far less prevalent in other research areas. The 101 papers classified as adversarial machine learning and backdoors, for example, contain 0.05 CVEs per paper, and 66 papers related to differential privacy do not contain a single CVE. We make similar observations for different subfields of cryptography and privacy. Our observations hint at the differing importance of CVEs depending on the research area.

Takeaway #1: CVE usage in academia is steadily increasing, especially in certain security research areas.

4 Qualitative Analysis of CVE Reception

We now *qualitatively* study the outcome of CVEs.

Collection. To contain manual effort, we limit our dataset to papers published at the four leading security venues between 2020 and 2024. We then conduct a keyword search to find papers containing CVEs, yielding 304 papers. For each of these, we manually extract all CVE IDs and determine the ones that are *claimed*, i.e., a result of the paper’s contributions. We discard those that are simply *used*, such as for benchmarking purposes [21, 48, 56, 57]. To ensure that our collection of CVE IDs is exhaustive, we also automatically extract CVE IDs using regular expressions and cross-check them against the manually extracted CVEs. In total, we collect 1,803 claimed CVEs over the 304 papers.

Labeling. Our goal is to identify the outcome of the bug associated with the CVE ID and to study maintainer agreement. To do so, we identify the following seven outcomes:

Fixed (F): The issue has been rectified.

Confirmed (C): The issue has not been *fixed*, but project maintainers have confirmed it.

Reserved (R): The CVE record is still being withheld, such that we can extract no further information.

Unsupported (U): We see no maintainer reaction, but the software appears unsupported, as it has no visible activity.

No-info (N): Publicly available information does not allow the assessment of the outcome. This differs from *reserved* CVEs, where, by design, no information is available. Here, the absence of (sufficient) information is involuntary.

Ignored (I): The bug report was ignored, yet the software has seen further updates or commits.

Opposed (O): Project maintainers have disputed whether the issue is a bug or whether it has a security impact (this may not necessarily include an official dispute/reject via MITRE’s system), opposing the assignment of a CVE.

These labels allow us to distinguish between outcomes where both sides agreed (*F, C*), where we cannot judge the agreement (*R, U*), and where the project maintainers did not agree (*I, N*) or outright disagreed with the CVE assignment (*O*).

To label CVEs, we rely on four independent researchers, who are experts in the field of systems security and have worked with vulnerabilities for between four and six years. These researchers individually assess the outcome of each CVE, with every CVE being independently assessed by at least two individuals. This double coding ensures robustness and consistency of our labeled data. For our labeling process, we leverage the official CVE record and its linked resources, in particular linked project repositories or bug tracker. Where the CVE record contains insufficient information, we also conduct Google searches. Our labeling was carried out between mid-2023 and the end of 2024. To resolve conflicts for misaligned labels, we conducted a discussion among all four experts to reach a consensus.

Labeling Limitations. Our approach is inherently tied to the maintainers’ reaction, as we assume they are the authoritative source to judge if a CVE should be assigned. This has two risks: First, maintainers may choose *not* to assign CVEs to qualified bugs due to error or deliberate considerations. For example, a maintainer might be hesitant to assign CVEs in an attempt to reduce reputational damage. Second, we do not assess if assignments are reasonable (although we have, in fact, observed assignments we considered questionable albeit without maintainer opposition). Our analysis does not and cannot capture these two aspects, as we are in general not an authoritative source to judge if reports represent a valid, security-relevant bug.

4.1 Analysis Results

Overall, 304 papers claim 1,803 CVEs, with an average of 5.9 and a median of 3 CVEs. 102 papers claim a single CVE, whereas 54 papers claim ten or more CVEs, with two papers claiming 71 each. A breakdown per year and conference is given in Table 2. In line with the quantitative analysis from Figure 1, USENIX Security is the most popular venue, featuring 41 % of the claimed CVEs. From 2020 to 2024, the claimed CVEs have more than tripled.

In terms of outcomes, 63 % (1,135) of the CVEs have been fixed, and 3 % (54) have been confirmed. Thus, in 66 % of the cases, both the reporter and the maintainers agreed upon the CVE assignment. In some cases, we cannot assess the quality of the CVE: 15 % (274) of the CVEs are reserved, leaving

Table 2: Analyzed CVEs from 304 papers by year and venue.

	<i>F</i>	<i>C</i>	<i>R</i>	<i>U</i>	<i>N</i>	<i>I</i>	<i>O</i>	Sum
2024	224	25	175	40	74	2	19	559
2023	309	18	25	22	35	12	12	433
2022	230	4	41	39	19	6	5	344
2021	242	3	22	0	18	1	7	293
2020	130	4	11	0	15	5	9	174
<i>Total</i>	1,135	54	274	101	161	26	52	1,803
USENIX	497	24	77	47	58	18	21	742
SP	271	22	78	50	48	5	12	486
NDSS	220	3	79	0	39	2	2	345
CCS	147	5	40	4	16	1	17	230

F(ixed); *C*(onfirmed); *R*(eserved); *U*(nsupported); *N*(o-info); *I*(gnored); *O*(pposed)

us with no information. A further 5.6 % (101) is assigned to software that appears to be no longer supported. Depending on whether the software is still relevant, this may represent a benign or questionable assignment. Unsupported and reserved CVEs amount to 21 % of all CVEs. Finally, there are several CVEs of potentially questionable quality: In 8.9 % (161) of the cases, the CVE records contain no information that allows us to trace the outcome, and for 1.4 % (26), the project has seen further activity but ignored the bug report associated with the CVE. In these cases, we cannot judge whether project maintainers agree with the CVE assignment. For 2.9 % (52) of the CVEs we studied, the maintainers explicitly opposed the CVE assignment.

🔍 **Takeaway #2:** 63 % of the analyzed CVEs are fixed and 13 % are potentially problematic (*N, I, O*).

4.1.1 Unspecified and Reserved CVEs

Oftentimes, papers claim to have received CVEs but do not specify the IDs, defeating the purpose of unique identification. We observed 57 papers that claim a total of 981 CVEs without specifying IDs, with three papers alone claiming 402 CVEs. We cannot verify such claims. Similarly, we cannot analyze CVEs in the reserved state, which occurs 274 times across 57 papers. A reserved CVE can be sensible for recent papers, where the vulnerability is not addressed yet. However, 99 of reserved CVEs were assigned to older bugs (before 2024). For such older vulnerabilities, little reason exists to withhold details, especially considering that MITRE suggests publishing CVEs as soon as details are publicly known—which is the case once details are released in a paper [41].

4.1.2 CVE Numbering Authorities

As of January 2025, MITRE lists 436 CNA partners [34]. Two of these, MITRE itself and CISA, officially function as CNA-LR. 19 CERTs and 32 other entities can also assign CVEs for third-party code (if not already covered by another

Table 3: Analyzed CVEs per CNA. CNA-LRs (bold) are responsible for almost all opposed and ignored CVEs. For reserved CVEs, we do not know the CNA.

CNA	<i>F</i>	<i>C</i>	<i>R</i>	<i>U</i>	<i>N</i>	<i>I</i>	<i>O</i>	Sum
Mitre	402	6	?	19	142	21	36	626
Snyk	69	3	?	51	4	4	2	133
Red Hat	65	3	?	0	0	1	5	74
VulDB	0	0	?	31	6	0	6	43
Other CNA-LRs	133	9	?	0	6	0	0	148
Regular CNAs	466	33	?	0	3	0	3	505
<i>Total</i>	1,135	54	274	101	161	26	52	1,803

F(ixed); *C*(onfirmed); *R*(eserved); *U*(nsupported); *N*(o-info); *I*(gnored); *O*(pposed)

CNA), which we consider as CNA-LRs for this work, too. Across the 1,803 CVEs in papers, we observe 66 different CNAs, including six CERTs (from the Netherlands, Germany, Japan, Switzerland, Taiwan, and the US) and 11 CNA-LRs (MITRE, Snyk, GitHub, RedHat, GitLab, VulDB, Protect AI, and others). The top five CNAs in terms of CVEs assigned are MITRE (41%, 626), Snyk (8.7%, 133), RedHat (4.8%, 74), Microsoft (4.3%, 62), and Apple (4.1%, 62).

Table 3 lists the analyzed CVEs per CNA. Expectedly, MITRE, which is the number one CNA in terms of assigned CVEs, is also the CNA assigning most of the opposed or ignored CVEs. Across all CNAs, almost exclusively CNA-LRs have assigned such CVEs—which is barely surprising, as companies functioning as CNAs for their own products verify bugs before assigning a CVE. Only in three cases, ARM, Zephyr, and Android have retracted a previously assigned CVE. In contrast, CNA-LRs often assign CVEs without verification, possibly leading to opposition by project maintainers who disagree with the report or its security sensitivity.

🔍 **Takeaway #3:** CNA-LRs are almost exclusively the source of CVEs that project maintainers do not agree with.

4.1.3 Paper Topics

We manually classify the primary topic of each paper (using the conference track it was published in as well as title and abstract) and list the CVEs sorted by topic in Table 4. In line with the quantitative analysis of Section 3, we observe that fuzzing is the field where most CVEs are assigned, making up 38% of all assigned CVEs, followed by web security with 12%. Notably, almost all opposed CVEs are from papers that aim to find bugs, such as fuzzing, web security, or bug finding. This may indicate that researchers feel a need to demonstrate their tool’s capabilities and push for CVEs even against the will of the maintainers of the analyzed software.

🔍 **Takeaway #4:** Research focused on finding bugs is more likely to contain CVEs without agreement.

Table 4: Analyzed CVEs by paper’s primary topic.

