Trust but Verify: An Assessment of Vulnerability Tagging Services

Szu-Chun Huang[†], Max van der Horst[§], Georgios Smaragdakis[†],
Michel van Eeten[†], and Yury Zhauniarovich[†]

[†]*Delft University of Technology*,
[§]*Dutch Institute of Vulnerability Disclosure*

Abstract

Vulnerability tagging services based on large-scale scanning campaigns are the de facto source for identifying vulnerable devices and for attack surface discovery in organizations and on the Internet. Enterprises, government agencies, and researchers increasingly rely on such services to assess the risk level of Internet-facing computing infrastructures. However, we lack independent evaluations of how accurate such services are when detecting vulnerable endpoints.

In this paper, we perform independent experiments to assess the trustworthiness of vulnerability tagging services. We compare the reports of the market leader in this space, the Shodan Search Engine, with the reports of our experiments using carefully crafted Nuclei templates that are designed to target requests based on specific vulnerability checks and payloads. We find that for Shodan payload-based detections (called "verified CVEs"), most Shodan detections are confirmed. Yet Nuclei finds many more vulnerable endpoints, so defenders might face massive underreporting. For Shodan banner-based detections ("unverified"), the opposite problem emerges: massive overreporting of false positives. Ironically, this is confirmed by Shodan's own verified detections. The actual vulnerable endpoints are completely outside the set of unverified detections. This suggests that banner-based detections, also widely used in academia, are completely unreliable. We confirm these patterns by analyzing the detections of ONYPHE, a competitor of Shodan. Our work has implications for industry users, policymakers, as well as the many academic researchers who rely on these services. Our study also is intended to increase awareness of the shortcomings of vulnerability tagging services, and it is a call for action to advance and standardize such services to make them more trustworthy.

1 Introduction

For about a decade now, Internet-wide scanning tools have been used to detect vulnerable hosts, both in industry and academia. There are two main approaches. One is based on interpreting banner data to determine that the host is running a vulnerable software version, the other relies on sending specific payloads, custom headers, and other crafted packets designed to probe the host for the presence of a vulnerability. The payload-based approach is seen as more reliable, but both techniques are used in practice. Services like Shodan [47], BinaryEdge [5], ONYPHE [41] or LeakIX [30], scan and tag hosts for specific vulnerabilities, usually in the form of CVE identifiers (Common Vulnerabilities and Exposures [39]).

An important use case for these services is that network defenders can run simple queries to detect vulnerable Internet-facing hosts. For example, the United States Cybersecurity and Infrastructure Security Agency (CISA) offers reports to other federal agencies using market-leader Shodan to continuously monitor and assess Internet-facing network assets and evaluate their vulnerability status [11]. Academic researchers have also relied on these services for collecting data (e.g., [19, 32]). Shodan is referenced in conjunction with CVE in around 1,400 research papers, according to Google Scholar.

While Shodan is widely used and, explicitly or implicitly, trusted for data collection on vulnerable hosts, earlier research did expose limitations. Shodan misses detections compared to more intrusive methods (e.g., Nmap scans [29]), privileged vantage points (e.g., Internet exchange traffic [3]), and more bespoke approaches (e.g., AI-based analysis of web interfaces [44]). These are not apples-to-apples comparisons, however. The alternative approaches are not as scalable as banner-based and payload-based methods, which Shodan uses, so they are not substitutes.

Given that many governments, companies and researchers use Shodan, it is important we gain a better understanding of the accuracy and completeness of its CVE detection. In the absence of ground truth about which endpoints are actually vulnerable, a perennial problem for vulnerability scanning research, our goal is to conduct an independent white-box measurement and compare the results. Shodan deploys two methods: banner-based and payload-based detection – which Shodan tags as "unverified" and "verified" CVEs, respec-

tively. We conduct a systematic comparison of the verified and unverified CVE detections against our independent and synchronous payload-based measurements using Nuclei templates [37]. The templates are designed to generate requests based on specific vulnerability checks and payloads. We repeatedly scanned 104,930 endpoints (IP:port pairs), all included in Shodan's scans, for 37 CVEs. Of these, 20 CVEs are part of Shodan's banner-based scans (unverified) and 17 CVEs are included in their payload-based scans (verified).

We then quantified the level of agreement among the results. Most of Shodan's verified detections are confirmed by Nuclei. Yet, Nuclei detects many more vulnerable endpoints, raising concerns about false negatives in Shodan's results. When looking at the unverified CVE detection, Nuclei finds that over 95% of these are false positives. Remarkably, Shodan implicitly also confirms the unreliability of those detections, as for two CVEs it runs both banner-based and payload-based scans. There is zero overlap among the results, confirming Nuclei's findings.

To corroborate the patterns we found. we repeated this analysis for a second commercial service in this market, ONYPHE. And we found very similar results that each method displays a different perspective on the vulnerability landscape. We also checked the agreement between ONYPHE and Shodan and found that they only agree on a minority of endpoints, rendering the results of each other as mostly consisting out of false negatives and false positives.

In sum, we make the following contributions:

- We use Nuclei to conduct the first independent evaluation of the vulnerability tagging performance of Shodan, as well as its competitor ONYPHE. We quantify the level of agreement among the three scanning services across 37 CVEs.
- For payload-based detections, we uncover large discrepancies. While there is a core of agreed-upon detections, for a much large set there is disagreement, raising concerns about false positives and false negatives for industry users relying on these services for attack surface monitoring.
- For banner-based detections, we find that they are extremely unreliable, to the point of being useless. Over 95% consists of false positives, rendering them unusable even as starting points for further analysis by network defenders. This also suggests the widespread use of banner-based vulnerability detection in academic work is highly problematic.
- We discuss implications of our findings for industry, government and academia and think of ways to move forward. We argue for more transparency on the performance of such services for those enterprises and governments that relying on them for attack surface monitoring.

The rest of the paper is organized as follows. Section 2 provides the necessary background information, while Section 3 presents our methodology for assessing vulnerability tagging services. Section 4 presents our measurement results. In Section 5, we discuss our results and their implications; in Section 6, we describe the limitations of our study, and in Section 7, we describe previous work. Finally, we conclude in Section 8.

2 Background

Shodan [47] is a commercial service providing a search engine based on Internet-wide scanning data. There are a range of competitors ([5, 8, 21, 30, 41, 56]) offering similar services to monitor the attack surface of organizations. Typically, the data provided by these search engines includes general information about hosts (e.g., country, organization, etc.), which ports are open, what services are running on them, and whether certain vulnerabilities are present.

To gather this information, these platforms typically employ a number of scanning tools. During the first step, they take advantage of fast open-ports network scanners, such as ZMap [18] or MASSCAN [24], that utilize enhanced SYN scanning methods to detect reachable hosts and open Transmission Control Protocol (TCP) and Universal Datagram Protocol (UDP) ports [35].

During the second step, the service performs a full handshake either using the protocol based on the IANA-assigned services list [14] or by deducing the protocol dynamically [25]. At this step, the service may also perform some further communications, e.g., send some application protocol-specific requests to get additional information like the version of the protocol from banner data. Over the years, researchers [4, 51] have been attempting to understand the Shodan scanning behaviors since it mostly acts as a black box to users. They ran virtual machines worldwide and discovered that Shodan allocates different scanning ports and scanning priorities across its scanners. Each VM received, on average, 176 scans from Shodan per day [4]. Tundis et al. [51] ran 8 honeypots and observed that Shodan discovered all of their services within 31 days. They also pointed out that the longest scan interval between two scans is around 15 days.

In the third step, the search engines analyze the obtained information with algorithms, which typically constitutes their "know-how". They annotate the hosts with *tags* or *annotations*, which can be later queried by users. Many of the services, including Shodan, provide information about possible vulnerabilities in the corresponding services. There are two main approaches to collecting the information that can be later used to assign those tags. The first employs the information from banners to extract the Common Platform Enumeration (CPE) [38] string, which is used to identify software and its version and that can be used to find vulnerabilities associated with it. The second approach implies the usage of

Table 1: Number of CVEs covered by different scan types

	# Shodan CVE	# CVE with Templates	# Validated Templates	# Unvalidated Templates	# CVE Selected
Payload	38	20	3	14	17
Banner	8,199	83	20	0	20
Total	8,237	103	23	14	37

CVE-specific payloads, custom headers, and other parameters designed to probe the target and confirm the presence of a vulnerability.

In the case of Shodan, the hosts found using the banner-based approach are tagged as *CVE:unverified*, indicating that Shodan has detected the potential presence of a CVE by associating the version of the software with known issues of that version. The documentation further states: "Unverified vulnerabilities can have significant false positives depending on the device/ software so they typically require additional verification to make sure the service is vulnerable. They should be seen as a starting point for further investigation" [48]. The results from payload-verified methods are tagged as *verified*. Shodan customer support informed us that these methods are based on "The tests are based on public research/tools for checking vulnerabilities."

3 Methodology

Figure 1 outlines the workflow of our study. There are two phases: (I) CVE Selection and (II) Vulnerability Scanning. The goal of Phase I is to select a set of CVEs that are scanned at Phase II. Our scanning toolchain is based on Nuclei [36], a fast and customizable vulnerability scanner. The procedure for performing a scan and analyzing the obtained data is specified in a *template*. Due to the simple YAML-based Domain Specific Language (DSL) used to develop templates, it has become very popular among security researchers. The templates are contributed by a large community of volunteers and stored in a GitHub repository [37]. As of June 2024, Nuclei offers 2,511 templates for detecting different CVEs [37].

3.1 Phase I: CVE Selection

Step 1a. Select CVEs from Shodan: We employed the `stats` Application Programming Interface (API) provided by Shodan to extract all CVE-IDs tracked by the platform. In total, Shodan tracks 8,237 CVE tags. Of these, just 38 are tagged as *verified*, so based on payload-based scans, while 8,199 are tagged as *unverified*, so based on banner data (see first column in Table 1).

Step 1b. Select CVEs from Nuclei: We retrieved a list of all Nuclei Templates from the official *nuclei-templates* repository [37] on May 10, 2024. We extracted the corresponding CVE-IDs from the *cves.json* file in the repository, which specifies the CVEs that these templates support. As of May 2024, the Nuclei repository contained 2,437 templates that are specifically dedicated to detecting CVEs.

