Identifying Vulnerabilities Using Internet-wide Scanning Data

Jamie O'Hare, Rich Macfarlane, Owen Lo
School of Computing
Edinburgh Napier University
Edinburgh, United Kingdom
{40168785, r.macfarlane, o.lo}@napier.ac.uk

***Abstract*—Internet-wide scanning projects such as Shodan and Censys, scan the Internet and collect active reconnaissance results for online devices. Access to this information is provided through associated websites. The Internet-wide scanning data can be used to identify devices and services which are exposed on the Internet. It is possible to identify services as being susceptible to known-vulnerabilities by analysing the data. Analysing this information is classed as passive reconnaissance, as the target devices are not being directly communicated with. This paper goes on to define this as contactless active reconnaissance. The vulnerability identification functionality in these Internet-wide scanning tools is currently limited to a small number of high profile vulnerabilities. This work looks towards extending these features through the creation of a tool Scout which combines data from Censys and the National Vulnerability Database to passively identify potential vulnerabilities. This is possible by analysing Common Platform Enumerations and associated Common Vulnerability and Exposures. Through this novel approach, active vulnerability scanning results can be gained, while mitigating the associated issues of active scanning, such as possible disruption to the target network and devices. In initial experiments performed on 2571 services across 7 local academic intuitions, 12967 potential known-vulnerabilities were identified. More focused experiments to evaluate the results and compare accuracy with industry standard vulnerability assessment tools were carried out and Scout was found to successfully identify vulnerabilities with an effectiveness score of up to 74 percent when compared to OpenVAS.**

***Index Terms*—Scout, Censys, Vulnerability Assessment, Internet-wide, Scanning, Passive, Contactless, National Vulnerability Database**

I. Introduction

Gartner believes that by 2020, 99% of vulnerabilities exploited will be known prior to their use [1]. The risk associated with known vulnerabilities cannot be overstated, with high profile incidents such as WannaCry and NotPetya both exploiting the Eternal Blue vulnerability, which was both well known and which had patches developed to mitigate prior to it being used in these attacks [2].

One way in which known vulnerabilities can be actively identified is through the use of vulnerability assessment tools such as OpenVAS and Nessus [3], [4], typically as part of the active reconnaissance phase of a pentest enagagement. In an attempt to identify target systems and their exploitable vulnerabilities, these tools aggressively scan networks and interrogate operating network services. This can be potentially disruptive to the target network causing issues such as denial of service, through the considerable time and resources required to perform the scans. Due to this potential issue, as well as specific legal requirements, vulnerability assessment tools typically require permission from the target organization before being used.

The known vulnerabilities identified by these tools are associated with a specific Common Vulnerabilities and Exposure (CVE), which highlights a vulnerability for a specific service. A CVE entry contains information associated with the vulnerability including a Common Platform Enumeration (CPE) and a Common Vulnerability Scoring System (CVSS). The CPE ties a CVE to a specific product and version, while the CVSS provides an impact score. These data feeds are all stored as parts of the National Vulnerability Database (NVD), a public repository managed by the National Institute of Standards and Technology. The NVD includes several separate data feeds including checklist references, software flaws, misconfigurations, product names and impact metrics. Since its inception in 1997, the NVD has published information about more than 100,000 vulnerabilities and is the standard reference for vulnerabilities.

Another way to identify known vulnerabilities is through the use of Internet-wide scanning projects such as Shodan and Censys [5], [6]. These tools collate lightweight active reconnaissance results from services operating on publicly available IP addresses, made available to users through their respective websites and APIs. This data is collected by crawlers which scan the entire IPv4 address range across many ports gathering data on network services in a similar way to the vulnerability assessment tools used in active reconnaissance [7], [8]. These crawlers operate by sending only one or a small number of packets to each individual service. Service specific packets are sent to check for vulnerable service configurations, however currently, this vulnerability functionality provided by Shodan and Censys is rather limited [9]. Only a very small number of vulnerabilities are checked for and each has bespoke identification functionality hard coded into the specific scanners [7], [10].

A way in which this functionality could be expanded is through cross-referencing the more general service specific data provided by these Internet-wide scanning projects, against vulnerability repositories such as the NVD. This piece of work looks at exploring and evaluating this type of vulnerability

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

identification by building on the functionality of the Censys Internet-wide scanning project.