Topic	#Paper	<i>F</i>	<i>C</i>	<i>R</i>	<i>U</i>	<i>N</i>	<i>I</i>	<i>O</i>	Sum
Fuzzing	82	458	18	94	39	41	13	31	694
Web security	27	125	3	17	56	14	6	4	225
Network security	28	41	2	96	1	33	1	0	174
Security analysis	21	95	5	13	0	7	0	9	129
Bug finding	14	88	3	4	1	8	1	2	107
Embedded security	15	45	1	18	0	21	0	0	85
HW side channels	37	35	11	6	0	12	0	3	67
Crypto	18	41	6	6	0	7	0	1	61
Kernel security	5	54	0	0	0	0	0	1	55
Mobile security	12	43	1	0	0	0	0	1	45
ReDoS	3	20	0	0	4	6	0	0	30
Wireless security	9	20	1	7	0	0	0	0	28
Privacy	5	13	0	0	0	3	0	0	16
TEEs	5	7	0	2	0	5	0	0	14
Measurements	1	4	2	7	0	0	0	0	13
Program analysis	2	8	1	0	0	2	0	0	11
Supply chain sec.	2	6	0	0	0	0	4	0	10
Blockchain	2	2	0	4	0	0	1	0	7
Censorship	1	0	0	0	0	1	0	0	1
Social issues	1	0	0	0	0	1	0	0	1
<i>Other topics</i>	14	30	0	0	0	0	0	0	30
<i>Total</i>	304	1,135	54	274	101	161	26	52	1,803

F(ixed); *C*(onfirmed); *R*(eserved); *U*(nsupported); *N*(o-info); *I*(gnored); *O*(pposed)

Comparison to Prior Work. Schloegel et al. [46] studied 339 CVEs claimed by 35 randomly selected fuzzing papers, finding that only 43 % of claimed CVEs have been fixed, compared to 63 % in our more general dataset. We attribute this difference to the different time frame (2018-2023), venues (they also include the software-engineering conferences ASE, ICSE, and FSE), and categorization methodology used. For example, they treat duplicates as opposed, while we conservatively mark them as fixed as long as a fix exists. The 304 CVEs we qualitatively analyzed included 98 of the 339 examined by Schloegel et al., whereas the remaining 241 CVEs fell out of the scope of our analysis (either because they were published before 2020 or did not appear at a security conference). Table 5 compares our dataset to theirs, mapping their categories to ours on a best effort basis. Our classification differs for six CVEs, three of which can be traced to different methodologies. In the other three cases, nuances made a difference: We could not confirm a fix (we: confirmed vs them: fixed), saw maintainer activity (we: ignored vs them: unsupported), or did not see explicit disagreement (we: ignored vs them: opposed). In any case, this comparison suggests that several older fuzzing papers or those published in general software engineering venues significantly contributed to the many unfixed CVEs observed by Schloegel et al. [46]. Indeed, we find that 54 out of the 69 CVEs that Schloegel et al. classified as ignored and that fall outside our analysis scope can be attributed to only three papers.

🔍 **Takeaway #5:** The high number of questionable CVEs observed in prior work is not generalizable and depends on the analyzed dataset, which is impacted by outliers.

Table 5: Analysis results by prior work [46] and ours.

	<i>F</i>	<i>C</i>	<i>R</i>	<i>U</i>	<i>N</i>	<i>I</i>	<i>O</i>	Sum
This work	76	1	2	4	1	6	8	98
Schloegel et al.	76	0	2	5	–	4	11	98

F(ixed); *C*(onfirmed); *R*(eserved); *U*(nsupported); *N*(o-info); *I*(gnored); *O*(pposed)

4.1.4 Assessing Impact

One of our goals is to investigate whether CVEs assigned to academic works actually increase the security of the applications studied or merely inflate the perceived impact and burden the developers who have to deal with them. We found a *positive* agreement (fixed, confirmed) for 1,189 CVEs, uncertain impact (unsupported, no-info, ignored) for 562 CVEs, and questionable impact for 52 CVEs. To further study papers with uncertain or questionable impact, we analyze outliers by looking at all papers that are three standard deviations above average, i.e., that are above 99% of all papers in terms of CVEs with uncertain or questionable impact.

To ensure a constructive debate and avoid finger-pointing, we anonymize the identifiers of these outlier CVEs and refrain from naming specific papers, authors, or institutions. Section 9 further documents our motivation behind this decision.

Questionable Impact. Our dataset contains four outliers, which we refer to as *Opposed-1,2,3,4*.

Opposed-1 claims 18 CVEs, 6 of which have been opposed. Out of these 6, 5 are officially disputed and 1 is officially rejected. MITRE notes complaints of multiple third-parties, stating these bugs do not qualify as security issues. Such a correction by MITRE is rare, as the dispute process is tedious, and the burden of proof is put onto the maintainers rather than the reporter. Out of the remaining CVEs, 4 are *reserved*, 2 are assigned to unsupported programs, and 6 were fixed. For two fixed CVEs, the discussions in the bug tracker do not clearly indicate if the issues were fixed on purpose or accidentally, as a maintainer refers to a prior commit or patch with comments such as “I think this was fixed by” or “I think this fixes it”. In the other four resolved cases, the developers are grateful and clearly reference the report in a commit.

Opposed-2 claims 10 CVEs, 3 of which are disputed. The first dispute pertains to a GitHub issue regarding a stack-based buffer underflow vulnerability. A developer assumes the report to be a false positive; the reporter refutes this thesis but does not provide further evidence. Developers then conclude the issue is not reproducible on their end. The second dispute concerns a presumed double-free issue, which the developer denies. The reporter responds and closes the issue without further discussion. Finally, the third dispute concerns a null-pointer dereference that the developers did not consider a vulnerability in the last message before closing the issue. Interestingly, all three issues were closed as invalid long before the paper could have been submitted, raising the question of

why the authors obtained CVEs and listed them in their paper. Beyond the 3 disputed CVEs, the paper had 1 fixed CVE, 4 in unsupported targets, and 2 ignored ones.

Opposed-3 claims a total of 61 CVEs, 5 of which are opposed. 3 are officially disputed with a note reporting that “the vendor position is that post-authentication issues are not accepted as vulnerabilities”. The remaining 2 face opposition by developers stating that “[w]e don’t consider a vulnerability anything that require administrator credentials to execute. Unfortunately there is no Filter to create CVE’s these days and people can publish whatever they want”. Out of the remaining CVEs, 15 are confirmed by the vendor, but all are accompanied by a public statement detailing that no updates will be released, as the affected software entered the end-of-life process. Similarly, 32 CVEs concern products that are end of life (since 2013 to 2019), and the vulnerability reports have not received a response from the vendor. MITRE even notes for 12 CVEs that “the vendor was contacted early about this disclosure but did not respond in any way”, a unique note we have not seen with other CVEs. As far as we can tell, the affected firmware was last updated in 2013, eleven years before the paper was published. Finally, we were unable to ascertain information for the remaining 9 CVEs.

Opposed-4 reports 11 CVEs, 6 of which are officially disputed. For 2, the vendor “does not agree that this is a security issue requiring a fix” and for the other 4 they argue the reported issues assume an unrealistic threat model, where “the user has to explicitly install a malicious app and [...] the Android platform provides fair warnings”. For the paper’s remaining CVEs, we were unable to obtain information in 4 cases, and 1 CVE is still reserved at the time of writing.

Uncertain Impact. Our dataset contains 5 outliers with uncertain impact, one of which (*Opposed-3*) we already discussed. Three outliers, namely *Uncertain-1*, *Uncertain-2*, and *Uncertain-3* have been recently published in 2024, and the majority (59/71, 23/33, 29/29) of their claimed CVEs are currently reserved. However, given the short time frame since publication, we do not consider this to be problematic, and a revisit of the CVEs after publication of the CVEs is required to fairly assess their quality. Thus, we believe it makes more sense to revisit these CVEs in the future.

This leaves us with only a single outlier not yet discussed: *Uncertain-4*. This paper has claimed 71 CVEs, 2 of which are officially rejected. The CNA withdrew both CVEs, as further investigation revealed that neither was a security issue. Beyond the rejected CVEs, the paper had 28 fixed CVEs, 2 ignored, 2 CVEs for which we were unable to find any information, and, strikingly, 37 CVEs that were claimed on unsupported projects.

Lessons Learned. Across all claimed CVEs, we find relatively few questionable assignments. We do, however, observe papers that are clear negative outliers regarding opposed

CVEs. Based on the timeline, the opposed CVEs, and even some of the accompanying unopposed CVEs, appear to have been reported in a way that suggests the primary intent may have been to obtain a CVE designation, rather than to address a pressing security issue. *Opposed-1*, for example, was accepted at a conference with earliest submission deadline in January 2024, but the last modified date for its officially rejected CVE was in June 2023. Hence, the authors were likely aware of the rejection at the time of submission. *Opposed-2* is arguably worse: The developers communicated the invalidity of the bug reports between late 2019 and early 2022, and the first possible submission deadline of the paper was in Summer 2022. Either case raises the question of why the CVEs were included in the publication. The remaining two outliers, *Opposed-3* and *Opposed-4*, also appear to prioritize CVE counts over improving software security and reporting vulnerabilities. Either publication reported security issues that are quite strongly rejected as not-a-vulnerability by the developers of the affected products. However, the authors of *Opposed-3* stated that their tool “discovered 79 unknown vulnerabilities on these 14 IoT devices and [we] reported them to the vendors, of which 61 have been assigned with CVEs”, giving the vendor disagreement no consideration. The authors of *Opposed-4* received their CVEs by reporting the vulnerabilities to the US Computer Emergency Response Team after reporting to the vendor as well. While the vendor responses to the reports are unknown, it is certain that they disputed the resulting CVEs. Finally, *Uncertain-4* also shows indicators that CVEs were obtained for their own sake, as 2 were rejected and only 27 out of 71 were positively received by the maintainers of the respective software.

4.2 Overarching Insights

Insufficient Threat Analysis. One frequent issue during our analysis was a lack of threat analysis by the reporter. Several publicly visible vulnerability reports contain some kind of tool output, be it fuzzing or static analysis, without any justification for how the found issue could be used to attack a system. Reporting recoverable exceptions in Java libraries as DoS vulnerabilities or simple wrapper functions for `exec` as being vulnerable to command injection are among the samples we observed. An example is [CVE-2020-28435](#), which reports on a command injection vulnerability in a JavaScript library that provides an abstraction to the `ffmpeg` command line interface. It exposes a function called `execute`, which directly passes its arguments to the shell. As the whole purpose of this function is to provide an API for the underlying `exec` functionality, the lack of sanitization is by design. Similarly, command-line interface tools, frequently trivial ones, are reported as containing command injection vulnerabilities.

Lack of Communication. Another frequent issue is a complete lack of communication from researchers. After initially

reporting their findings, e.g., via GitHub issues, they frequently leave the respective maintainers on their own. Maintainers then are often taken by surprise upon learning of newly assigned CVEs for their projects. See, for example, [CVE-2021-34141](#), where the NumPy maintainers conclude that the reporter “seem[s] like a known bad actor, lots of bogus CVEs and no response after that anymore” [29]. As another example, [CVE-2020-20665](#) was first reported in a GitHub issue. However, the reporter simply dumped output from Clang’s `LeakSanitizer` and claimed the memory leak looks like a double-free error. The maintainer acknowledged and fixed the memory leak but disagreed on the double free. The reporter did not discuss any security implications and did not claim that the bug was a vulnerability—still, a CVE was assigned and reported in a paper anyway. In another example, a ReDoS vulnerability was reported privately to a maintainer of a parsing library. In the patch, the maintainer thanked the reporters for their (responsibly disclosed) findings but explicitly stated that a “CVE will not be assigned for this vulnerability” [25]. Nevertheless, [CVE-2021-29060](#) was assigned and reported in a paper.

As all conferences mandate responsible disclosure, lack of communication seems like a worrying trend. At times, we encountered glaring cases where the information in the CVE report directly indicates that no disclosure has taken place, for example, for [CVE-2021-40893](#), where the reporter-provided reference specifies “Contacted maintainer?: No”.