Step 2. Find CVE Intersection: We computed the intersection between both sets of CVE-IDs. There are 103 CVE-IDs present on both platforms. Of these, 20 CVEs are tagged as verified by Shodan, while 83 are identified as unverified (see the second column in Table 1).

Step 3. Validate Templates: Given that Nuclei templates are community-developed, it is crucial to validate their effectiveness in detecting the specified vulnerabilities. To accomplish this, we utilized Docker [15] to set up isolated, portable environments known as *containers* against which we can test the Nuclei templates. To speed up the process of finding relevant Docker environment specifications, we utilized the Vulnhub [53] data. Vulnhub is a GitHub repository that stores a collection of pre-built vulnerable docker environments for specific CVEs. As of June 2024, it provided environment-building specifications for 204 CVEs.

Out of the 83 Shodan CVE detections based on banner data, the Vulnhub project provides Docker environments for 25 of them. We deployed these 25 Docker environments and manually validated the performance of the Nuclei templates. A total of 7 Nuclei templates failed to confirm the presence of vulnerabilities in their respective vulnerable environments. We conducted a detailed investigation of these cases. For 5 CVEs, we identified the cause and made modifications to improve the templates' detection capabilities. For *CVE-2014-3704*, *CVE-2018-7600*, and *CVE-2019-3396*, we adjusted the headers either in the sent requests or in the matching conditions. For the *CVE-2021-41773* template, a modification to the URL path was required. In the case of *CVE-2018-18778*, we did not alter the template itself but instead modified the input format. The original template executed HTTP requests but did not revert to HTTPS when an error occurred. To ensure both protocols were checked, we manually adjusted the input file to specify both HTTP and HTTPS explicitly. The template for *CVE-2018-19518* failed to detect the associated vulnerability and no straightforward remediation was available, so we excluded it. Another template, for *CVE-2018-7602*, was excluded because it required knowing the username and password for the target service. So this left us with 23 validated templates. We excluded an additional 3 templates (for *CVE-2022-0543*, *CVE-2022-24706*, and *CVE-2020-1938*) because they required additional interactions with target hosts, potentially resulting in indefinite scan durations. Ultimately, we selected 20 CVEs from the Shodan banner-based category, for which the corresponding Nuclei templates were deemed valid and effective.

Among the 20 Shodan payload-based CVE detections, only 2 had corresponding Nuclei templates with compatible Docker environments available from Vulnhub. To expand

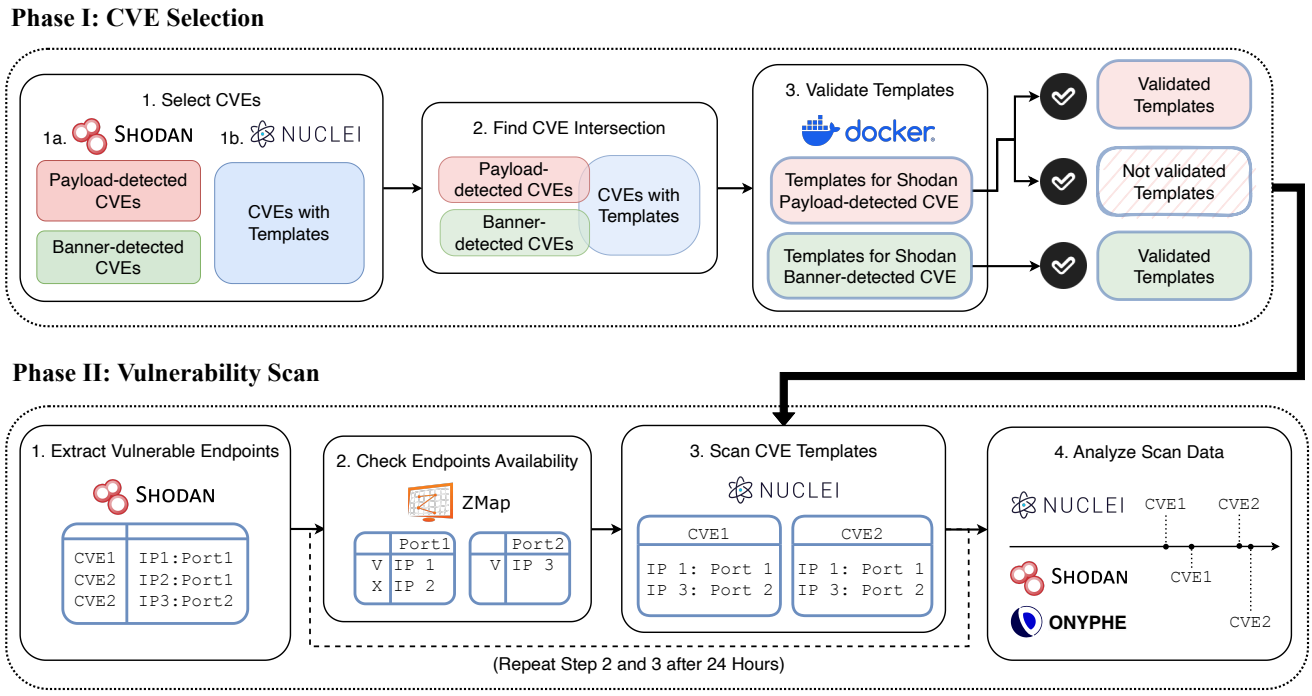

Figure 1: Research Workflow.

the number of templates we could validate, we explored official Docker repositories and identified suitable environments for additional 4 templates. We then tested these 6 templates.

Of the 6 templates tested, 4 passed the validation process. Notably, the *CVE-2015-2080* template failed due to an illegal character in the header field, intended to trigger the vulnerability. However, this illegal character prevented Nuclei from sending the correct packets. This issue has been reported to the Nuclei development team for further investigation. Furthermore, we excluded the template for *CVE-2023-33246* from the validated set because it required external interactions with target hosts, potentially leading to indefinite scan durations.

The remaining 14 templates are not validated because the associated software, e.g., Microsoft Exchange Server, is extremely difficult to set up in a Docker environment. However, we decided to include these CVEs in our final set, so as to have a broader comparison of payload-based CVE scans, even though we were unable to independently validate them. We assume their performance is comparable to the ones we were able to validate. In the end, for Shodan payload-based CVE scans, we selected 3 CVEs with validated templates and 14 with not-validated templates.

In conclusion, our final set of CVEs includes 20 Shodan banner-based CVEs and 17 Shodan payload-based CVEs. Table 2 in Appendix A provides a detailed overview of these 37 CVEs and their characteristics. The CVEs span a range of severity levels, with CVSS scores varying from 5.3 to 10.0, indicating a broad spectrum of risk from moderate to critical vulnerabilities. The average CVSS score among these CVEs is 8.9, reflecting a predominance of high-severity vulnerabilities that pose significant security risks across the affected endpoints.

3.2 Phase II: Vulnerability Scan

To collect data points to compare Shodan CVE detections with our Nuclei results, we followed four steps.

Step 1. Extract Vulnerable Endpoints: We need to build a set of endpoints to scan with both Nuclei and Shodan – where endpoints are defined as `IP:port` tuples. To enable a fair comparison, we want to only include endpoints that we know to be reachable and scannable by Shodan. For this reason, we build our set of endpoints from recent Shodan scan results – thereby ensuring that Shodan can reach and scan each endpoint. If we were to choose random IPv4 addresses or networks, it would likely contain some IP space where Shodan is blocked by the network operator, while our ad hoc scans might have normal access. This would bias the comparison against Shodan.

To build the set of endpoints, we used Shodan's `search` API to extract the set of endpoints that it had detected as vulnerable for any of the 37 selected CVEs in the first three weeks of July 2024, just before the start of our scanning period. Shodan performs scans irregularly. According to Tundis et al. [51], the longest scan interval is around 15 days. So we

decided to employ a slightly longer period of 21 days to maximize the number of endpoints that have an updated scan result.

Some CVEs are detected at a large scale: hundreds of thousands of vulnerable endpoints. For instance, we collected 956,543 `IP:port` tuples for *CVE-2017-15715* and 721,310 instances for *CVE-2021-40438*. As we want to perform our scans in a reasonable time, we restricted number of selected endpoints to 15,000, by randomly sampling them from all results for that CVE.

We combined all selected endpoints into a single set, obtaining a total of 105,232 entries. We excluded the entries with IPv6 addresses, since ZMap only accepts IPv4 addresses. This generates the final dataset containing 104,930 endpoints (unique IP:port combinations), which we call the *superset*. Across the set, there are 1,244 unique ports and 94,265 IPv4 addresses in this final superset.

Step 2. Check Endpoint Availability: We employed ZMap [18] to scan the IPs and ports from the superset obtained in the previous step. We employed the ZMap version that has not yet got the multi-port scanning functionality, therefore, we grouped our dataset by a port number and run the ZMap scan for each group and port number. We ran 15 instances of ZMap in parallel, limiting the speed of each scanning process to 1,000 packets per second. It took us approximately 14 minutes to check all the endpoints from our dataset. The results of the scan were merged into one resulting file in a random order.

Step 3. Scan CVE Templates: From July 28 to August 17, 2024, we scanned all endpoints obtained during the previous step with all 37 Nuclei templates. Since the maximum scan interval for Shodan is 15 days [51], our three week collection period would ensure we would capture a fresh Shodan scan result for each endpoint to compare our scan results against. We applied these templates sequentially, meaning that at each point in time we only scanned one vulnerability for all available endpoints. We configured Nuclei to analyze 150,000 endpoints in parallel (async mode). It took 1.5 hours on average to complete the scan of all available endpoints for one CVE, and about 3 days and 4 hours to finish scanning for all selected CVEs.