This paper is organized as follows. The methodology includes the design decisions and their impact on implementation. Initial Validation presents an applicable use case. Experiments and Results describe the topology and design of the experiments conducted along with the corresponding results. Analysis and Evaluation examine and commentates on the results of the conducted experiment results. Finally, the Conclusion closes this paper, providing the outcomes.

II. Methodology

This section presents the design and subsequent implementation process towards extending the vulnerability analysis functionality within an Internet-wide scanning project.

This work resulted in a tool known as Scout, named due to the connotations of reconnaissance, exploration, and discovery. Previous work has defined Contactless Reconnaissance when using the output of Internet search engines such as Google [11] to limit the contact with target systems, and this work extends this with the use of the Censys Internet-wide scanning cached information and associated NVD vulnerability data. Thus the term Contactless Active Reconnaissance has been proposed for this type of reconnaissance using Internet-wide subject-specific search engines which index and make available active reconnaissance results [9], [11]. A graphical representation of the comparison between the approach outlined above to a more standard vulnerability assessment approach can be seen in Fig. 1. Fig. 1 illustrates a difference between these approaches can be highlighted. This comparison was originally performed by Simon, Moucha and Keller however, latency was not considered [11]. This latency can vary across services across Internet-wide scanning projects. The standard active approach interacts with the target receiving immediate feedback through their correspondence. While contactless active approach does not directly interact with the target and therefore, inherits a latency which varies.

This comparison will serve as the basis for the evaluation of Scout.

A. Design

Scout has 3 distinct operations, gathering service scanning data, associating possible vulnerabilities, and presentation of results. The first stage of gathering Internet-wide scanning data can differ considerably depending on the service chosen. There are three main Internet-wide scanning projects, Shodan, Censys and ZoomEye [5], [6], [12]. The oldest, most popular and well-known internet scanning project is Shodan. Founded in 2009 by John Matherly, it is infamous for revealing the sheer number of sensitive information about devices connected to the internet. More recently Censys has emerged, a product of a research group from the University of Michigan. The same group is also responsible for ZMap, which will be discussed later. Unlike the other projects, the methodology for Censys can be found in the associated academic literature. Finally, there is ZoomEye, developed by Knownsec Inc from

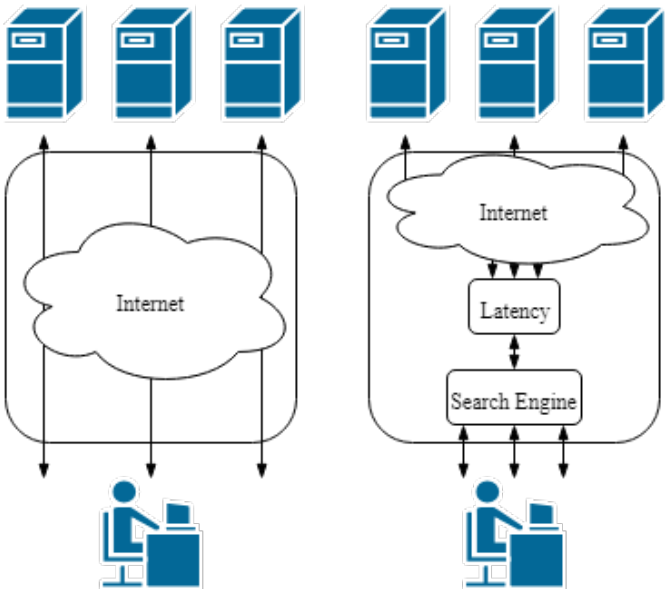

Fig. 1. Approach Comparison

Beijing. Despite being released in July 2013, ZoomEye does not seem to have been discussed much literature. From the little research available, most of it appears to be not transcribed into English yet. There is also seems to be a lack of detailed documentation. Due to these factors, ZoomEye was not be considered suitable. When deciding which scanning project to utilize, the determining factors were the value and format of the data provided. There are several factors which impact the value of data coming from an Internet-scanning project, these are age, bias, and comprehensiveness.