Lack of Vendor Updates. A CVE entry can include multiple references that link to a technical issue description or a statement thereof, the affected product, a patch if available, or other additional resources. Unfortunately, not all types of resources are always provided. Usually, the CVE entry includes the technical description of the underlying vulnerability, as otherwise an assessment by affected third parties is impossible. However, we found multiple CVE entries that only contained references to a description of the vulnerability, without references to the vendor’s response and without an indication if the vulnerability was fixed. Some of these CVEs concerned implementation flaws, where the only vendor-related reference was a link to the vendor’s online webshop. Others were related to specification flaws, where the only vendor reference was a link to the vulnerable specification, but without a concrete vendor response, e.g., [CVE-2020-35473](#). All CVEs with a similar lack of vendor responses were assigned by MITRE.

The missing information may be due to a lack of a (public) response from the vendor or due to the CVE entry not having been updated after the disclosure. Regardless, the result is that others are often unable to determine if a vulnerability was fixed or confirmed, limiting the value of the CVE. For MITRE, the only requirement for a CVE to become public is the existence of a reference describing the vulnerability. A link to a public vendor response is not required.

Redundant CVEs. We observed cases where multiple CVEs were assigned to what was essentially a single bug. For example, this can occur when two crashing inputs were fixed by a single code change or when duplicated code, located just a few lines apart, was assigned CVEs. This also aligns with Schloegel et al.’s observation that multiple CVEs were requested for different functions in the call stack, with only one underlying bug [46]. The broader pattern appears to be a focus on creating as many CVEs as possible. In our analysis, we conservatively counted such CVEs as fixed—unless maintainers opposed the bug reports—even in cases where we considered CVE assignments questionable.

5 Survey of Academics

To better understand what our community thinks of CVEs and their use within academic papers, we conducted a survey of 103 individuals within the security community in early 2024. We randomly selected 200 people from all members that served on the 2023 Program Committees (PCs) of the top 4 security conferences (after removing conflicts of interests and members without a public email address). This sampling aims at ensuring active community members while limiting manual work needed to identify public contacts. Our sample of 103 responses suffices to capture results that are representative of the reviewer population (~750 individuals at top 4 in 2023) at a 95% confidence level and 10% margin of error. We contacted the selected persons via email, asking them to anonymously fill out our survey and share it within their respective groups. Our motivation was to sample a sufficient number of senior academics who have reviewed multiple papers, and thereby gained a unique overview of the process. Sharing the survey with the group aimed at capturing the opinion of junior researchers, such as PhD students, too.

Survey Structure. Our survey follows multiple goals, structured into five main parts. The first group of questions is asked before fully briefing the participants on the survey’s goal of measuring how CVEs are used and thought of in the community. Here, our goal is to capture a potential *implicit bias*.

Following this initial part, participants are briefed on the subject of the survey. Subsequent parts then (1) test the knowledge of the participants regarding CVEs, (2) capture their behavior as academic *authors*, (3) capture their assumptions as academic *reviewers*, (4) ask them how they believe the academic community should use CVEs, (5) perform an attention check and request consent to process the participant’s answers in accordance with best-practices for surveys, and (6) invite participants to optionally share further stories pertaining to CVE misuse or leave feedback. We list all content-related questions in Appendix A and upload the survey to our artifact. Honesty of participants’ responses is essential for the validity of a survey. To address this, we employ a number of

safeguards: We conduct an explicit attention check, asking if participants have responded truthfully. We use (few) repetitive questions (e.g., Q24 and Q30), and we manually check the data for bad patterns (e.g., participants that always selected the leftmost answer).

When evaluating statistical significance w.r.t. observations central to our conclusions, we apply the widely used and accepted standard threshold of $\alpha = 0.05$. In line with current recommendations [4], we also provide the concrete p-values as a continuous measure of (un)certainly.

5.1 Demographics and Domain Knowledge

103 participants have completed the survey, 102 of which declared to have had prior contact with CVEs; the one remaining participant was not shown additional questions. The majority of the participants are currently working in an academic context, with 68 participants declaring to be professors, followed by 10 PhD students and 4 postdoctoral researchers. 15 participants declared that they work in an industry position. We also asked the participants to select one or more fields of research that their work focuses on. From their answers, it becomes apparent that our participants come from a wide range of different disciplines in security, from software security (53 participants), security in the context of machine learning (22 participants), web security (21 participants), and 12 more areas. Almost all participants have experience as authors and reviewers at academic conferences and journals. Note that this implies participants may respond to both author and reviewer questions; any bias they may hold is probably carried over. We stress that this is a natural feature of *peer review*, where authors review the work of their peers. Half of our participants have actively requested CVEs, while the other half are at least familiar with the concept. This underscores the pervasive role of CVEs in the field of systems security. An overview of the demographics of the participants is shown in Figure 2, more details are provided in Table 7 in Appendix A.

5.2 Knowledge on CVEs

To assess our participants’ knowledge and perception of CVEs, we asked them several factual questions (cf. Appendix A.2) based on the official CVE documentation.

Purpose & Reliability of CVEs. When asked about the purpose of CVEs, almost every participant (95 %) answered correctly that CVEs are primarily intended to uniquely identify a vulnerability. However, we also observe that many participants see CVEs as a way to determine the validity and impact of a vulnerability: 21 participants agreed with the statement that “A CVE acknowledges that a bug has a high real-world impact”. Strikingly, 55 participants (54 %) believed that, if a CVE has been assigned, the underlying bug has been verified as a *mandatory* step in the CVE assignment process.

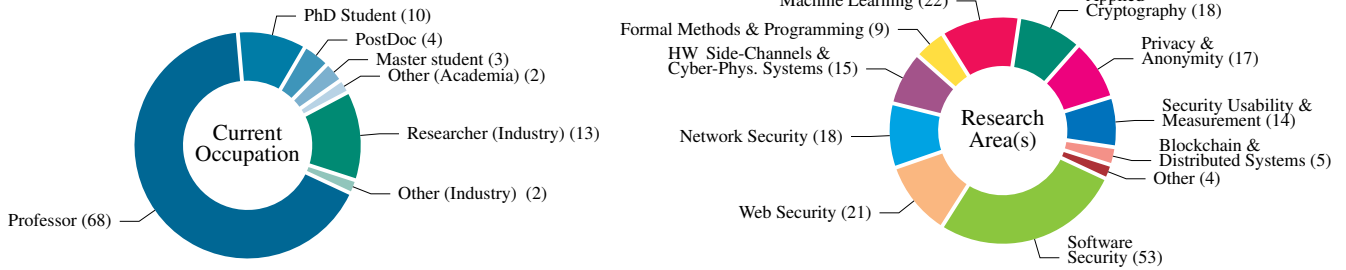


Figure 2: Survey participants’ demographics; detailed information is provided in Table 7 in Appendix A.

A slight minority of 47 participants correctly answered that validation of vulnerabilities is *not* a mandatory step in the CVE assignment process. We note that we cannot rule out misunderstandings regarding what *verification* implies in this context; however, we see a statistically significant correlation (at $p = 0.01 < 0.05 = \alpha$) of respondents believing CVEs to be verified and thinking they mattered for paper acceptance (Q15), when applying a Kruskal-Wallis test. Combined, these misconceptions may inspire an unwarranted confidence in the existence and impact of a vulnerability if a CVE is assigned. This creates the risk that the value of obtaining a CVE increases beyond uniquely identifying a bug to a perceived certification of correctness and real-world impact of the described vulnerabilities.

CVE Assignment Process. Overall, we observe that most participants generally know about the existence of CNAs and that CVEs can be assigned by MITRE (87 participants) and software vendors if they are registered as CNAs (86 participants). However, our results reveal that misconceptions are prevalent when examining the nuances of the process. For instance, 37 participants suggested that a CVE can be assigned by a third-party CNA *even if* the manufacturer of the affected product is a CNA. Such behavior would, in most cases, violate the CNA rules 7.3.1 and 4.1.2, as a CNA is only allowed to assign CVEs for products within their scope (usually products they operate or manufacture). Third parties, such as CNA-LRs, are only allowed to assign a CVE if there is *no* CNA for which the affected product is in scope. In addition, almost half of our participants (49 %) did not know that the information about which CNA has assigned the CVE is public.

Understanding the assignment process, the distinction between CNAs and CNA-LRs, and the availability of information is crucial for requesting a CVE from the appropriate CNA and assessing the credibility of a vulnerability already identified by a CVE.

Takeaway #6: There are widespread misconceptions about the purpose and reliability of CVEs and CNA processes. Most worryingly, 54 % of the participants incorrectly believe vulnerabilities are always verified as part of the CVE assignment.

5.3 Authors’ Perspective

Reasons for CVEs in Papers. First, we asked the participants to make forward-looking statements on whether and why they would mention CVEs in papers. 71 % of the participants agree or strongly agree that they try to obtain CVEs for vulnerabilities they discover, further supporting the hypothesis that CVE assignment is seen as desirable in the community. To assess the reasons behind this desirability, we asked our participants whether they believe that CVEs help demonstrate the practical impact of their work and that reviewers look for CVEs when evaluating their paper. 76 % of our participants somewhat agreed or strongly agreed that the presentation of practical relevance of research results is bolstered by the presence of CVEs. This strong consensus further emphasizes that CVEs are seen as a means to highlight the relevance and impact of a discovered vulnerability. In comparison, whether reviewers appreciate the presence of CVEs is more uncertain to our participants. Particularly, 36 % of the participants neither agree nor disagree with the assumption that reviewers look for CVEs, 42 % assume that reviewers check the presence of CVEs, and 22 % think that reviewers do not look for CVEs in their papers.

When the participants were asked if reviewers brought up CVEs in reviews for papers that did not claim CVEs, the majority (65 %) reported no such experiences. However, 35 % reported that reviewers explicitly requested CVEs. One participant even mentioned, “they wanted to hear about the security relevance of the bugs (or they asked explicitly for CVEs as a proxy for it). We learned our lesson”. This once again suggests possible misconceptions about the value and purpose of CVEs among both reviewers and authors.


Experiences with CVEs in Papers. We subsequently focused on the 50 participants who had published a paper in the past that presented vulnerabilities for which CVEs were assigned. First, we asked whether they think that obtaining CVE(s) has made a difference for paper acceptance. 48 % of the participants believe their CVEs had at least some impact on paper acceptance, while 36 % are unsure about the impact of CVEs, and 16 % somewhat or strongly disagree that CVEs had an impact on paper acceptance. The experiences made

by the subgroup of participants who have published papers entailing CVEs thus roughly match the expectations of the entirety of survey participants.