Step 4. Analyze Scan Data: Any scan is a snapshot in time. If the scan result of Nuclei is different from Shodan, this might be caused by a difference in timing. In between the scans, the situation at the endpoint might have been changed, e.g., it might be patched. To remove this timing factor, we conducted five consecutive scan cycles. This way, any scan by Shodan would be 'sandwiched' between two or more Nuclei scans just before and after the Shodan scan. If the subsequent Nuclei scans present a consistent result, then a different Shodan result is unlikely to be caused by a sudden change at the endpoint itself.

After the scanning of all 37 CVE templates is finished, we let our scanner sleep for 24 hours. Then, we resume the scan starting from *Step 2*. We repeated this procedure for 5 times between July 28 and August 17, 2024. For the same period of time, we also re-extracted the Shodan data about all the endpoints from our superset. Then, we compared both sets of results. Additionally, we obtained ONYPHE data from the same timeframe and conducted a further comparative analysis against both the Shodan and Nuclei datasets.

4 Measurement Results

In this section, we present the results of our measurements that aim to assess the reliability of Shodan CVE detections. Specifically, Section 4.1 presents an overview of the performance of our scans. Then, we dive deep into analyzing the agreement between Nuclei and Shodan's results (Section 4.2). We then turn to where we found inconsistent scan results (Section 4.3). Finally, in Section 4.4, we conduct a comparison with ONYPHE to provide additional insights on the landscape of scanning from the user perspective.

4.1 Scan Performance

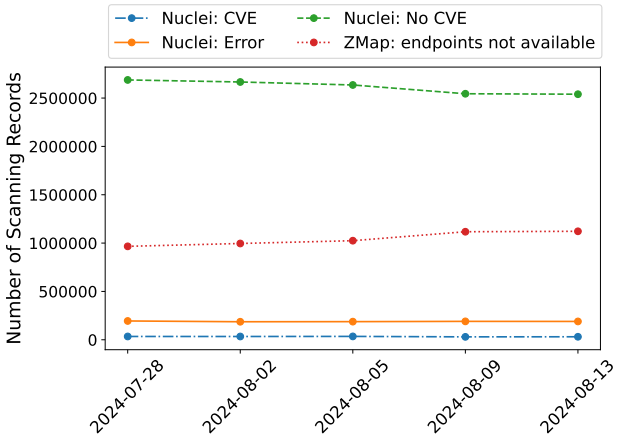

Figure 2: Summary of scanning records by scan.

Each of the 104,930 endpoints in our set is scanned 5 times with all 37 CVE templates. Thus, in total, we collected 19,412,050 scan records. Figure 2 illustrates the statistics across these scans, visualizing the number of ZMap-detected not reachable endpoints and the Nuclei detection outcomes, which are categorized into cases where some error occurred and cases with a clear outcome (CVE and No CVE).

We present the categorization of scan records for each CVE in Figure 3. This figure illustrates the overall composition of scan results for each CVE across the entire scanning period, including the number of records where ZMap identified endpoints as unresponsive, the instances where Nuclei returned errors (indicating indeterminate CVE status), and the records

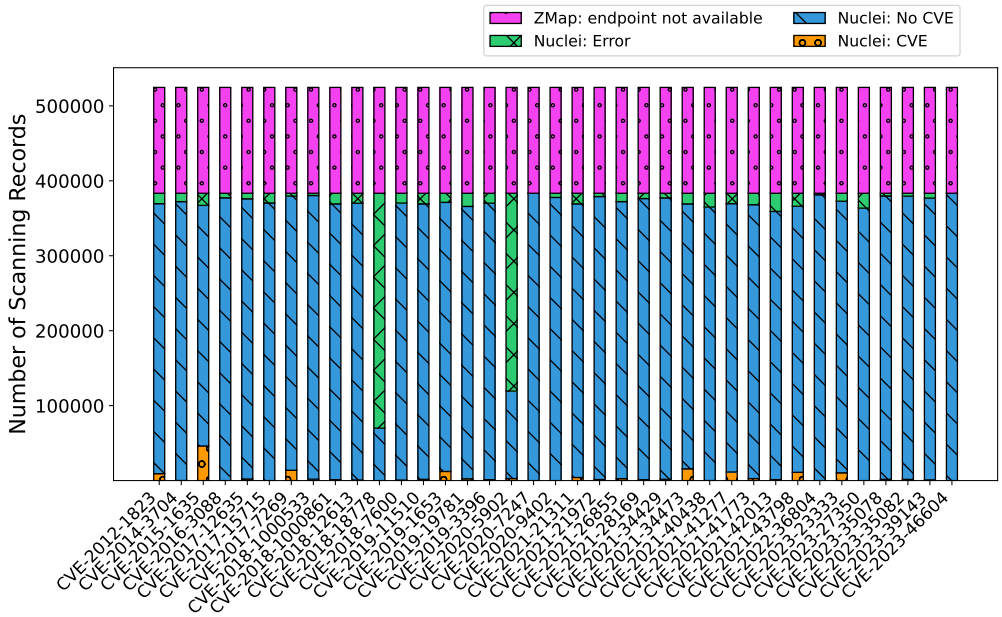

Figure 3: Categorization of CVE scan records by failure types and detection outcomes in ZMap and Nuclei scans.

where Nuclei succesfully completed the scan and returned a result of CVE detected or not detected. As shown in the figure, all CVEs share a very similar share of ZMap "unavailable endpoint" records (141,251 scanning records). For the majority of CVEs, the predominant outcome category is "No CVE," with an average of 353,338 records per CVE falling into this category. The highest count of "No CVE" detected records is observed for *CVE-2020-7247*, with 383,310 records, while *CVE-2018-18778* has the fewest, at 69,933 records.

Interestingly, Nuclei error accounts for only a small fraction of all results, an average of 25,629 records per CVE. However, specific CVEs, such as *CVE-2018-18778* (a vulnerability in ACME mini_httpd) and *CVE-2020-5902* (a vulnerability affecting F5 BIG-IP-related products), exhibit a disproportionately large number of Nuclei error records, with 313,464 and 263,972 records, respectively. This indicates that Nuclei could not conclusively determine the presence of these CVEs due to issues such as fallback failures. *CVE-2018-18778* also encountered errors related to malformed HTTP responses or incorrect status codes, while *CVE-2020-5902* experienced errors associated with the failure to parse the response header. While the number of scan records indicating a detected CVE presence is relatively small compared to other categories, these records provide critical insights into the vulnerability landscape as observed by Nuclei. The most frequently detected CVE among our superset of endpoints is *CVE-2015-1635*, a vulnerability related to Microsoft Windows that carries a CVSS score of 10 (HIGH), underscoring its prevalence and severity. Of the 37 CVEs evaluated, seven have more than 10,000 scan records indicating a confirmed CVE presence.

From the 19,412,050 scan records, we dropped the scan records with *ZMap: endpoint not available* and *Nuclei: Error*. Then, we conducted a detailed analysis for each CVE to determine the number of endpoints with consistent vulnerability detection results versus those with inconsistent results. To illustrate, we pick *CVE-2021-21972* as an example, depicted in Figure 4. For instance, for the 13th endpoint (marked with the dotted line), Nuclei consistently detects it as vulnerable across four consecutive scans. For endpoint number 10, we can see an endpoint where Nuclei consistently reported no CVE.

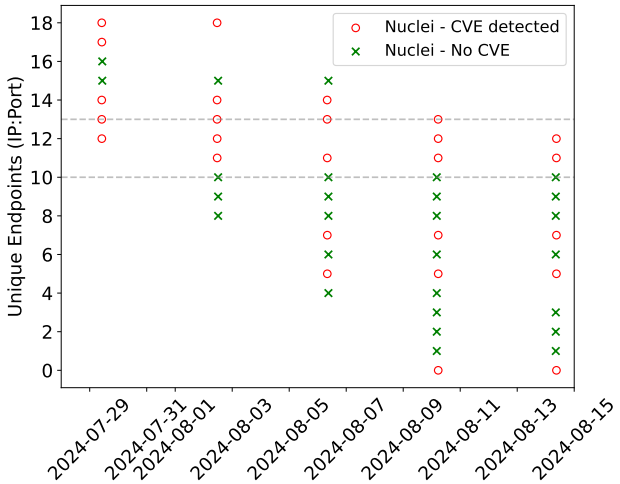

Figure 4: Nuclei scan records for *CVE-2021-21972*.

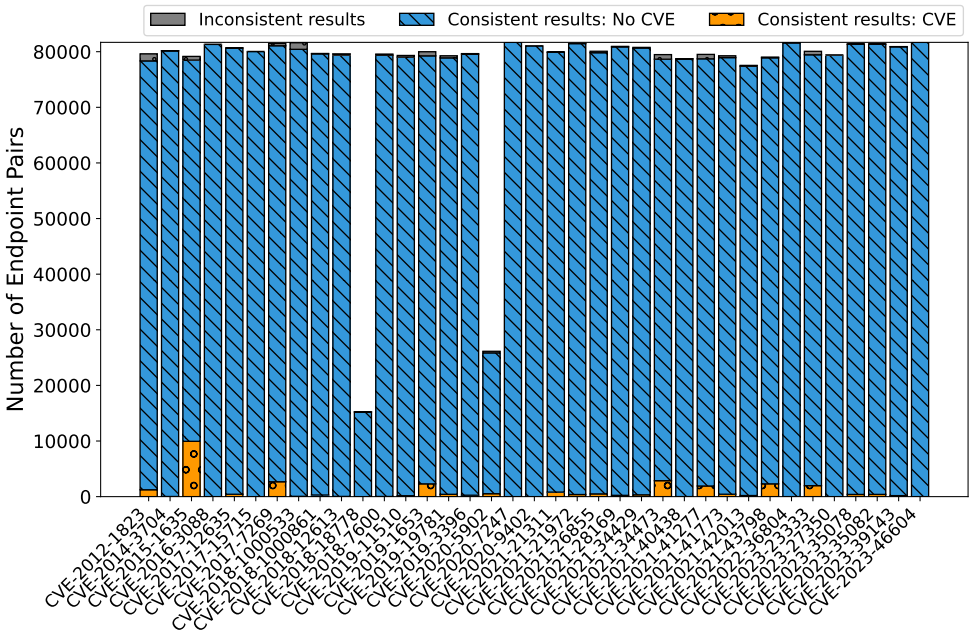

Figure 5: Classification of endpoints by consistent and inconsistent responses and their vulnerability status over time.