1) Age: Shodan's scans perpetually across a massive number of services in the IPv4 address range, indexing 550 million services a month [8], [13]. Shodan scans are performed in a blend of horizontal and vertical scanning as numerous services are scanned for, across the Internet in a random order [14]. Matherly comments that the scan results are continually used to update the underlying database, however, research suggests otherwise. In research conducted by Bodenhiem, Butts, Dunlap, and Mullins, they noted a varying delay in scan results appearing on the web interface of Shodan, a delay varying from 1 day to 19 days [15]. This finding has impacted the methodology employed in related research [16]. Censys' scans horizontally with scans scheduled daily, biweekly and weekly depending on the service [14]. These scans are performed over a 24-hour period, even though the underlying ZMap technology can perform this faster [7]. This regimental scanning behavior and its corresponding updates to the underlying database at fixed times is in contrast with the Shodan behaviour.

2) Bias: Bias in Internet-wide scanning stems from the scanning method, specifically the geographic source(s) of the scan. Shodan distributes crawlers world-wide to minimize the effect of a geographic bias [8], [14]. An example of potential geographic bias is the banning of Chinese IP

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

ranges by network administrators in the United States [13]. Upon initial deployment, Censys'scans were conducted from a single source at the University of Michigan [7]. Despite not documented in the more recent literature, this can be confirmed through the use of GreyNoise, an Internet-wide scanning aggregator [17].

3) Comprehensiveness: Comprehensiveness is the completeness of the scans. The scanning method used looks to have an impact on the completeness of the scan data. For scanning, Censys uses the ZMap scanner, another project from the same research group [7]. ZMap was released in 2013, later upgraded in 2014 to increase scanning speed using specialist hardware. A comprehensive comparison was carried out against Masscan, which at the time was seen to be the recommended port scanner [7], [18]. Masscan offeres incremental or random scanning. During incremental scanning, the scanner is more likely to be blocked automatically by firewalls due to the predictable behaviour and common scan source [19]. Random scanning is recommended to avoide these issues. Masscan uses their own encryption algorithm to randomize target IP addresses. The encryption algorithm employed in Masscan is a custom algorithm which uses the first 3 rounds of DES, but when visualized, this seems to reveal artifacts which suggest the randomization is not entirely random. This issue was identified by the author, however it was deemed suitable for the randomizing required [20]. However, in comparison to ZMap, Masscan was considered to be less random, and led to poorer performance due to the potential for overloading target networks [18]. Matherly comments that Shodan's crawlers are randomized, using a similar method to ZMap, although it seems the implementation is not well documented [13].

Assessing these qualities: age, bias, and comprehensiveness, Censys looked to possibly have fewer bias data than Shodan, however it is a more comprehensive, up-to-date and predictable data source. Therefore, Censys was chosen to be taken forward as the Internet-wide scanning project for the work.

With Censys selected, an approach to associate vulnerabilities to the Censys scanning data was then considered. A key factor in this process is the way Censys provides their information. Using the Python API a user must specify the desired fields along with an acceptable query. Output fields are returned and provide specific attributes of the active reconnaissance results, for example a web service would include the HTTP status code and HTTP body. Metadata fields are those which are needed to identify possible vulnerabilities associated with a service, as they contain the required information about the service. There can be a greater number of useful fields than the 20 field limit in a Censys query depending on the services being scanned [6]. The approach which was chosen to maximize testing was to simply use one query which looked for the applicable metadata field for a specific service. The service chosen was HTTP operating on port 80 due to the high density of these services online as well as service information found within the NVD. '80.http.get.metadata.description'is the field which is specifically queried for in this. Future work could look to expand to a wider range of public Internet services.

The link between service information and any possible vulnerabilities, is the Common Platform Enumeration (CPE). A CPE is a unique identifier comprised of several fields for a specific version for a particular hardware, service or operating system [31]. They are used to partition issues identified in other data feeds within the NVD to an exact application and version. The following is an example of a CPE:

cpe:/a:apache:http_server:2.4.7

where /a denotes an application, apache denotes the vendor, http_server denotes the product, 2.4.7 denotes the version. Typically the metadata fields do not supply information applicable beyond the version field. The Internet-wide scanning data which corresponds to this CPE will commonly look similar to the below:

Apache httpd 2.4.7

Several related work have put forward methods to deduce a CPE from analysis of related data.