Factors Promoting CVEs as an Objective for Paper Acceptance. To assess which factors promote the authors' impressions of CVEs increasing the chance of a paper being accepted, we calculate the pairwise Spearman's rank correlation coefficients between the authors' answers to this question and their responses to various other questions in our survey. To maintain our significance threshold of $\alpha = 0.05$ in the presence of multiple testing of four hypotheses, we apply a Bonferroni correction. This leads to an effective threshold for statistical significance of $\hat{\alpha} = \alpha/5 = 0.01$, applied until the next section only. First, we tested whether authors from areas where CVEs are particularly common tend to think that including CVEs in their papers has been crucial to increasing the chance of paper acceptance. However, we observe only very slight correlations between authors thinking that CVEs played a role in paper acceptance and these authors including "Software Security" or "Web Security" in their list of research areas, at correlation coefficients of 0.11 (respectively -0.11) at very low confidence levels ($p=0.22$, $p=0.79$, both $> \hat{\alpha}$). Given our data, we can thus not deduce that in the research areas of Software Security and Web Security, authors particularly often observed that CVEs increased the chance of paper acceptance. Likewise, we cannot identify a statistically significant correlation ($p = 0.95 > \hat{\alpha}$) between authors observing that their CVEs played a role in getting their paper accepted and authors reporting that they primarily obtain CVEs from MITRE, despite the significant share of MITRE-assigned CVEs that we observed in Section 4.1.2.

Instead, we observe that assuming verification to be part of the CVE assignment process correlates with participants reporting that CVEs have increased acceptance chances at a correlation coefficient of 0.37 and a very high certainty of $p = 0.005$. We thus performed a Kruskal-Wallis test to verify if there is indeed a statistical difference between the groups who think that CVEs are verified as part of the assignment process and those who do not. With $p = 0.01 < 0.01 = \hat{\alpha}$, we confirmed that, indeed, participants who think that CVEs are verified are more likely to think that CVEs have played a role in paper acceptance. We see two potential reasons for this: Either the authors who think that CVE-identified vulnerabilities are verified indeed had all their vulnerabilities verified by the affected vendor and promoted this in their paper—which may be appreciated by reviewers. Alternatively, the CVEs that the authors had reported were not verified, but the authors still assumed so. In that case, their perception of CVEs promoting paper acceptance may be either a non-empirically provable observation, or the reviewers made the same incorrect assumption. In either case, this indicates that wrong assumptions on CVE assignment processes seem to affect the writing process. This is also visible in another ob-

servation: The more authors think that the presence of CVEs positively affected the paper acceptance decision, the higher their agreement on the Likert scale that they mention CVEs in the introduction of their paper (correlation coefficient of 0.41 at $p = 0.001 < \hat{\alpha}$). In contrast, the correlation coefficient of 0.303 between thinking that CVEs positively affected the paper acceptance decision, and the authors agreement that they mention CVEs in the abstract of their paper is not statistically significant at $p = 0.017 > \hat{\alpha}$. Thus, it appears that authors consider the abstract as a slightly less appropriate location for CVEs, compared to the introduction.

 **Takeaway #7:** CVEs are seen as desirable and impactful by authors, even if their influence on acceptance is unclear.

5.4 Reviewers' Perspective

Implicit Bias. First, in the uninformed part of our survey (cf. Appendix A.1), we wanted to measure the *implicit bias* that CVEs possibly introduce into the reviewing process. We asked the participants to give us their opinions on an excerpt of a fictitious¹ paper's abstract. Half of the participants received an excerpt that explicitly mentioned "several CVEs" that were discovered using the fictitious technique described in the paper, while the other half of the participants received a nearly identical excerpt that just mentioned that the technique found "several vulnerabilities". We then asked the participants to rate this paper's real-world impact on a Likert scale, based on the excerpt. In this case, CVEs had no statistically significant effect, as evaluating the response distributions using a Kruskal-Wallis test yielded a $p = 0.13 > 0.05 = \alpha$. Therefore, we observe no effect on the implicit, perceived real-world impact by including CVEs into the fictitious excerpt, as the response distributions of both groups are similar.

Positive Impact of CVEs on Paper Perception. We then briefed participants on the objective of our survey and, for this part, filtered out participants without experience as reviewers. We asked our participants whether they actively check for the presence of CVEs (cf. Appendix A.3) during reviewing. 38 % of the participants actively check for CVEs, 24 % are indifferent, and the remaining 38 % do not check for the presence of CVEs. This may be a potential explanation for our previous observation regarding implicit bias: Some reviewers do not actively check for the presence of CVEs, and referring to a vulnerability instead of a CVE may already be sufficient to convey real-world impact. Nevertheless, when assuming four reviewers per paper, there is a 85.22 % chance that at least one reviewer actively checks for CVEs.

We thus suspect that CVEs may become more relevant in the reviewing process once a reviewer has actively brought up CVEs in a discussion or when they are actively sought

¹We avoid real abstracts to avoid bias from familiarity.

after during a discussion between reviewers. In cases where we explicitly ask about the effect of CVEs, 45 % of the participants consider a paper containing CVEs to be more likely to have a real-world impact than a paper without a CVE. An additional 27 % are indifferent on a paper containing CVEs, and only 28 % do not assume that a paper with CVEs has a higher real-world impact than a paper without CVEs. This is in line with another result: 68 % of the participants think that their impression of a paper is positively affected if a paper contains CVEs. Consequently, again assuming four reviewers per paper, there is a 98.95 % chance that at least one reviewer’s impression is positively affected by the presence of a CVE and a 90.04 % chance that the the impressions of two or more reviewers are positively affected. There is also almost no downside for authors when including CVEs in a paper, as only 2 % of reviewers somewhat agree that their impression of a paper worsens if the paper reports CVEs. Therefore, CVEs are likely to positively influence decision outcomes.

🔔 **Takeaway #8:** With four reviewers per paper, mentioning CVEs has a 99 % chance to improve at least one reviewer’s impression of the work. Also, at least one reviewer will actively look for CVEs with 85 % probability.

5.5 Academics’ Opinions on CVEs

Using CVEs as Identifiers and for Recognition. A majority of 63 % of the participants think that CVEs should be used in academia, and a slight majority of 54 % thinks that CVEs should be blinded during submission, for instance, by replacing CVE identifiers by CVE-XXXX-XXXX. The latter is particularly relevant since 54 % of the participants also think that CVEs should be used to credit researchers, 21 % are indifferent, and 25 % believe that CVEs should not be used to credit researchers for their discovery.

Using CVEs as an Impact Metric. While these findings indicate that it seems to be an accepted, and potentially even desirable, practice among academics to use CVE as a means to be recognized for one’s work, our survey participants are more critical with regard to the impact that a CVE should have on paper acceptance. 67 % of the participants oppose the idea that CVEs should impact the paper acceptance decision, 24 % are undecided, and only 10 % think that CVEs should positively affect paper acceptance decisions.

These results stand in surprising contrast to the previous parts of our survey, which suggest that CVEs have an impact on paper acceptance from both authors’ and reviewers’ perspectives. Given that almost all of our participants are reviewers, it is unclear why their perception of a paper improves (68 %) or why they believe a paper has more real-world impact (45 %) when it includes CVEs, while simultaneously feeling strongly that CVEs should not influence paper acceptance.

Table 6: Ways to demonstrate real-world impact, ranked by area-under-curve of response distribution.

Rank	Method	++	+	0	-	--
#1	Evaluation on popular real-world targets	63	35	3	1	0
#2	Found bugs are likely exploitable	45	46	6	4	1
#3	Reproduction of existing bugs	31	45	12	12	2
#4	Reviewers can inspect and verify bugs	23	50	13	14	2
#5	A high number of found bugs	10	42	22	26	2
#6	CVEs were obtained for found bugs	4	41	36	17	4

From our data, we suspect that CVEs are inducing an unwarranted, explicit bias: Once CVEs are actively brought up (e.g., during reviewer discussion), they might increase the paper acceptance probability by suggesting a verified real-world impact, although there is a high consensus that the presence of CVEs should not positively affect acceptance probabilities. Without access to the reviewing process, we cannot investigate this hypothesis. Beyond our survey results, another data point is that five out of 19 papers that claimed CVEs and were accepted at IEEE S&P 2024 had the CVEs mentioned as reason for acceptance in the publicly available meta reviews.

Alternative Impact Metrics. Different means of demonstrating the existence and real-world impact of a vulnerability could replace CVEs in papers. We presented our survey participants with six different ways of how papers could demonstrate real-world impact, and we asked them to state their agreement with each option on a five-point Likert scale (cf. Appendix A.5). Table 6 ranks the suggested methods based on the agreement distribution among the participants, as quantified by the area-under-curve of the cumulative distribution function of responses.

Interestingly, our participants prefer *all* other presented methods over using CVEs to demonstrate impact. Demonstrating real-world impact through evaluation on popular real-world targets was seen as the most favorable option, followed by showing or arguing for exploitability of discovered vulnerabilities. Besides these methods, which rely on the discovery of new vulnerabilities to demonstrate real-world impact, our participants also expressed that they appreciate the replication of existing, known vulnerabilities over purely mentioning CVEs. These results indicate that there may be more effective ways of demonstrating real-world impact of a paper than relying on CVEs. Given the problematic amount of misconceptions around assumed verification of CVEs that we previously discovered, relying on these alternative methods of demonstrating real-world impact may be desirable to bolster scientific integrity of published results.

🔔 **Takeaway #9:** Academics do not want CVEs to influence paper acceptance and prefer alternative impact metrics. Still, CVEs appear to positively affect decisions.

6 Dissecting the Root Causes

Before we provide recommendations on improving CVE utilization in academia, we outline the fundamental root causes leading to the present situation.

We primarily attribute questionable CVEs to the disconnect between the “code owner”, i.e., maintainers or vendors, and a CNA-LR tasked with determining the eligibility of a report for a CVE. Without owning the code, no in-depth verification is possible. Misaligned incentives and poor communication result in the allocation of numerous low-quality CVEs, which is further worsened by unclear procedures and ineffective mechanisms to appeal these assignments.

Misaligned Incentives. Given the prestige of CVEs and academia’s use thereof as a proxy for real-world impact, both academic researchers and individuals strive to receive CVE assignments for their bugs. While not inherently bad, an overt focus on the CVE rather than fixing an actual problem may lead to CVEs for questionable findings rather than only security-relevant bugs. On the other hand, vendors serving as CNAs for their own products may issue CVEs reluctantly to uphold an impression of security.

Several participants in our study highlighted these misaligned incentives in academia. One respondent asserted, for instance, that “CVEs are not a good metric to evaluate a paper’s impact due to both the possibility of self-reporting (without often any verification from the vendor) and different policies/metrics different vendors apply”. Another respondent acknowledged that “sometimes they [CVEs] are gamed to influence paper decisions”.