Figure 5 presents the distribution of consistent and inconsistent results across CVEs. It shows that only a very small fraction of endpoints have inconsistent results. We further distinguished between those consistently showing "No CVE" detected and those consistently indicating the presence of a CVE. Notably, *CVE-2018-18778* and *CVE-2020-5902* show the lowest number of consistently responsive endpoints, primarily because a significant portion of the scan records for these CVEs are categorized as "Nuclei: Error" (see Figure 3).

To ensure accurate analysis, we exclude endpoints with inconsistent results due to their fluctuating vulnerability status. Such variability makes it difficult to draw reliable conclusions or compare them with data from other scanning sources, as these endpoints may change their status over time, and different scanners may yield varying results.

We further queried Shodan to get the data on whether the endpoints from our superset were vulnerable to our 37 selected CVEs during the same period as our Nuclei scans (July 28 to August 17, 2024) and obtained 1,533,026 entries of scan records for 1,290,450 endpoints. Over the 21-day period of Nuclei scan data collection, we ran our scans five times. This scanning frequency exceeds that of Shodan, which only recorded a single scan for each endpoint within the intersecting dataset. To facilitate a comparative analysis between the two data sources, we map Shodan scan records against the preceding and subsequent Nuclei scans.

We started with selecting all endpoints (IP:port combinations) that were labeled as vulnerable by Shodan. Shodan detected a CVE or multiple CVEs for 40,349 endpoints that overlap with Nuclei scan results. For these endpoints, Nuclei reached the same conclusion across all its scan cycles.

4.2 Agreement in Vulnerability Detection

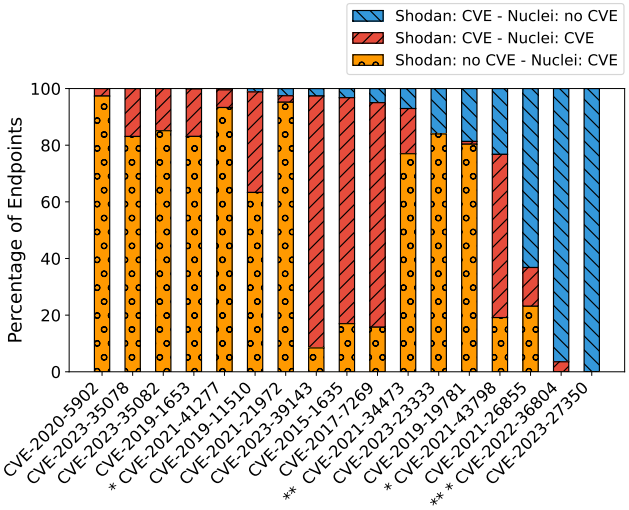

Figure 6: Percentage of endpoints classified as vulnerable or non-vulnerable by Nuclei and/or Shodan for payload-detected CVEs.

For the 40,349 endpoints where Nuclei produced consistent scan results, we now analyze to what extent our Nuclei scans agree with Shodan on the vulnerability status of an endpoint.

We present the results for Shodan's payload-based scans

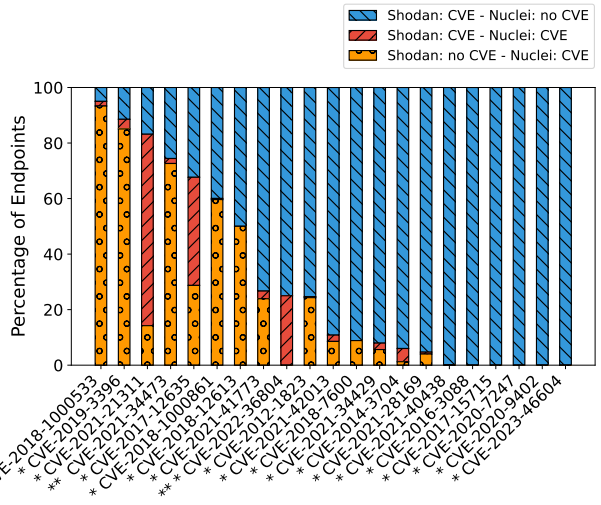

Figure 7: Percentage of endpoints classified as vulnerable or non-vulnerable by Nuclei and/or Shodan for banner-detected CVEs.

separately from the banner-based ones since those methods are very different, and the latter is seen as less reliable.

Figure 6 shows the Nuclei results for all Shodan's payload-based detections ("verified"). The results show noticeable variation in agreement across different CVEs. Some CVEs exhibit high agreement between Shodan and Nuclei (areas marked in red). In others, Nuclei confirms the detections of Shodan, but finds many more vulnerable endpoints that Shodan missed (orange). Finally, for other CVEs Nuclei contradicts the detections of Shodan, finding no vulnerability (blue).

For 11 out of 17 CVEs (65%), we find that Nuclei does not substantially contradict the Shodan results, with less than 10% of the Shodan detections not confirmed by Nuclei. That said, for 10 CVEs, Nuclei finds many more vulnerable endpoints than Shodan: a factor of 2-36 more vulnerable endpoints depending on the CVE. Across all results, Nuclei reports 13,275 more detections than Shodan. This raises the question of whether Shodan verified results suffer from a large false-negative rate. Since there is no ground truth, we cannot ascertain that this is the case. The alternate explanation is that the Nuclei detections are incorrect.

These differences seem unrelated to whether the template is validated or not. There are three validated templates (marked by a single asterisk at the beginning of the CVE-ID). In the first case Nuclei confirms Shodan's detections, but discovers a factor of 10 more vulnerable endpoints (*CVE-2021-41277*). The second case reports a majority of consistent results (*CVE-2021-43798*), while the third case reports overwhelmingly contradictory results, where Nuclei disagrees for almost all endpoints with Shodan's detection of a CVE (*CVE-2022-36804*).

We further investigate the underlying factors contributing to the varying levels of detection agreement among these three CVEs. For *CVE-2021-41277*, it is a vulnerability in Metabase with GeoJSON map support that allows potential local file inclusion via specific path queries, such as path with string: */api/geojson?url*. Similarly, *CVE-2021-43798* is a directory traversal vulnerability affecting Grafana services, detectable by probing paths including string: */public/plugins/<"plugin-id">/* (Nuclei template for this CVE uses "alertlist" as the plugin-id). Nuclei scans for these CVEs both rely on targeting paths associated with default service configurations. In contrast, detecting *CVE-2022-36804* in Bitbucket requires access to a public repository or read permissions for a private repository. As noted by Bitbucket Support, temporary mitigation involves globally disabling public repositories: "If you're unable to upgrade Bitbucket, a temporary mitigation step is to turn off public repositories globally..." [6]. Nuclei's template attempts to access the latest project within the targeted repository; however, the detection rate may be low because vulnerable services can be quickly mitigated by disabling public repository access. This makes confirming the presence of the vulnerability challenging, despite its high severity (CVSS 8.8). We speculate that the disagreement in detection results for this CVE is due to differing detection methods used by the two scanning tools.

Next, we compare Shodan's banner-based CVE detections ("unverified") with Nuclei detections (Figure 7). In this comparison, almost all Nuclei templates were validated (as marked by an asterisk at the beginning of each CVE-ID) except one template for *CVE-2021-34473*. The first thing we can see is that Nuclei disagrees with the bulk of the Shodan unverified detections (marked by the blue area). Only for 3 out of 21 CVEs is there a meaningful degree of agreement, meaning that Nuclei confirms some portion of Shodan's banner-based detection: *CVE-2021-21311*, *CVE-2017-12635*, and *CVE-2022-36804*. For 18 out of 21 CVEs, Nuclei contradicts over 95% of the Shodan banner-based detections. While it is well-known that banner-based detections are less reliable, these findings suggest that they consist almost completely out of false positives.

There are two CVEs (*CVE-2022-36804* and *CVE-2021-34473*) where Shodan conducts both types of scans: payload-based and banner-based. So its search engine reports two separate sets of scan results for each CVE: one labeled as verified and one as unverified. Stunningly, we find zero overlap in endpoints between the verified and unverified results. So even Shodan's own scans contradict completely the unverified CVE detections.

Now, one might reasonably expect that banner-based detections are overly broad. The software version in banner data might indicate a vulnerability that in practice is no longer there or that has been mitigated to the extent that it cannot be detected or exploited. The value of the banner-based detections would then be that some fraction of them will be true

positives. This is also how Shodan positions the banner-based (unverified) CVE detections: "Unverified vulnerabilities can have significant false positives [...] They should be seen as a starting point for further investigation". Following this logic, then Nuclei's detections should fall within the set of Shodan unverified detections. Yet that is not what we see. The bulk of the Nuclei detections are not detected by Shodan's banner-based scans (the areas in orange), so they are not a subset of the banner-based detections. For the two CVEs where Shodan has both verified and unverified detections, the situation is even more stark: there is zero overlap. This means that the unverified detections do not seem a useful "starting point for further investigation". For researchers using Shodan's unverified CVE detection as data, the problem is arguably even worse. The doubts about data quality seem to render these detections unusable for scientific research.

Overall banner-based detection seems highly problematic, yet for three templates the performance is better and with some merit, as discussed above (*CVE-2021-21311*, *CVE-2017-12635*, and *CVE-2022-36804*). What might explain why banner-based detections produce better results in these cases? One speculation is that some CVEs are associated with much more specific types of products, as indicated by their CPE (Common Platform Enumerations) description. This specificity would allow for more targeted matching with banner data. For *CVE-2021-21311*, 80.37% of the vulnerable endpoints identified by Shodan were confirmed by Nuclei's payload-based scans. The CPE strings for this CVE are closely linked to *Adminer* database software. For the two other CVEs, the CPE strings match with *CouchDB* and *Atlassian Bitbucket*, respectively.

The other CVEs, where Nuclei mostly contradicted Shodan detections, are associated with CPE strings that encompass a broad spectrum of potential CVEs. For instance, for *CVE-2020-7247* the CPE string *cpe:2.3:o:canonical:ubuntu_linux:19.10:*:*:*:*:*:*:** corresponds to 446 different CVEs in the NVD CPE database [38]. Any of these CVEs could potentially exist, or not, on a device that matches the service and version criteria described in the CPE string.