Na, Kim, and Kim propose a hierarchical tree approach when conducting CPE creation through service banner analysis [21], [22]. This method includes the generation and utilization of a CPE dictionary tree. After conducting keyword analysis on a service-banner, the keyword(s) extracted are used in conjunction with the CPE tree. This process is repeated at each step of the tree, the longest (greater number of child nodes) CPE is created. The outlined approach has a significant weakness, the inability to operate on incomplete data. The reliance on sequential keywords starting with the vendor creates a dependency on identifying a vendor-specific keyword despite having enough keywords from other sections to identify the CPE.

Gawron, Cheng, Meinel explored CPE creation in their research surrounding patch monitoring [23]. The authors propose a new passive vulnerability detection method by analyzing system logs and creating CPEs to tie to CVEs. Log files are parsed to identify program and version information. This approach omits vendor information because their analysis of system logs concluded that system logs do not include vendor details. This omission hinders the possibility of manufacturing a CPE without additional computation to associate the correct vendor. This omission also could create a problem when services share the same name but not the same vendor causing incorrect results.

The approach employed by the ShoVAT tool utilizes hash tables to store elements of CPEs corresponding to version numbers [24]. Instead of searching for specific substrings through the large CPE database, ShoVAT identifies version numbers matching the regular expression pattern, Vpat = .(I.)*I'. Once the pattern is identified, the hash table is used to associate the other elements of the CPE such as Vendor and Product. If this association is not decisive then the

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

surrounding information in the banner is used to identify the most appropriate banner.

A significant weakness with this and aforementioned approaches, is the reliance and explicit dependency on the CPE dictionary. The CPE dictionary has some issues, such as it contains over 2,500 depreciated entries, all of which are due to name correction [25]. This problem may have a knock on affect with all of the aforementioned approaches, as it seems none consider this issue. There seems to be only one approach in the current literature which considers these typographical errors, the work by Sanquino & Uetz, in which they explore CPE creation in relation to Vulnerability Management Systems. The authors employ the use of the Levenshtein distance algorithm to address the erroneous NVD. The Levenshtein distance algorithm returns the number of dissimilarities between two strings, and for their use, the authors set a threshold of 2. This threshold of 2 was determined by analysing the deprecated CPEs in the CPE dictionary, commonly depreciation took place due to a typo in the product with a Levenshtein distance of 1 or 2. This step has the ability to produce more than one CPE, therefore the CPEs are then ordered by version as required. Although this method reduces the errors, it can result in the creation of more than one possible CPEs.

The approach for Scout was to incorporate a hybrid of previous suggestions. Using symmetric comparisons in conjunction with a tiered list and some usage of the Levenshtein distance algorithm. This is discussed in further detail in the next section prior to presenting results.

B. Implementation

```
for CPE in dictionary:
    intersection = candidate ⋂ CPE
    if length(intersection) = 0 then
        elected candidate append CPE
    else if length(intersection) < 2 then
        vetted candidate append CPE
    else if length(intersection) = 2 then
        if levenshtein(intersection)
        < length(intersection) then
            accepted candidate append CPE
```

Listing 1. CPE matching pseudocode

Python was chosen for the Scout tool, mainly due to the ease of integration with the Censys API, and other libraries including a Levenshtein distance algorithm, editdistance. The Censys'API requires a valid set of API keys, therefore, the tool's user must also register for Censys use. To obtain information from the NVD, an existing project known as cve-search was used [26]. cve-search aggregates several data feeds including those of the NVD within a MongoDB instance, which can be easily accessed and used through the Python library pymongo. The most significant piece of implementation was the Internet-wide scanning data to vulnerability information. As mentioned prior, the approach taken to in service banner analysis tried to address issues found in previous research. The CPE dictionary was iterated over and for each iteration, the given metadata returned by Censys was symmetrical compared

Fig. 2. An example of Scout output with IP addresses censored

to the CPE. If the result of this symmetric difference was 0 then the CPE was appended to the elected candidate list. If the difference was less than 2 then CPE was appended to a vetted candidate. Finally, if the difference was equal to 2 then further analysis was done to discover if there were similar strings between the metadata and CPE through the use of the Levenshtein distance algorithm. Pseudocode for this process can be found in Listing 1.