Regarding the vendor side, respondents even more clearly expressed that “certain vendors issue CVEs very reluctantly”. One testimony reads that the vendor “rejected to assign a CVE because they don’t want to (cannot) fix the bug”. Multiple participants highlighted the vendors’ incentive to “protect their reputation” or avoid a CVE that “looks bad for their advertisement”; thus “some vendors do (implicitly) take the CVEs on their products as kind of a ‘shame’, so they mostly are not willing to issue”.

Lack of Verification. As CNA-LRs are, by design, not familiar with the underlying code for which a flaw was reported, they cannot perform any in-depth verification. Typically, they err on the safe side and assign a CVE instead of risking not assigning one to a possibly exploitable vulnerability. The inherent gap between the “code owner” and CNA also makes it challenging to ensure the bugs were responsibly disclosed to maintainers (who simply may no longer be around to patch the software). This enables the assignment of CVEs to findings of questionable quality. Some respondents are aware that “the verification process is very manual and rather subjective, so I think inaccuracies are inevitable”. Another participant acknowledged that “CVEs are wrongly assigned all the time

[...] quite liberally, too”. Indeed, at least one participant has “seen papers mentioning CVEs they got that, after careful inspection, are irrelevant or wrong”.

Ineffective Retraction. Once a finding is published, it rarely undergoes retraction. While formal mechanisms to *dispute* and even *reject* a CVE exist, they are rarely used in practice. Maintainers are often unaware of these mechanisms, and thus do not attempt to reject a CVE [29]. Even when maintainers inform a CNA-LR such as MITRE, a CVE may not be rejected: If the reporter maintains a different opinion, MITRE may opt to simply mark the CVE as *disputed*, noting the differing opinions (regardless of underlying facts). Generally, the effort to prove the absence of a vulnerability is pushed onto the maintainer, who has to follow a multi-stage dispute process [42]. Paired with subpar CVE records, where references point to repositories or papers rather than bug reports, verifying (and potentially rejecting) CVEs after assignment is difficult and arduous.

7 Recommendations

We make six recommendations of varying invasiveness to improve the use of CVEs in academia. Note that our recommendations are explicitly intended to encourage community discussion and are not meant to serve as definitive answers. Hence, some recommendations are orthogonal, offering alternative options within the decision space and leaving the choice of direction to the community itself.

We structure the recommendations into three categories: those targeting authors’ *submissions*, those aimed at enhancing *peer review* and *artifact evaluation*, and those addressing broader issues beyond academia. We emphasize that reducing misuse benefits not just the academic community. Bogus CVEs are already a burden to the maintainers. For example, supply chain scanners may flag dependencies with known CVEs [30], leading to end users requesting the release of a fixed version, even if the CVE does not represent a security vulnerability or bug.

7.1 Submission-Centric Recommendations

Imposing constraints on submitted papers may reduce incentives to obtain CVEs solely for their own sake or ease the identification of questionable CVEs during review.

Recommendation 1: Prohibit CVE Mentions. As the wrong incentives for obtaining CVEs primarily arise due to the need to persuade the reviewers during peer review, a straightforward yet effective solution would be to prohibit mentioning any obtained CVEs in the initial submission. This would reinforce the true purpose of CVEs as mere *identifiers* and eliminate incentives for authors to acquire CVEs of questionable quality. Additionally, as certain projects or

vendors do not file CVEs at all, this recommendation would put bugs found in such projects on equal footing. Moreover, citing obtained CVEs can lead to deanonymization, as the details associated with CVEs often reveal the identities of the reporters. This concern has prompted the IEEE S&P 2025 call for papers [1] to explicitly mandate the anonymization of any claimed CVEs, raising the question of why CVEs would not be prohibited entirely at submission time.

We emphasize that this recommendation does *not* entail that authors cannot claim real-world impact in their submissions anymore. Instead, authors would need to provide a more nuanced argument for real-world impact, such as detailing the vendor response and mitigations rather than solely referencing a CVE number. Crucially, this recommendation does not rule out legitimate uses of CVEs as unique identifiers in academia, as authors can still request CVEs and include them into the camera-ready version of their paper.

Recommendation 2: Require More Details. While we personally do not believe CVEs should be used as a metric, we recognize other community members do (see Q26). Hence, a less drastic alternative could be to require mentioning more details for obtained CVEs. For example, mandating the inclusion of the assigning CNA would be a logical step, as our analysis clearly shows that a small number of CNA-LRs are responsible for the vast majority of questionable CVEs. However, this recommendation assumes that reviewers are aware of CNA quality and assignment processes, which is not always the case according to the results of our survey. Another interesting detail to specify would be *who* requested the CVE, whether it was the authors themselves (and if so, their reasoning) or the maintainers of a project. While good reasons exist, almost all problems arise from authors contacting a CNA-LR on their own rather than coordinating with maintainers. Another apparent solution might be to mandate the inclusion of the CVSS score along with the CVE as a measure of impact. However, CVSS has been repeatedly criticized [50,55,63] for being an inherently inconsistent and biased scoring methodology, limiting its reliability as a definitive measure of severity.

7.2 Reviewing Recommendations

Orthogonal to imposing additional constraints on authors, program chairs can take further steps to prevent CVE misuse.

Recommendation 3: Clarify Expectations. We believe it is worthwhile to inform PC members about expectations surrounding CVEs. This could include explicitly highlighting the pitfalls of treating CVEs as a metric, clarifying the CVE assignment process to address the common misconception that CVEs are verified (cf. Takeaway #6), and discouraging or prohibiting the use of CVEs as indicators of real-world impact in both private discussions and public meta-reviews.

If the community chooses to continue relying on CVEs and builds upon Recommendation 2, program committee members can be guided on which aspects of CVE-related information may warrant closer scrutiny during the review process. This can help ensure that CVEs with questionable or uncertain impact do not lead to confusion. Ambiguous cases could be referred to a Research Ethics Committee (REC) for further evaluation. In addition to informing the PC, we recommend that conferences clearly articulate their stance on the use of CVEs in the call for papers, thereby setting expectations and outlining any relevant constraints for authors.

Recommendation 4: Verification in Artifact Evaluation.

Given that USENIX Security recently moved to an open science policy where the availability of artifacts is expected by default, and other major security conferences similarly adopted artifact evaluation, it is already an integral part of the submission cycle. At this point in time, authors are no longer anonymized, and artifact reviewers already need a certain technical expertise to understand and verify the artifact. Thus, a simple but effective recommendation could be to include validation of CVEs as an expected part of the artifact evaluation process. Like artifact evaluation itself, the default should be a verification of CVEs or, if inapplicable, authors should provide a valid explanation of why evaluation (or CVE verification) cannot take place. We stress there are reasons beyond the authors' control that may prevent a verification during artifact evaluation, such as when the maintainer applied for a CVE and has not yet issued a disclosure, such that the CVE remains in the reserved state. Papers found to contain mostly questionable CVEs could be required to issue an erratum, thereby imposing consequences for the misuse of CVEs.

7.3 Recommendations Beyond Academia

Since the issue is not limited to academia, outside stakeholders may implement changes that affect our use of CVEs.

Recommendation 5: Verification during Assignment.

CNA-LRs could implement additional *verification* of bugs before assigning a CVE. This is inherently challenging, as a CNA-LR by definition is not familiar with the particular code base. Still, any improvement in this realm would make the system more robust. Interestingly, the opposite move, assigning CVEs to any bug without verification regarding security impact, as implemented by the Linux kernel, may lead to the same effect, as CVEs will lose their prestige over time.

Recommendation 6: Simplified Dispute Process. While it may not prevent questionable CVEs outright, establishing a *transparent, efficient, and well-documented* appeal process would allow maintainers to challenge CVE assignments more effectively. Such a correction mechanism would help others more easily recognize and disregard dubious CVEs, thereby reducing CVE misuse over time.

8 Conclusion

In this work, we have demonstrated that the use of CVEs in academic security papers is steadily rising, highlighting that CVEs are perceived as a proxy for real-world impact beyond mere identification. A large-scale review of 1,803 CVEs across 304 A* papers revealed that some papers may follow misaligned incentives in obtaining questionable CVEs. Surveying more than 100 members of the security community, we revealed widespread misconceptions about the CVE ecosystem in our community. Many authors and reviewers believe that CVEs measure a real-world impact and are verified. Based on our findings, we propose six actionable suggestions on how to move forward in our use of CVEs in the academic security community. We hope that our work and recommendations can contribute to this ongoing debate.

9 Ethics Considerations

User Survey. As a reflective meta-science study that includes a user survey, our paper adheres to best practices for conducting surveys to ensure ethical and rigorous research. Prior to performing the user survey, our study plan and related materials underwent evaluation and approval by the Institutional Review Board (IRB) hosted at our institution. We also consulted social scientists to review our survey design, incorporating their feedback to enhance the study’s setup and ensure methodological rigor. To maintain transparency, we provided participants with a detailed document outlining the study’s purpose, benefits, and funding sources. Participants were asked to review this information and provide informed consent before deciding whether to join. They also retained the right to withdraw from the study at any time without any consequences. To ensure privacy and confidentiality, no data that could identify individual participants of the study was collected. Participants could voluntarily leave an email address or contact us anonymously via email. Any data entered in the feedback fields is considered sensitive and has been removed before further processing. No personally identifiable information or sensitive data will ever be released. Our survey is GDPR-compliant.

Anonymization of Questionable CVEs. As part of this work, we critically assessed the CVE assignments obtained by fellow researchers. Among these, we identified several outlier papers that claim CVEs with questionable or uncertain impact (cf. Section 4.1.4). After extensive internal discussion among the authors, we decided not to disclose individual author names or paper titles in the submitted manuscript. Furthermore, after considering input from the Research Ethic Committee and the USENIX Security’25 reviewers, we decided to also not disclose affected papers to the PC chairs of the respective conference editions in which they were pub-

lished. We motivate and document our reasoning behind these decisions below.

Most importantly, we emphasize that our goal is not to blame specific individuals, institutions, or papers, but to inform and spark constructive discussion around a widespread practice — i.e., using CVEs as a poor proxy for real-world impact — that affects the entire community, not just a selected few papers. In this respect, our decision to anonymize the analyzed papers is in line with other meta-studies that critically examine prior work [3, 11, 59].

The observed CVEs with questionable or uncertain impact in Section 4.1.4 first and foremost reinforce our main argument that CVEs are an unreliable metric, but they may occur for various reasons and do not necessarily constitute clear, incontestable evidence of fraud. For example, authors may have been simply sloppy or followed the (common) practice of “overclaiming”, or there could have been circumstances beyond their control (e.g., an uncooperative vendor who wishes to suppress CVEs to reduce perceived customer impact). Likewise, even CVEs that are associated with a fix may not necessarily constitute cases where a CVE should have been assigned (for example, a CVE may be obtained without maintainer consent for a non-security-relevant bug). Therefore, in the absence of objective evidence of bad-faith actions, PC chairs have no clear basis to act on the information we can provide. As a result, we see no benefit in sharing a list of papers with unclear CVEs to the PC chairs. Moreover, doing so could unintentionally imply that the issue is limited to cases where problematic CVEs are identifiable solely by their assessed status.