Take-Away. In sum, for Shodan payload-based detections (verified CVEs), defenders might face massive underreporting of vulnerable endpoints, while for Shodan banner-based detections (unverified) they face massive overreporting. Given that payload-based methods are more reliable than banner-based, it seems the overwhelming majority of banner-based detections are false positives. Even Shodan itself confirms this implicitly, since for two CVEs, their verified and unverified detections have no overlap whatsoever. Comparing them corroborates our finding that banner-based detection is extremely unreliable. In the Discussion (Section 5), we will reflect on the implications of these findings for both security professionals as well as academic researchers relying on these detections.

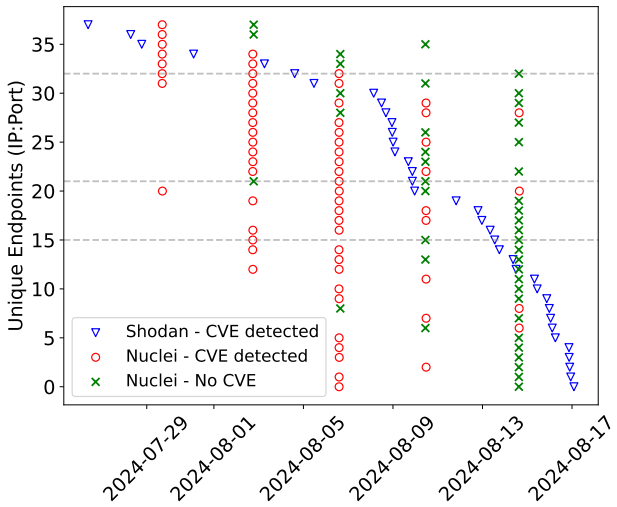

Figure 8: The overlap between Shodan and Nuclei scan records for *CVE-2021-43798*.

4.3 Inconsistent Scan Results

A total of 739 Shodan scan records (1.5% of all records), were identified as overlapping with *Nuclei: CVE* and *Nuclei: No CVE* scan records, yet they exhibited inconsistent results in Nuclei scans across the scanning period. These Shodan scan records contain 729 endpoints for 21 different CVEs. Of these scan records, there are 682 of them associated with Shodan payload-based CVE labels, and 57 were linked to banner-based CVE labels. To exemplify these inconsistencies, we compared the Shodan with Nuclei scan results for *CVE-2021-43798*, which corresponds to the Shodan payload-based CVE tag and has the validated Nuclei template. Figure 8 shows the scan results timeline, where Y-axis values correspond to the index number of the endpoints (IP:port) and the X-axis is the date. We can note from this figure that Shodan scans the same CVE on different dates. Thus, we have some Shodan hits that are available before our first scan and after our last scan.

We concentrate only on discussing Shodan hits that are between our first and last scan for this CVE, i.e., with index values between 13 and 34. We classify these cases by comparing the Shodan and Nuclei *immediate* scan results, i.e., one Nuclei before and one after the corresponding Shodan scan.

Immediate Consistent. Some of the cases, e.g. 32, can be considered as showing consistent results. In this case, the immediate Nuclei scans before and after Shodan show the presence of CVE. The Nuclei scan at the beginning also detected CVE. However, the last Nuclei scan does not show the presence of the CVE. One potential explanation for this is that the vulnerability on this endpoint has been patched.

Immediate Inconsistent. There are also cases, e.g., 15, where Shodan and Nuclei disagree with the results. However, we have only the cases where there is only one Nuclei scan after

Shodan, so we cannot conclude if the same behavior remains. As future work, it would be interesting to perform a more longitudinal study that would cover several Shodan scans.

Inconsistent. Finally, some cases, e.g., 21, show very inconsistent behavior, with Shodan and a Nuclei scan detecting CVE and other Nuclei scans not observing it. Such behavior may point to the IP churn in the corresponding network.

Despite these categorizations, the inconsistent Nuclei scan results account for only a small fraction (1.5%) of the intersecting records with Shodan and do not significantly impact our findings. In the future, we plan to conduct more longitudinal studies for a more comprehensive characterization.

4.4 Comparison with ONYPHE

To provide an alternative viewpoint, we have contacted ONYPHE [41], a competitor of Shodan. ONYPHE focuses on the CVEs in the CISA KEV (Known Exploited Vulnerabilities) Catalog [12]. They adopt a selective approach in choosing CVEs, prioritizing those that are exploited at scale, as vulnerabilities that remain unexploited offer limited value from a defensive standpoint. They employ both banner-based (`tag:vulnerableversion`) and payload-based (`tag:vulnerable`) methods to identify vulnerable endpoints, covering a total of 115 CVEs.[1] Unlike Shodan, ONYPHE performs weekly scans for their targeting 115 critical CVEs. ONYPHE generously shared their detection results with us, after we provided them with our superset of 104K endpoints and CVE list. The overlap between the CVEs that ONYPHE tracks and our set of 37 templates consists of 10 CVEs for those where they both have results. These detections were collected in the same period where we performed our experiments: July 28 to August 17, 2024. This dataset contains a total of 16,714 detections across 1,053 endpoints. The data for the 10 overlapping CVEs contains results from both banner-based and payload-based vulnerability detection methods employed by ONYPHE. These are detailed in Table 2 in Appendix A. One important limitation to acknowledge is that the superset is derived from Shodan hits, i.e., endpoints scanned by Shodan for specific CVEs. However, ONYPHE conducts scans based on a substantially different set of CVEs, focusing on those deemed critical and actively exploited through their proprietary threat intelligence. Additionally, ONYPHE targets specific ports, resulting in limited overlap between their scans and those performed by Shodan.

We plot the comparison between our Nuclei results and the ONYPHE results in the same way as in our Shodan comparison. Figure 9 presents the results. The pattern looks remarkably similar. For 8 out of 10 CVEs, our scans do not contradict ONYPHE's detections. Less than 5% of the detections are not confirmed by Nuclei. Yet, just as with Shodan, for those 8

[1]The complete list of CVEs tracked by ONYPHE, along with their CVE selection policy, is available at https://www.onyphe.io/docs/dorkpedia/vulnscan-cve-list.

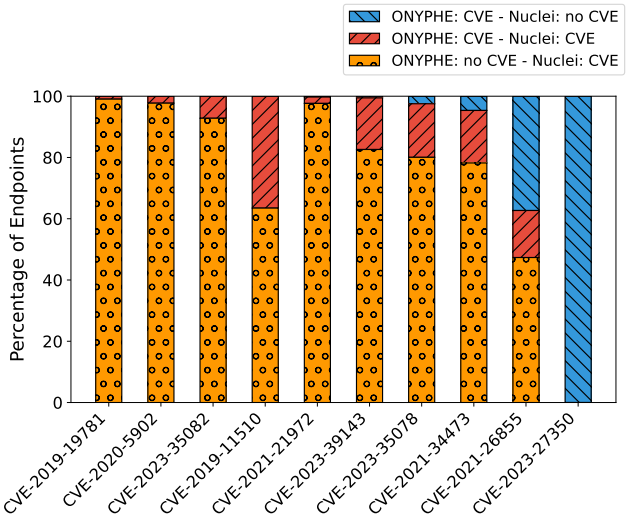

Figure 9: Percentage of endpoints identified as vulnerable or non-vulnerable by Nuclei compared to ONYPHE.

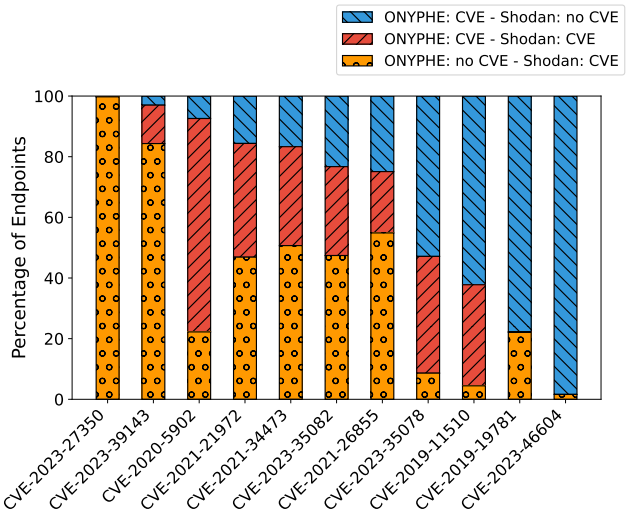

Figure 10: Percentage of endpoints identified as vulnerable by Shodan (payload-based CVE detections) compared to ONYPHE.

CVEs Nuclei detects substantially more vulnerable endpoints: ranging from 1.1 to 104 times more (as measured by dividing the area marked in orange by the areas in red and blue). Also, for *CVE-2023-27350*, we find no detections in Nuclei, while ONYPHE found three. The same result occurred in the comparison to Shodan, where Nuclei found none, while Shodan found 55 vulnerable endpoints (Figure 6). This strongly suggests that this Nuclei template is not valid.

Since this pattern is consistent across both Shodan and ONYPHE, it raises the question of whether our detection has a higher false positive rate. Perhaps a commercial provider is

more likely to report conservatively and avoid false positives to its clients? Still, since companies rely on these services for attack surface monitoring, our findings do confirm the urgency of investigating the potential for high false negative rates in future work.

We performed a direct comparison between ONYPHE and Shodan by analyzing all of Shodan's results alongside ONYPHE's CVE detection data during the same scanning period, from July 28 to August 17, 2024. Both platforms identified scan results for 11 common CVEs. The results are depicted in Figure 10. Surprisingly, the same pattern emerges, where both services see a large amount of false negatives in the other service. For 8 out of 11 CVEs, both services agree on fewer detections than they disagree on. The portion of the detections that both services agree on (the red area) is a bit higher than the portion of detections agreed between Nuclei and each of the services separately, but disagreement is still the dominant pattern.

Take-Away. Overall, although all three datasets rely on payload-based detection methods, actively checking the presence of CVEs, they exhibit varying levels of agreement depending on the CVE and the detection methods employed. Even among the two commercial services, each sees a significant amount of false positives and false negatives in the other service. This raises serious concerns about the accuracy of vulnerability detection in attack surface monitoring services.

5 Discussion

Many countries are adopting more stringent cybersecurity regulations. In the EU, for example, NIS2 (Network and Information Security Directive 2) [20] requires patch management policies to be implemented by all medium to large-sized organizations operating in sectors deemed essential or important. This will increase the demand for vulnerability tagging services. Already, such services are very widely used by organizations to monitor their attack surface for vulnerability management programs.

One example is market leader Shodan's Monitor service, which lets defenders register their IP ranges to be monitored for security-relevant events, like the presence of CVEs. A related use case is when oversight bodies or sectoral CERTs (computer emergency response teams) rely on these services to support their constituents. In the US, CISA relies on Shodan and other tools for vulnerability scanning as part of the "Cyber Hygiene services" it offers to other federal agencies [11] [1].

While the promise of Shodan is to "gain complete visibility into what you have connected" [46], in practice the professionals relying on these services will expect some inaccuracy. Yet they have no way of gauging its performance. As far as we know, there has been no independent testing of Shodan's CVE detections.

Our study found two different patterns, related to Shodan's verified and its unverified CVE results. Since these are based on different scanning methods – payload-based versus banner-based, respectively – it makes sense that they face different issues. Starting with the evaluation of the verified detections, we observed that our measurements confirmed the bulk of Shodan's CVE detections. Only 10% of the Shodan detections were not corroborated by our findings. However, our scans found many more vulnerable endpoints that Shodan had not detected. For 10 out of 17 CVEs, we found a factor of 2-36 more vulnerable endpoints. This raises questions about the promised "complete visibility". Even if clients take that with a grain of salt, they are unlikely to expect this level of potential inaccuracy. For clients, the question is how critical false negatives are for their use case. If they rely on Shodan detections for keeping their attack surface secure, it might slow down the patching processes, rather than accelerate it. In terms of false positives, our analysis suggests that the rate is low for most verified CVEs, which avoids burdening IT staff.

The situation is quite different for Shodan's "unverified", banner-based detections. For 18 out of 21 CVEs, Nuclei contradicts over 95% of the Shodan banner-based detections. Shodan says that these detections are known to contain "significant false positives," yet their value is to serve as a "starting point for further investigation". Our results question this use case, since nearly all of our CVE detections fall outside of Shodan's banner-based detections. So investigating the noisy Shodan CVE tags won't lead you to discover those. This would negatively impact the task.

We have focused our analysis on market leader Shodan, but these patterns are not unique to that service. We confirmed them by an analysis of ONYPHE's CVE detections for the same set of endpoints. Even when comparing the two commercial services directly, each sees a significant amount of false positives and false negatives in the results of the other service.

The limited overlap among the results of Nuclei, Shodan and ONYPHE is reminiscent of another area with a lot of industry and academic effort: threat intelligence. Numerous studies have looked at the indicators of compromise (IoCs) that are provided by different commercial and free sources. A consistent finding in that literature has been that there is very little overlap among the IoCs detected by each provider [31]. This even holds true for the high-end market leaders who claim to be tracking the same threat actors groups. One study found an overlap of less than 4% in the IoCs detected by these firms [7]. In other words, every threat intelligence provider sees only a limited slice of the attacker ecosystem. This has led to an industry practice where enterprises feel the need to acquire, on average, seven different threat intelligence services, to have a bit more confidence in their coverage [42]. We certainly found significantly higher overlap among the CVE tagging services than is found among threat intelligence providers, but it does seem that a single service might not be sufficient for adequate attack surface monitoring.

Overall, vulnerability tagging services play an increasingly

important role. The concerns that are raised by our findings are not meant to imply these services should be abandoned. They are indispensable in light of increasingly complex IT infrastructures and perennial problems like 'shadow IT'. Rather, we argue for improved transparency about performance and for awareness among practitioners and regulators about what these services actually provide. Our findings also argue for a strong commitment to improvement. Community-wide efforts to adopt state-of-the-art solutions and standardization will improve the quality of vulnerability tagging. Regulatory intervention may be required to give incentives to adopt best practices and improve the quality of vulnerability tagging, e.g., via certification under the EU's Cyber Resilience Act. Cyber insurance companies might also help establish which vulnerability tagging services are more accurate, based on their claims data around breaches.

Finally, our findings also serve to caution academic researchers. Hundreds, if not thousands, of papers rely on Shodan for detection of one kind or another. Though the CVE tags do not seem to be widely used in papers, our findings also question the use of banner metadata for CVE detection – which is a much more widespread practice (e.g., [49, 54]). There would be great value in a future study to better understand under what limited conditions banner-based detection is accurate enough for scientific purposes.

6 Limitations

Our study faces several limitations. First and foremost, we have no ground truth on the presence of the CVEs at the endpoints. This limitation is faced by all research on Internet-wide scans for vulnerabilities. Only direct contact with the administrators of the affected systems would get closer to ground truth [19], but that obviously does not scale. So, the best anyone can do is to provide white-box implementations of scanning methods that can be independently validated, replicated, and compared against the results of other toolchains.

A second limitation is that our evaluation is based on 37 CVEs. While this set contains variation in terms of the affected services and severity of the vulnerabilities (CVSS scores range: 5.3 — 10.0, average: 8.9), there may be other factors that become visible if the comparison is based on a larger set of CVEs.

Another limitation is that we can only speculate about why results for some CVEs were very similar between Nuclei, Shodan, and ONYPHE, while for others they were wildly different, even when using the same payload-based approach. We do know that small changes in the scanning templates can make a significant difference (Section 3.1). This issue is implicitly present in every vulnerability scanning study, though rarely surfaced. Future work might do sensitivity analysis to quantify the impact of certain changes.

Finally, payload-based scanning methods may not always accurately determine the absence of a CVE. Even if a scan does not detect a CVE, the vulnerability could still be present (e.g., the software is unpatched). However, other security measures, such as application firewall rules or network configurations, might prevent the payload from reaching the vulnerable code path. This potential discrepancy highlights that a "no CVE" result does not necessarily confirm the absence of a vulnerability but rather indicates that it was not detectable given the current conditions. Future work should assess the impact of these external factors on vulnerability scan assessments.

7 Related Work

This section summarizes related work on CVE identification in Internet-wide scan data. There are two categories based on the methodologies adopted to obtain CVEs: (i) mapping the CPE information to the corresponding CVE list, and (ii) finding CVEs through payloads (also called 'benign exploits'). Both have strengths and weaknesses in CVE disclosure.

7.1 Banner-based detection

Using CPE information to track CVEs is a way to understand the Internet-wide vulnerability landscape, though its accuracy is contested. Researchers either collected data themselves by building scanning tools [27] [10] or used scan data from public sources [22] [23] [40]. They then leveraged service version information or metadata to generate CPE strings and mapped them to a list of CVEs that may exist on target services.

Researchers have proposed various scanning infrastructures to collect host information. Kim et al. [27] proposed an Internet-scanning architecture that leveraged ZMap [18] and ZGrab [16] projects. They collected CPE information from packet payloads or banner information to identify CVEs on target hosts. Cigoj et al. [10] presented a scanning tool targeting web application vulnerabilities. They collected version information from websites' metadata and mapped it to CVEs in their database. Moghiss et al. [33] proposed a search engine for Internet-facing services, claiming their improvements include scan fail detection and using a small number of packets for their banner grabber.

Besides collecting host metadata, some researchers also leveraged public scan data for their vulnerability study. Genge et al. [22] [23] and O'Hare et al. [40] developed tools that leveraged metadata of hosts from public scan data, such as Shodan [47] and Censys [8], to generate CPE information and identify potential existing CVEs. They then used CVE data to analyze the CVE population and the CVSS score distribution. Laštovička et al. [29] used NetFlow [13] and Nmap [34] to conduct passive and active scans in their university network. They collected host information and formed CPE labels for detected hosts. Then, they compared the CVEs inferred from CPE information collected by passive and active scans with CVE tags from Shodan [47] for the same target IPs.

However, there are some downsides for CPE-inferred CVEs, which may cause the CVE results to become inaccurate. For example, banner or service version information can form multiple CPE combinations due to different naming schemes among vendors and devices. Ushakov et al. [52] developed an algorithm to map software products to related CPE entries to obtain CVE lists and assess security risks. Sanguino et al. [43] proposed a tool to help users find suitable CPE strings for target software to decrease the potential wrong CVE mapping. Thomas et al. [50] studied vulnerabilities in Siemens Industrial Control Systems (ICS) related devices. They discussed the inconsistency of product and vendor information presented in CPE format and between CVE and CPE information. Among the selected 207 CVEs, there were 15% of CVEs had affected devices in their description but were not entirely in their CPE lists. This inconsistency may cause a false negative or positive for understanding any potential risk.

Moreover, the formed CPE may reflect false CVEs due to backporting. West et al. [54] used the Censys dataset [8] to observe OpenSSH software updates for enterprises. They pointed out that version information may not be accurate for measuring how outdated the software is since some of them may apply a backport version, i.e., a new patched software based on an old version.

In this work, we use Nuclei [36] to benignly exploit vulnerabilities and evaluate the trustworthiness of CVEs mapped from CPE information. We found that CPE information may fail to disclose the existence of CVEs fully.

7.2 Payload-based detection

Finding the existence of CVEs through actual payload exchange is more reliable than using CPE information. However, this type of CVE detection is often highly specialized and could be impractical to use on a large scale [29].

Researchers studied various ways to detect vulnerabilities on the Internet through benign payload-based methods [17] [28] [2] [9]. Durumeric et al. [17] sent out crafted packets without payload or padding to discover the population of Heartbleed vulnerability in the IPv4 address space. Koot [28] studied the prevalence of *CVE-2019-11510* in the Netherlands. He tested the vulnerability by exploiting a crafted path. Antrobus et al. [2] developed a vulnerability scanner to identify multiple vulnerabilities ranging from weak cipher checks to Denial-of-Service tests. They conducted actual vulnerability checks, such as parsing URLs or testing the usage of unpredictable tokens. Gao et al. [9] studied vulnerability detection through Internet-wide scans for three CVEs. They used ZMap to detect open ports and sent special packets containing certain strings or objects to fingerprint the vulnerabilities. Similarly, Yu et al. [55] discussed Internet-wide vulnerability detection by sending special packets.