Note that our data sharing approach with two separate lists, outlined in Section 10, avoids putting blame on people, while still allowing for reproduction of the results presented in the paper. Additionally, we stress that our analysis is based on publicly available information viewable by anyone at any time, including CVE records or published papers.

10 Open Science

To encourage future research and promote open science, we publish analysis scripts and (non-sensitive) datasets with the detailed qualitative analysis and survey results at <https://doi.org/10.5281/zenodo.15611161> and https://github.com/mu00d8/cves_in_academia.

In particular, we release:

- our mapping of CVEs to outcome, albeit without connecting them to specific papers or authors;
- a separate list of papers qualitatively analyzed (not linked to CVEs or outcomes);
- the raw survey results without personally identifiable information;
- all scripts used for evaluation purposes.

For legal reasons, we do not release PDF files of papers or any representation resulting directly thereof.

Acknowledgments

We thank the participants of our user survey. We are further grateful to the anonymous reviewers and our shepherd for their valuable feedback. Additionally, we thank Michael Schilling and Ananta Soneji for their comments on a draft of this work and our survey. This work was supported by the European Research Council (ERC) under the consolidator grant RS³ (101045669), the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Germany's Excellence Strategy – EXC 2092 CASA – 390781972 as well as DFG 491039149, the European Union's Horizon 2020 research and innovation programme under project TESTABLE, grant agreement No 101019206, the German Federal Office for Information Security (FKZ: Pentest-5GSec - 01MO23025B), the Research Fund KU Leuven, and the Cybersecurity Research Program Flanders.

References

- [1] 46th IEEE Symposium on Security and Privacy. Call for Papers. <https://www.ieee-security.org/TC/SP2025/cfpapers.html>, 2024.
- [2] Nikolaos Alexopoulos, Manuel Brack, Jan Philipp Wagner, Tim Grube, and Max Mühlhäuser. How Long Do Vulnerabilities Live in the Code? A Large-Scale Empirical Measurement Study on FOSS Vulnerability Lifetimes. In *USENIX Security Symposium*, 2022.
- [3] Daniel Arp, Erwin Quiring, Feargus Pendlebury, Alexander Warnecke, Fabio Pierazzi, Christian Wressnegger, Lorenzo Cavallaro, and Konrad Rieck. Dos and Don'ts of Machine Learning in Computer Security. In *USENIX Security Symposium*, 2022.
- [4] Yoav Benjamini, Richard De Veaux, Bradley Efron, Scott Evans, Mark Glickman, Barry I. Graubard, Xuming He, Xiao-Li Meng, Nancy Reid, Stephen M. Stigler, Stephen B. Vardeman, Christopher K. Winkle, Tommy Wright, Linda J. Young, and Karen Kafadar. ASA President's Task Force Statement on Statistical Significance and Replicability. *Harvard Data Science Review*, 3(3), jul 30 2021. <https://hdsr.mitpress.mit.edu/pub/50v12b07>.
- [5] Guru Bhandari, Amara Naseer, and Leon Moonen. CVEfixes: Automated Collection of Vulnerabilities and their Fixes from Open-source Software. In *International Conference on Predictive Models and Data Analytics in Software Engineering*, 2021.
- [6] Lukas Blecher, Guillem Cucurull, Thomas Scialom, and Robert Stojnic. Nougat: Neural Optical Understanding for Academic Documents, 2023.
- [7] Neophytos Christou, Di Jin, Vaggelis Atlidakis, Baishakhi Ray, and Vasileios P. Kemerlis. IvySyn: Automated Vulnerability Discovery in Deep Learning Frameworks. In *USENIX Security Symposium*, 2023.
- [8] Jonathan Corbet. What to do about CVE Numbers. <https://lwn.net/Articles/944209/>, October 2019.
- [9] !CVE Program. About the !CVE Program. <https://notcve.org/about.html>, 2023.
- [10] Miriam E David. Research Quality Assessment and the Metrication of the Social Sciences. *European Political Science*, 7:52–63, 2008.
- [11] Nurullah Demir, Matteo Große-Kampmann, Tobias Urban, Christian Wressnegger, Thorsten Holz, and Norbert Pohlmann. Reproducibility and Replicability of Web Measurement Studies. In *ACM Web Conference (WWW)*, 2022.
- [12] Trevor Dunlap, Elizabeth Lin, William Enck, and Bradley Reaves. VFCFinder: Seamlessly Pairing Security Advisories and Patches. *CoRR*, abs/2311.01532, 2023.
- [13] Zakir Durumeric, Frank Li, James Kasten, Johanna Amann, Jethro Beekman, Mathias Payer, Nicolas Weaver, David Adrian, Vern Paxson, Michael Bailey, et al. The Matter of Heartbleed. In *ACM SIGCOMM Conference on Internet Measurement (IMC)*, 2014.
- [14] Jake Edge. The Bogus CVE Problem. <https://lwn.net/Articles/944209/>, Sept 2023.
- [15] Jiahao Fan, Yi Li, Shaohua Wang, and Tien N Nguyen. A C/C++ Code Vulnerability Dataset with Code Changes and CVE Summaries. In *International Conference on Mining Software Repositories*, 2020.
- [16] Jerry Gamblin. For the First Time Ever, over 5,000 CVEs were Published in a Month. <https://x.com/jgamblin/status/1796941848230969367>, June 2024.
- [17] Maarten Grootendorst. KeyBERT: Minimal keyword extraction with BERT, 2020.
- [18] Florian Hantke. How to Get CVEs Online (Fast): Some Thoughts about CVEs. <https://infosecwriteups.com/how-to-get-cves-online-fast-c0d6d897c04d>, January 2024.
- [19] Greg Kroah Hartman. Linux is a CNA. <http://www.kroah.com/blog/2024/02/13/linux-is-a-cna/>, February 2024.
- [20] Ayako A. Hasegawa, Daisuke Inoue, and Mitsunori Akiyama. How WEIRD is usable privacy and security research? In *USENIX Security Symposium*, 2024.
- [21] Liang He, Hong Hu, Purui Su, Yan Cai, and Zhenkai Liang. FreeWill: Automatically Diagnosing Use-after-free Bugs via Reference Mismatching Detection on Binaries. In *USENIX Security Symposium*, 2022.
- [22] Cormac Herley and Paul C Van Oorschot. SoK: Science, Security and the Elusive Goal of Security as a Scientific Pursuit. In *IEEE Symposium on Security and Privacy (S&P)*, 2017.
- [23] Hyunji Hong, Seunghoon Woo, Eunjin Choi, Jihyun Choi, and Heejo Lee. xVDB: A High-Coverage Approach for Constructing a Vulnerability Database. *IEEE Access*, 10:85050–85063, 2022.
- [24] jalan. pdftotext. <https://github.com/jalan/pdftotext>, 2024.
- [25] Josh Junon. Fix ReDos in hwb() Parser (Low-severity). <https://github.com/qix-/color-string/commit/0789e21284c33d89ebc4ab4ca6f759b9375ac9d3>.
- [26] Jan H Klemmer, Julian Schmäser, Byron M Lowens, Fabian Fischer, Lea Schmäser, Florian Schaub, and Sascha Fahl. Transparency in Usable Privacy and Security Research: Scholars' Perspectives, Practices, and Recommendations. In *IEEE Symposium on Security and Privacy (S&P)*, 2025.
- [27] Seth Michael Larson. The Python Software Foundation has been Authorized by the CVE Program as a CVE Numbering Authority (CNA). <https://pyfound.blogspot.com/2023/08/psf-authorized-as-cna.html>, August 2023.
- [28] Victor Le Pochat and Wouter Joosen. Analyzing Cyber Security Research Practices through a Meta-Research Framework. In *ACM Cyber Security Experimentation and Test Workshop (CSET)*, 2023.
- [29] Wen Li and NumPy team. Insecure String Comparison (Incomplete Comparison) in `_convert_from_str` of descriptor.c. <https://github.com/numpy/numpy/issues/18993>, 2022.
- [30] Wen Li and NumPy team. Missing Return-value Validation of the Function `PyArray_DescrNew` #19038. <https://github.com/numpy/numpy/issues/19038#issuecomment-1026590881>, 2022.
- [31] MITRE Corporation. CVE Website. <https://www.cve.org/>.
- [32] MITRE Corporation. History of CVEs. <https://www.cve.org/About/History>.
- [33] MITRE Corporation. Metrics. <https://www.cve.org/About/Metrics>.
- [34] MITRE Corporation. List of Partners (CNAs). <https://www.cve.org/partnerinformation/listofpartners>, 2024.