To accelerate the vulnerability detection process, Schagen et al. [45] examined a concept that used a fuzzer to find the discriminating seeds that could differentiate patched and unpatched services. Users could apply these seeds to detect Internet-wide vulnerable services. However, as Yu et al. [55] pointed out in their work, this kind of vulnerability scanning may have ethical concerns and CVEs that can be tracked simply through external benign scans may be limited.

Our work focuses on CVEs that can be detected easily through benign external scans. Due to the massive amount of IPs we target, obtaining approval from each host owner is infeasible. We address this concern by executing benign payload-based CVE scans and obtaining ethical approval for the project from our university before performing our scanning campaigns. Our scanning results revealed that detecting vulnerabilities through payload exchange may require a specialized understanding of target vulnerabilities, which may lead to different designs of scanning scripts returning inconsistent CVE findings.

8 Conclusion

Enterprises, government agencies, and academics increasingly rely on vulnerability tagging services to assess the risk level of Internet-facing computing infrastructures. However, such reports are typically used as ground truth without scrutinizing their accuracy. In this paper, we perform independent experiments to assess the trustworthiness of such vulnerability tagging services. We compare the reports of a market leader in vulnerability tagging, the Shodan Search Engine, with the reports of our synchronous experiments that use carefully crafted Nuclei templates tailored to target requests based on specific vulnerability checks and payloads for a given vulnerability. Our analysis shows that for "unverified" CVEs by Shodan where banner information is utilized, users of Shodan vulnerability tagging service face massive overreporting (on average 70%), which for some CVEs can be up to 95%. It is also noticeable that 52.07% of the vulnerable endpoints identified by our experiments were not reported by Shodan. Also, according to our study, for CVEs that Shodan targets as "verified", Shodan users are massively under-reporting vulnerability hosts. Our work shows that the above-mentioned shortcomings do exist when comparing our results with a different vulnerability tagging service, ONYPHE, which uses a similar tagging methodology. This suggests that different vulnerability tagging services may yield differing results.

Our findings have significant implications for industry users, policymakers, and security researchers, as they challenge the trustworthiness of vulnerability reports used for operational and regulatory decisions. With this study, we would like to make the different stakeholders aware of the limitations of current vulnerability tagging services and open a debate on concrete steps needed to advance and standardize such services based on best current practices and community efforts to make them more trustworthy in the future.

Ethical Considerations

This research adheres to the Menlo Report's ethical principles of Respect for Persons, Beneficence, Justice, and Respect for Law and Public Interest [26].

Respect for Persons: We respected individual privacy by scanning only publicly accessible systems without collecting personally identifiable information (PII). Our research project involves processing sensitive information about vulnerable systems. We obtained approval from our Institutional Review Board (IRB), which also required a Data Management Plan to ensure the data was stored on a secured server with access limited to the researchers involved.

Beneficence: To maximize benefits and minimize harm, we first examined all of the selected Nuclei templates and ruled out any presence of malicious exploits or otherwise harmful payloads. Scans were conducted with minimal impact; on average, each endpoint is probed once every 90 minutes. While scans might trigger alerts that network operators have to deal with, we assess this impact as modest, since operators experience thousands of daily scans. While it is infeasible for Internet-wide scans to obtain prior consent, we ran a web page on port 80 of our scanning source address, explaining our scan purpose and providing contact information for opting out of future scans. We did not receive any opt-outs or complaints. The benefit of this research is to support network operators, CSIRTs and others who rely on commercial scanning services with information on their accuracy and value for network defense.

Justice: We ensured a fair distribution of risks and benefits by focusing on publicly accessible enterprise systems and disseminating findings to benefit the broader community, including researchers and system owners.

Respect for Law and Public Interest: Our research complied with relevant laws, most notably the General Data Protection Regulation. Some jurisprudence has considered IP addresses a form of PII since it might be possible to tie it to natural persons, but we scanned enterprise software on servers, which is not directly tied to individuals, except for some edge cases. The GDPR does provide legal grounds for collecting PII for statistical purposes, provided that the processing adheres to the principles of lawfulness, fairness, transparency, and data protection safeguards as outlined in the regulation.

Open Science Policy Compliance

In this work, we adhere to the principles of Open Science.

Reproducibility. In Section 3, we describe in details the methodology that we used in this study. Within this work, we updated several Nuclei templates to detect CVEs. We share these updated templates as artifacts for evaluation.[2] To our opinion, the templates and the detailed description of the methodology should provide sufficient details to replicate our study. However, we admit that obtaining exactly the same numbers is infeasible due to several factors, e.g., different scanning period, different scanning vantage points, network instability, etc. Still, the final percentages should not change significantly. In any case, we welcome the reproducibility studies because they would also provide us with valuable data to perform further investigation.

Within this work, we use two public sources of information about vulnerable hosts: *Shodan* and *ONYPHE*. So as we provide the exact list of CVEs and the dates for which we extracted the data, the interested reader can extract the same dataset from both these platforms. Although for CVEs with a large number of vulnerable endpoints, we use only a subset of them in our study, we do not expect the final numbers to change significantly. Unfortunately, we cannot share the exact list of endpoints as it might put them in danger of being compromised.

[2]The updated Nuclei templates are available at https://osf.io/2n9zs/?view_only=40843bfa397f4c93817a5705ba6c5782

References

[1] America's Cyber Defense Agency. Cyber Hygiene Services. https://www.cisa.gov/cyber-hygiene-services, 2024.

[2] Rob Antrobus, Benjamin Green, Sylvain Frey, and Awais Rashid. The forgotten i in iiot: A vulnerability scanner for industrial internet of things. 2019.

[3] Giovanni Barbieri, Mauro Conti, Nils Ole Tippenhauer, and Federico Turrin. Assessing the use of insecure ics protocols via ixp network traffic analysis. In *2021 International Conference on Computer Communications and Networks (ICCCN)*, pages 1–9, 2021.

[4] Christopher Bennett, A Abdou, and Paul C van Oorschot. Empirical scanning analysis of censys and shodan. In *Workshop on Measurements, Attacks, and Defenses for the Web*, 2021.

[5] BinaryEdge. BinaryEdge: We gather data for you. https://www.binaryedge.io/, 2024.

[6] Bitbucket. Bitbucket Server and Data Center Advisory 2022-08-24. https://confluence.atlassian.com/bitbucketserver/bitbucket-server-and-data-center-advisory-2022-08-24-1155489835.html, 2022.

[7] Xander Bouwman, Harm Griffioen, Jelle Egbers, Christian Doerr, Bram Klievink, and Michel Van Eeten. A different cup of TI: the added value of commercial threat intelligence. In *29th USENIX security symposium (USENIX security 20)*, pages 433–450, 2020.

[8] Censys. The Leading Internet Intelligence Platform for Threat Hunting and Attack Surface Management. https://censys.com/, 2024.

[9] GAO Chuan, Han-bing YAN, and JIA Zi-Xiao. Method for measuring the internet devices and applications based on the features. In *International Conference on Computer Networks and Communication Technology (CNCT 2016)*, pages 36–46. Atlantis Press, 2016.

[10] Primoz Cigoj and Borka Jerman Blazic. An intelligent and automated wcms vulnerability-discovery tool: the current state of the web. *IEEE Access*, 7:175466–175473, 2019.

[11] CISA. CY2021 ADMINISTRATIVE SUBPOENA FOR VULNERABILITY NOTIFICATION YEAR IN REVIEW. https://www.cisa.gov/sites/default/files/2023-01/CY2021_Admin_Subpoena_Summary_Factsheet_FINAL.pdf, 2024.

[12] CISA. Known Exploited Vulnerabilities Catalog. https://www.cisa.gov/known-exploited-vulnerabilities-catalog, 2024.

[13] Cisco. Introduction to Cisco IOS NetFlow - A Technical Overview. https://www.cisco.com/c/en/us/products/collateral/ios-nx-os-software/ios-netflow/prod_white_paper0900aecd80406232.html, 2012.

[14] Michelle Cotton, Lars Eggert, Dr. Joseph D. Touch, Magnus Westerlund, and Stuart Cheshire. Internet Assigned Numbers Authority (IANA) Procedures for the Management of the Service Name and Transport Protocol Port Number Registry. RFC 6335, August 2011.

[15] Docker. Docker Builds: Now Lightning Fast. https://www.docker.com, 2024.

[16] Z. Durumeric. ZGrab2. https://github.com/zmap/zgrab2, 2018.

[17] 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 *Proceedings of the 2014 conference on internet measurement conference*, pages 475–488, 2014.

[18] Zakir Durumeric, Eric Wustrow, and J Alex Halderman. ZMap Fast internet-wide scanning and its security applications. In *22nd USENIX Security Symposium (USENIX Security 13)*, pages 605–620, 2013.

[19] Aksel Ethembabaoglu, Rolf van Wegberg, Yury Zhauniarovich, and Michel van Eeten. The unpatchables: Why municipalities persist in running vulnerable hosts. In *33rd USENIX Security Symposium*, USENIX Security '24, Philadelphia, PA, August 2024. USENIX Association.

[20] European Commission. NIS 2 Directive. https://eur-lex.europa.eu/eli/dir/2022/2555/oj, 2022.

[21] FOFA. FOFA. https://en.fofa.info/, 2024.

[22] Béla Genge and Călin Enăchescu. Shovat: Shodan-based vulnerability assessment tool for internet-facing services. *Security and communication networks*, 9(15):2696–2714, 2016.

[23] Béla Genge, Piroska Haller, and Călin Enăchescu. Beyond internet scanning: Banner processing for passive software vulnerability assessment. *International Journal of Information Security Science*, 4(3), 2015.

[24] Robert Graham. Masscan: Mass IP port scanner.

[25] Liz Izhikevich, Renata Teixeira, and Zakir Durumeric. LZR: Identifying unexpected internet services. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 3111–3128, August 2021.

[26] Erin Kenneally and David Dittrich. The Menlo Report: Ethical Principles Guiding Information and Communication Technology Research. *Available at SSRN 2445102*, 2012.

[27] Taeeun Kim and Hwankuk Kim. A design of automated vulnerability information management system for secure use of internet-connected devices based on internet-wide scanning methods. *IEICE TRANSACTIONS on Information and Systems*, 104(11):1805–1813, 2021.

[28] Matthijs Koot. Field note on cve-2019-11510: Pulse connect secure ssl-vpn in the netherlands. *Digital Threats: Research and Practice*, 1(2):1–7, 2020.

[29] Martin Laštovička, Martin Husák, and Lukáš Sadlek. Network monitoring and enumerating vulnerabilities in large heterogeneous networks. In *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–6, 2020.

[30] LeakIX. LeakIX. https://leakix.net/, 2024.

[31] Vector Guo Li, Matthew Dunn, Paul Pearce, Damon McCoy, Geoffrey M Voelker, and Stefan Savage. Reading the tea leaves: A comparative analysis of threat intelligence. In *28th USENIX security symposium (USENIX Security 19)*, pages 851–867, 2019.

[32] Naif Mehanna, Walter Rudametkin, Pierre Laperdrix, and Antoine Vastel. Free Proxies Unmasked: A Vulnerability and Longitudinal Analysis of Free Proxy Services. In *MADWeb 2024 - Workshop on Measurements, Attacks, and Defenses for the Web*, pages 1–12, February 2024.

[33] Vahid Moghiss and Alireza Shameli-Sendi. I-recon: An iot based search engine for internet-facing services vulnerability reconnaissance. *IEEE Access*, 2024.

[34] Nmap. Nmap: the Network Mapper - Free Security Scanner. https://nmap.org/, 2021.

[35] Nmap. UDP Scan (-sU). https://nmap.org/book/scan-methods-udp-scan.html, 2024.

[36] Nuclei. Nuclei: Fast and customisable vulnerability scanner based on simple YAML based DSL. https://github.com/projectdiscovery/nuclei, 2024.

[37] Nuclei. Nuclei Templates. https://github.com/projectdiscovery/nuclei-templates, 2024.

[38] NVD. NATIONAL VULNERABILITY DATABASE: Official Common Platform Enumeration (CPE) Dictionary. https://nvd.nist.gov/products/cpe, 2024.

[39] NVD. NATIONAL VULNERABILITY DATABASE: Vulnerabilities. https://nvd.nist.gov/vuln, 2024.

[40] Jamie O'Hare, Rich Macfarlane, and Owen Lo. Identifying vulnerabilities using internet-wide scanning data. In *2019 IEEE 12th International Conference on Global Security, Safety and Sustainability (ICGS3)*, pages 1–10. IEEE, 2019.

[41] Onyphe. Onyphe: Cyber Defense Search Engine. https://www.onyphe.io/, 2024.

[42] Ponemon Institute. The Value of Threat Intelligence: Annual Study of North American & United Kingdom Companies. https://stratejm.com/wp-content/uploads/2019/08/2019_Ponemon_Institute-Value_of_Threat_Intelligence_Research_Report_from_Anomali.pdf, 2019.

[43] Luis Alberto Benthin Sanguino and Rafael Uetz. Software vulnerability analysis using cpe and cve. *arXiv preprint arXiv:1705.05347*, 2017.

[44] Takayuki Sasaki, Akira Fujita, Carlos H. Gañán, Michel van Eeten, Katsunari Yoshioka, and Tsutomu Matsumoto. Exposed infrastructures: Discovery, attacks and remediation of insecure ics remote management devices. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 2379–2396, 2022.

[45] Nathan Schagen, Koen Koning, Herbert Bos, and Cristiano Giuffrida. Towards automated vulnerability scanning of network servers. In *Proceedings of the 11th European Workshop on Systems Security*, pages 1–6, 2018.

[46] Shodan. Shodan Monitor. https://monitor.shodan.io/dashboard, 2024.

[47] Shodan. Shodan: Search Engine for the Internet of Everything. https://www.shodan.io/, 2024.

[48] Shodan. Understanding Shodan Vulnerability Assessment. https://help.shodan.io/mastery/vulnerability-assessment, 2024.

[49] Carlotta Tagliaro, Martina Komsic, Andrea Continella, Kevin Borgolte, and Martina Lindorfer. Large-scale security analysis of real-world backend deployments speaking iot-focused protocols. *arXiv preprint arXiv:2405.09662*, 2024.

[50] Richard J Thomas, Joseph Gardiner, Tom Chothia, Emmanouil Samanis, Joshua Perrett, and Awais Rashid. Catch me if you can: An in-depth study of cve discovery time and inconsistencies for managing risks in critical infrastructures. In *Proceedings of the 2020 Joint Workshop on CPS&IoT Security and Privacy*, pages 49–60, 2020.

[51] Andrea Tundis, Eric Marc Modo Nga, and Max Mühlhäuser. An exploratory analysis on the impact of shodan scanning tool on the network attacks. In *Proceedings of the 16th International Conference on Availability, Reliability and Security*, pages 1–10, 2021.

[52] Roman Ushakov, Elena Doynikova, Evgenia Novikova, and Igor Kotenko. Cpe and cve based technique for software security risk assessment. In *2021 11th IEEE International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications (IDAACS)*, volume 1, pages 353–356. IEEE, 2021.

[53] Vulhub. Vulhub. https://github.com/vulhub/vulhub, 2024.

[54] Jonathan Codi West and Tyler Moore. Longitudinal study of internet-facing openssh update patterns. In *International Conference on Passive and Active Network Measurement*, pages 675–689. Springer, 2022.

[55] Miao Yu, Jianwei Zhuge, Ming Cao, Zhiwei Shi, and Lin Jiang. A survey of security vulnerability analysis, discovery, detection, and mitigation on iot devices. *Future Internet*, 12(2):27, 2020.

[56] Zoomeye. Zoomeye. https://www.zoomeye.hk/, 2024.

A CVE Information

In this section, we list down all of the selected 37 CVEs in this work in Table 2, including 17 Shodan-verified CVEs and 20 Shodan-unverified CVEs. We show the Shodan verification status, CVSS score, and application type for each CVE, as well as its Nuclei template validation status.

Table 2: **Selected CVEs Details.** *CVSS3* – CVSS 3 score (asterisks mean that CVSS2 score is used since the CVSS3 is unavailable); *Application* – vulnerable application/service; *Shodan Detection* – if the corresponding CVE is verified on Shodan (✔- verified, ✘- unverified); *Nuclei Detection* – if the corresponding Nuclei template is validated (✔- validated, ✘- not validated); *ONYPHE Detection* – CVE detection methods adopted by ONYPHE (★ - payload-based, ☆ - banner-based).

CVE-ID	CVSS3	Application	Shodan Detection	Nuclei Detection	ONYPHE Detection
CVE-2015-1635	10*	Microsoft Windows	✔	✘	-
CVE-2017-7269	9.8	Microsoft Internet Information Services (IIS)	✔	✘	-
CVE-2019-11510	10	Pulse Secure Pulse Connect Secure (PCS)	✔	✘	★
CVE-2019-1653	7.5	Cisco Small Business RV320 and RV325 Routers	✔	✘	-
CVE-2019-19781	9.8	Citrix Application Delivery Controller (ADC) and Gateway	✔	✘	★
CVE-2020-5902	9.8	F5 BIG-IP	✔	✘	★
CVE-2021-21972	9.8	VMware vCenter Server and VMware Cloud Foundation	✔	✘	★
CVE-2021-26855	9.8	Microsoft Exchange Server	✔	✘	★ ☆
CVE-2021-34473	9.8	Microsoft Exchange Server	✔	✘	★ ☆
CVE-2021-41277	7.5	Metabase	✔	✔	-
CVE-2021-43798	7.5	Grafana (Open Source)	✔	✔	-
CVE-2022-36804	8.8	Atlassian Bitbucket Server and Data Center	✔	✔	-
CVE-2023-23333	9.8	SolarView Compact	✔	✘	-
CVE-2023-27350	9.8	PaperCut	✔	✘	☆
CVE-2023-35078	9.8	Ivanti Endpoint Manager Mobile (EPMM)	✔	✘	☆
CVE-2023-35082	9.8	Ivanti Endpoint Manager Mobile (EPMM)	✔	✘	☆
CVE-2023-39143	9.8	PaperCut NG and PaperCut MF	✔	✘	☆
CVE-2012-1823	7.5*	PHP	✘	✔	-
CVE-2014-3704	7.5*	Drupal core	✘	✔	-
CVE-2016-3088	9.8	Apache ActiveMQ	✘	✔	-
CVE-2017-12635	9.8	Apache CouchDB	✘	✔	-
CVE-2017-15715	8.1	Apache httpd	✘	✔	-
CVE-2018-1000533	9.8	klaussilveira GitList	✘	✔	-
CVE-2018-1000861	9.8	Jenkins	✘	✔	-
CVE-2018-12613	8.8	phpMyAdmin	✘	✔	-
CVE-2018-18778	6.5	ACME mini_httpd	✘	✔	-
CVE-2018-7600	9.8	Drupal	✘	✔	-
CVE-2019-3396	9.8	Atlassian Confluence Server	✘	✔	-
CVE-2020-7247	9.8	OpenSMTPD	✘	✔	-
CVE-2020-9402	8.8	Django	✘	✔	-
CVE-2021-21311	7.2	Adminer	✘	✔	-
CVE-2021-28169	5.3	Eclipse Jetty	✘	✔	-
CVE-2021-34429	5.3	Eclipse Jetty	✘	✔	-
CVE-2021-40438	9	Apache HTTP Server	✘	✔	-
CVE-2021-41773	7.5	Apache HTTP Server	✘	✔	-
CVE-2021-42013	9.8	Apache	✘	✔	-
CVE-2023-46604	9.8	Java OpenWire protocol marshaller	✘	✔	☆