- [35] Christian Niesler, Sebastian Surminski, and Lucas Davi. HERA: Hot-patching of Embedded Real-time Applications. In *Symposium on Network and Distributed System Security (NDSS)*, 2021.
- [36] Georgios Nikitopoulos, Konstantina Dritsa, Panos Louridas, and Dimitris Mitropoulos. CrossVul: A Cross-language Vulnerability Dataset with Commit Data. In *ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, 2021.
- [37] Lin Padgham, Young Lee, Shazia Sadiq, Michael Winikoff, Alan Fekete, Stephen MacDonell, Dali Kaafar, and Stefanie Zollmann. CORE Rankings.
- [38] Eric Pauley, Paul Barford, and Patrick McDaniel. The CVE Wayback Machine: Measuring Coordinated Disclosure from Exploits against Two Years of Zero-Days. In *ACM SIGCOMM Conference on Internet Measurement (IMC)*, 2023.
- [39] Henning Perl, Sergej Dechand, Matthew Smith, Daniel Arp, Fabian Yamaguchi, Konrad Rieck, Sascha Fahl, and Yasemin Acar. VCCFinder: Finding Potential Vulnerabilities in Open-Source Projects to Assist Code Audits. In *ACM Conference on Computer and Communications Security (CCS)*, 2015.
- [40] The CVE Program. What does it mean when a CVE Record is marked REJECTED? https://www.cve.org/ResourcesSupport/FAQs#pc_cve_recordsreject_signify_in_cve_record.
- [41] The CVE Program. Why is a CVE Record marked as RESERVED when a CVE ID is being publicly used? https://www.cve.org/ResourcesSupport/FAQs#pc_cve_recordswhy_cve_record_marked_RESERVED_when_being_publicly_used.
- [42] The CVE Program. CVE program policy and procedure for disputing a CVE record. <https://www.cve.org/Resources/General/Policies/CVE-Record-Dispute-Policy.pdf>, 2022.
- [43] Kevin Purdy. Nginx Core Developer Quits Project in Security Dispute, Starts “freenginx” Fork. <https://arstechnica.com/information-technology/2024/02/nginx-key-developer-starts-a-freenginx-fork-after-dispute-with-parent-firm/>, February 2024.
- [44] Antonino Sabetta, Serena Elisa Ponta, Rocio Cabrera Lozoya, Michele Bezzi, Tommaso Sacchetti, Matteo Greco, Gergő Balogh, Péter Hegedűs, Rudolf Ferenc, Ranindya Paramitha, et al. Known Vulnerabilities of Open Source Projects: Where Are the Fixes? *IEEE Security & Privacy*, 22(2), 2024.
- [45] Tobias Scharnowski, Simon Wörner, Felix Buchmann, Nils Bars, Moritz Schloegel, and Thorsten Holz. Hoedur: Embedded Firmware Fuzzing using Multi-Stream Inputs. In *USENIX Security Symposium*, 2023.
- [46] Moritz Schloegel, Nils Bars, Nico Schiller, Lukas Bernhard, Tobias Scharnowski, Addison Crump, Arash Ale-Ebrahim, Nicolai Bissantz, Marius Muench, and Thorsten Holz. SoK: Prudent Evaluation Practices for Fuzzing. In *IEEE Symposium on Security and Privacy (S&P)*, 2024.
- [47] Christian Schneijderberg, Lars Müller, and Nicolai Götze. Is the German Academic Profession on Metrification Autopilot? A Study of 25 Years of Publication Outputs. 2020.
- [48] Abhishek Shah, Dongdong She, Samanway Sadhu, Krish Singal, Peter Coffman, and Suman Jana. MC²: Rigorous and Efficient Directed Greybox Fuzzing. In *ACM Conference on Computer and Communications Security (CCS)*, 2022.
- [49] Ananta Soneji, Faris Bugra Kokulu, Carlos Rubio-Medrano, Tiffany Bao, Ruoyu Wang, Yan Shoshitaishvili, and Adam Doupe. “Flawed, but like democracy we don’t have a better system”: The Experts’ Insights on the Peer Review Process of Evaluating Security Papers. In *IEEE Symposium on Security and Privacy (S&P)*, 2022.
- [50] Jonathan Spring, Eric Hatleback, Allen Householder, Art Manion, and Deana Shick. Time to Change the CVSS? *IEEE Security & Privacy*, 19(2), 2021.
- [51] Jonathan M. Spring, Tyler Moore, and David Pym. Practicing a Science of Security: A Philosophy of Science Perspective. In *New Security Paradigms Workshop*, 2017.
- [52] Daniel Stenberg. CVE-2020-19909: Bogus Report Filed by Anonymous. <https://curl.se/docs/CVE-2020-19909.html>, 2023.
- [53] Daniel Stenberg. CVE-2020-19909 is everything that is wrong with CVEs. <https://daniel.haxx.se/blog/2023/08/26/cve-2020-19909-is-everything-that-is-wrong-with-cves/>, August 2023.
- [54] Daniel Stenberg. CVE-2023-52071: Bogus Report Filed by Anonymous. <https://curl.se/docs/CVE-2023-52071.html>, 2024.
- [55] Daniel Stenberg. CVSS is dead to us. <https://daniel.haxx.se/blog/2025/01/23/cvss-is-dead-to-us/>, January 2025.
- [56] Octavian Suciu, Connor Nelson, Zhuoer Lyu, Tiffany Bao, and Tudor Dumitras. Expected Exploitability: Predicting the Development of Functional Vulnerability Exploits. In *USENIX Security Symposium*, 2022.
- [57] Seyed Mohammadjavad Seyed Talebi, Zhihao Yao, Ardalan Amiri Sani, Zhiyun Qian, and Daniel Austin. Undo Workarounds for Kernel Bugs. In *USENIX Security Symposium*, 2021.
- [58] Xin Tan, Yuan Zhang, Chenyuan Mi, Jiajun Cao, Kun Sun, Yifan Lin, and Min Yang. Locating the Security Patches for Disclosed OSS Vulnerabilities with Vulnerability-Commit Correlation Ranking. In *ACM Conference on Computer and Communications Security (CCS)*, 2021.
- [59] Erik van der Kouwe, Gernot Heiser, Dennis Andriess, Herbert Bos, and Cristiano Giuffrida. SoK: Benchmarking Flaws in Systems Security. In *IEEE European Symposium on Security and Privacy (EuroS&P)*, 2019.
- [60] Steven Vaughan-Nichols. Now it’s PostgreSQL’s turn to have a bogus CVE. <https://opensourcewatch.beehiiv.com/p/now-postgresqls-turn-bogus-cve>, 2024.
- [61] Xinda Wang, Shu Wang, Pengbin Feng, Kun Sun, and Sushil Jajodia. PatchDB: A Large-Scale Security Patch Dataset. In *IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, 2021.
- [62] James Wilsdon. The Metric Tide: Independent Review of the Role of Metrics in Research Assessment and Management. 2016.
- [63] Julia Wunder, Andreas Kurtz, Christian Eichenmüller, Freya Gassmann, and Zinaida Benenson. Shedding Light on CVSS Scoring Inconsistencies: A User-Centric Study on Evaluating Widespread Security Vulnerabilities. In *IEEE Symposium on Security and Privacy (S&P)*, 2024.
- [64] Congying Xu, Bihuan Chen, Chenhao Lu, Kaifeng Huang, Xin Peng, and Yang Liu. Tracking Patches for Open Source Software Vulnerabilities. In *ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering (ESEC/FSE)*, 2022.
- [65] Carter Yagemann, Simon P. Chung, Brendan Saltaformaggio, and Wenke Lee. Automated Bug Hunting With Data-Driven Symbolic Root Cause Analysis. In *ACM Conference on Computer and Communications Security (CCS)*, 2021.
- [66] Yuchen Zhang, Chengbin Pang, Georgios Portokalidis, Nikos Triantopoulos, and Jun Xu. Debloating Address Sanitizer. In *USENIX Security Symposium*, 2022.
- [67] Sebastian Österlund, Kaveh Razavi, Herbert Bos, and Cristiano Giuffrida. ParmeSan: Sanitizer-guided Greybox Fuzzing. In *USENIX Security Symposium*, 2020.

Table 7: Survey participants’ demographics.

Current occupation			
Academia		Industry	
Professor	68	Researcher	13
PhD student	10	Other	2
Postdoctoral researcher	4		
Master student	3		
Other	2		
Research area or professional focus			
Multiple answers possible.			
Software Security	53	Machine Learning and Security	22
Web Security	21	Applied Cryptography	18
Network Security	18	Privacy and Anonymity	17
Hardware, Side Channels, and CyberPhysical Systems	15	Security Usability and Measurement	14
Formal Methods and Programming	9	Blockchain and Distributed Systems	5
Other	4		
Academic conferences: Authoring			
Have you published at an academic conference or journal before?			
Yes	97	No	5
Experience [years]			
Min	2	Max	35
Median	11	Std.Dev.	6.27
Academic conferences: Reviewing			
Have you reviewed for an academic conference before?			
Yes	100	No	2
Experience [years]			
Min	1	Max	30
Median	7	Std.Dev.	6.15
Experience with CVEs			
Have you worked with CVEs before?			
Yes, I requested or received CVEs before	52	No, never heard of them	1
Yes, I have seen them before (but I never requested nor received one before)	50		

A Academic Survey Results

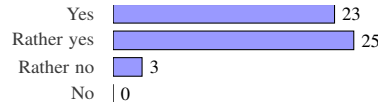
We surveyed 103 participants, whose demographics are presented in Table 7. For full transparency, we make the full survey available in our artifact. The distributions with the absolute number of answers for the different questions are provided below.

A.1 Implicit Bias

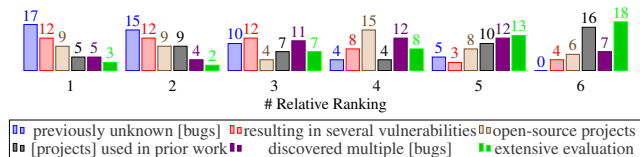
Q1 (variant A). Assume you are a member of the program committee of a top-tier security conference and need to review a paper. Consider the following excerpt from a (fictitious) abstract in which we have deliberately omitted all unnecessary technical details:

Through an extensive evaluation of open-source projects used in prior work, our technique discovered multiple, previously unknown bugs resulting in several vulnerabilities.

Do you believe this paper would have real-world impact based on this sentence?



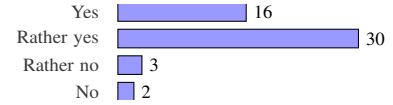
Please sort the following snippets from the above sentence according to their relationship to real-world impact (strongest first, move by dragging).



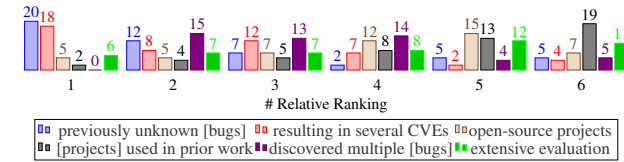
Q1 (variant B). Assume you are a member of the program committee of a top-tier security conference and need to review a paper. Consider the following excerpt from a (fictitious) abstract in which we have deliberately omitted all unnecessary technical details:

Through an extensive evaluation of open-source projects used in prior work, our technique discovered multiple, previously unknown bugs resulting in several CVEs.

Do you believe this paper would have real-world impact based on this sentence?

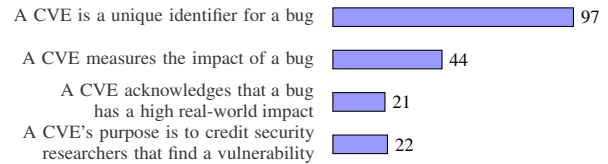


Please sort the following snippets from the above sentence according to their relationship to real-world impact (strongest first, move by dragging).

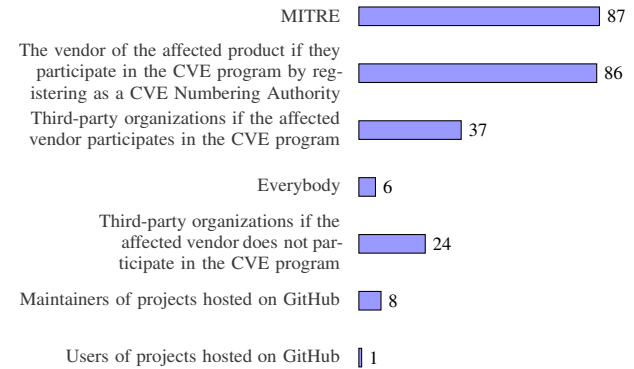


A.2 CVE Knowledge

Q2. Which of the following points applies to CVEs? (multiple options can be checked)



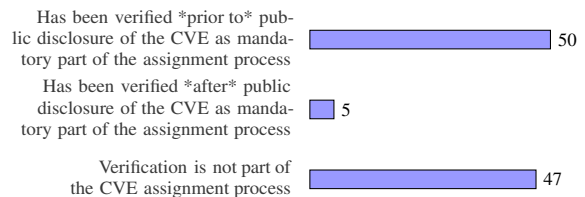
Q3. CVEs can be assigned by... (multiple options can be checked)



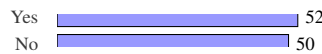
Q4. CVE details (such as affected products) are publicly visible once a CVE number is assigned?



Q5. If a CVE is assigned, this implies that the underlying bug...

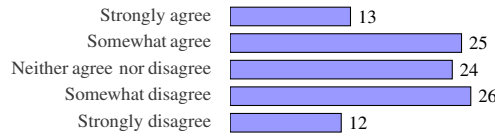


Q6. Information on whether a published CVE has been assigned by an external entity or the vendor of the affected product is publicly available?

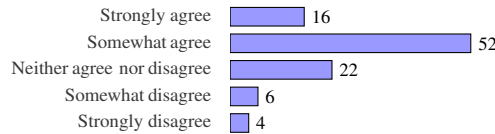


A.3 Reviewer Perspective

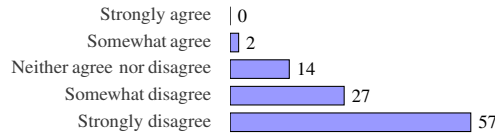
Q7. I explicitly check if a paper reports CVE(s).



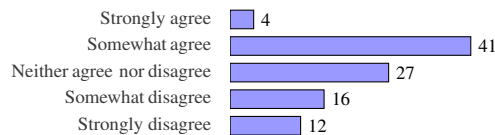
Q8. If a paper reports CVEs, my opinion of it improves.



Q9. If a paper reports CVEs, my opinion of it worsens.

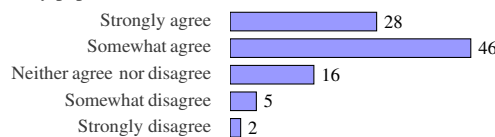


Q10. I believe a paper having CVEs is more likely to have real-world impact than a paper without any.

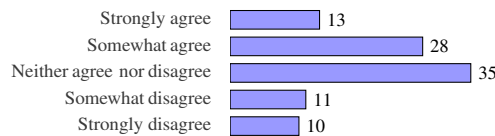


A.4 Author Perspective

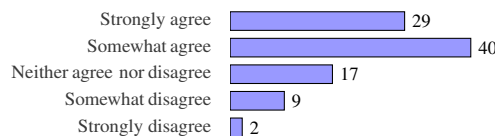
Q11. Mentioning CVEs assigned to my work helps to demonstrate the real-world impact of my paper.



Q12. The reviewers of my papers check if I have obtained CVEs.



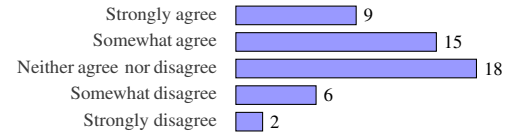
Q13. I try to obtain CVEs for new vulnerabilities.



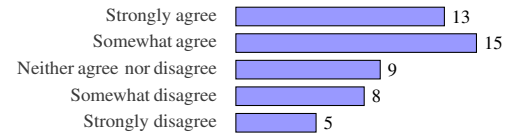
Q14. In the past, have you obtained CVEs for bugs found by a method, technique, or tool presented in a paper of yours?



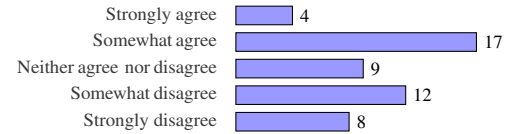
Q15. Having obtained CVE(s) has made a difference for paper acceptance.



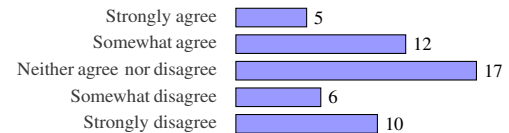
Q16. Usually, we asked the maintainers/vendors to assign CVEs.



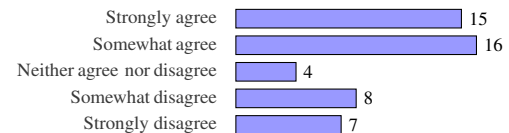
Q17. The maintainers/vendors have requested CVEs without us having asked them for CVEs.



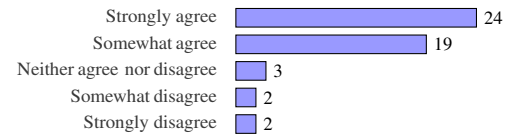
Q18. We usually requested CVEs directly from MITRE.



Q19. We usually mention obtained CVEs in the abstract of a paper.



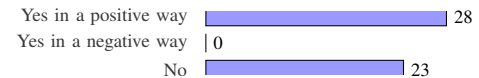
Q20. We usually mention obtained CVEs in the introduction of a paper.



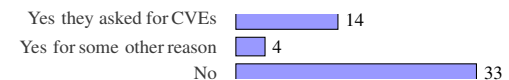
Q21. Have you (or a co-author) contacted MITRE to request a CVE before? If so, why did you choose to request the CVE directly from MITRE?



Q22. When having obtained and mentioned CVEs in your paper, did the reviewers mention the CVEs in their reviews?

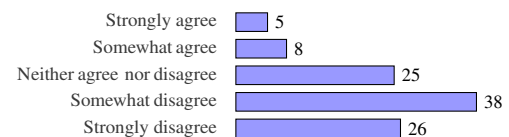


Q23. For a paper that did not report CVEs, have reviewers mentioned CVEs in their reviews?

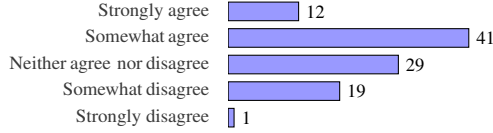


A.5 Academic Opinions

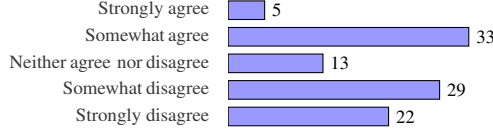
Q24. Academia should not use CVEs.



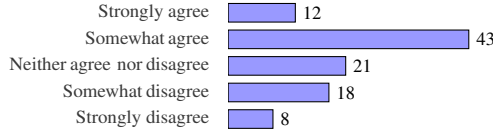
Q25. CVEs should be only a unique identifier for bugs.



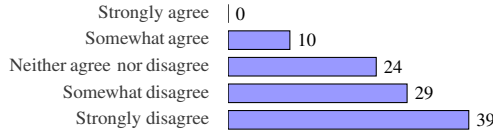
Q26. CVEs should be a metric (e.g., to measure real-world impact of found bugs).



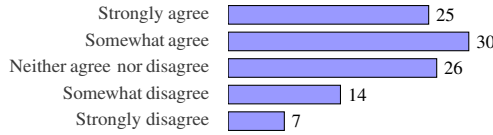
Q27. CVEs should be used to credit security researchers.



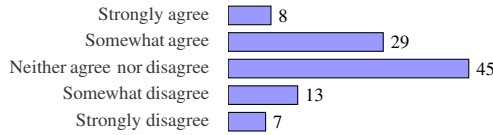
Q28. CVEs should influence the decision of whether a paper is accepted or rejected.



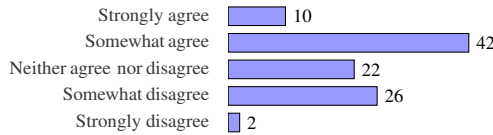
Q29. CVEs should be blinded during submission (e.g., as CVE-XXXX-XXX). Not blinding CVEs would allow reviewers to assess the validity of the CVEs during their review, but authors would have to submit CVE requests anonymously (and cannot take credit directly).



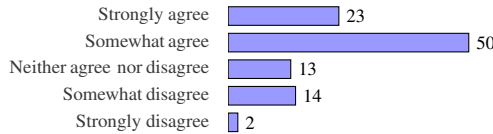
Q30. Authors should request the assignment of CVEs.



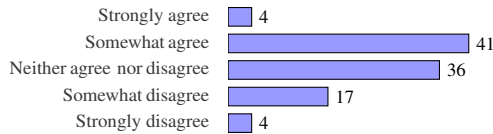
Q31. What is your preferred way for a paper to underline its real-world impact: A high number of found bugs.



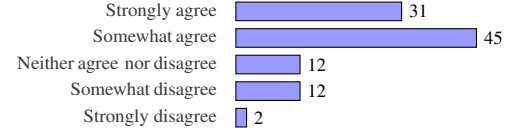
Q32. What is your preferred way for a paper to underline its real-world impact: Reviewers can inspect and verify bugs (i.e., bugs are not blinded).



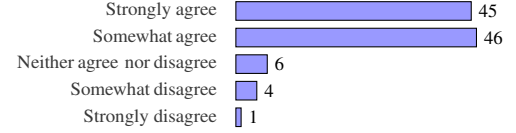
Q33. What is your preferred way for a paper to underline its real-world impact: CVEs were obtained for found bugs.



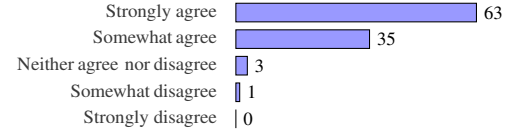
Q34. What is your preferred way for a paper to underline its real-world impact: Reproduction of existing bugs.



Q35. What is your preferred way for a paper to underline its real-world impact: Found bugs are likely exploitable.



Q36. What is your preferred way for a paper to underline its real-world impact: Evaluation on popular real-world targets.



B CVE Location

To better understand the context of academic use of CVEs, we analyze the location of the CVEs mentioned in the paper. For this, we leverage the OCR dataset, as it preserves the structure of a paper. The location of a CVE in the paper may indicate its potential use: While CVEs mentioned in the main parts of the paper are more likely to indicate original contributions by the authors, CVEs included in the reference section are often citations to other CVEs. We visualize the position of CVEs in papers as a normalized heat map in Figure 3, which shows that the vast majority of CVEs appear in the references or appendix of a publication. We also notice that papers that claim many CVEs as part of their contribution do not mention the CVE IDs early in the paper and rather have them in their appendix or results section. Only 5.2% of the publications in our dataset mention a CVE in their introduction section.

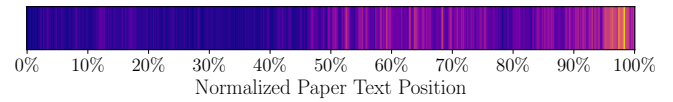


Figure 3: A normalized heat map showing where CVEs appear in the text of a publication for our dataset.