Once the CPE dictionary has been enumerated, the 3 lists produced are analysed. Beginning by checking the length of items in the elected, vetted and candidate list in sequential order. If the result returns 0 then the next list is analysed, if the length of returns 1 then that result is treated as the final CPE, however, if the length is greater than 1 then the matching finishes inconclusively. A major advantage of this method is that it is conclusive by providing a suitable CPE where possible instead of producing a list of potential CPEs. This is achieved through the usage of a tiered list along with the Levenshtein distance algorithm. The latter of which allows for consideration of any erroneous NVD entries. An additional benefit of this approach is that it is non-service specific, as there is no hard coded rules for specific individual services.

With the CPE identified, it is then used in the CVE assignment. Included within CVEs, is a field of the vulnerable configurations. This field is a list of CPEs which correspond to services vulnerable to the CVE. By using cve-search, this field can be searched with the chosen CPE. For each CVE that is returned the ID and corresponding CVSSv2 score is stored. The final step is to present the findings. The method chosen was simply to print the results with clear indentation to isolate each result. An output example can be seen in Fig. 2.

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

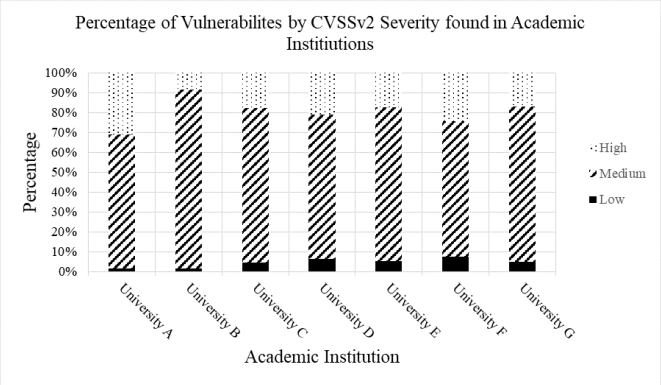

Fig. 3. Vulnerabilities in local institutions

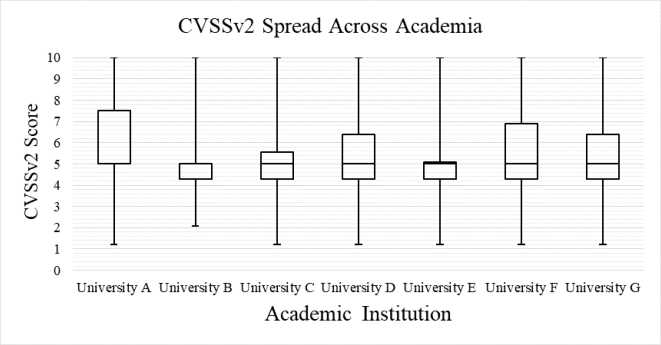

Fig. 4. Average and Outliers Vulnerabilities

III. Initial Validation

To present a potential use case for Scout, 7 local academic institutions were used as a dataset, leading to a total of 12967 services being processed. Results from this analysis can be found in Fig. 3 and Fig 4. Fig. 3 illustrates University A having the largest percentage of high severity vulnerabilities, while also having the smallest percentage of low severity vulnerabilities. Fig. 4 illustrates University B has the highest lowest CVSSv2 score across all institutions. When included in reports and audits, this can be especially useful to portray the significance of the findings to individuals without technical knowledge. This is just one of the potential uses of Scout.

IV. Experiments and Results

To gain an insight into the possibility and the potential usefulness of Scout, an array of experiments were carried out. Included in this section is the experiments conducted along with their results.

A. Experiment 1 Manual CPE Assessment

In previous research, CPE creation accuracy conducted automatically by a tool is compared against performing the process manually [24]. Precision and Recall have been used to allow for comparison in future work. Additionally, the data used is publicly available so the work can be reproduced [27]. For this experiment the scanning data was generated from a Censys query for a Scottish university, resulting in output which includes 52 unique services. This data is then processed by Scout and the results are were analysed manually.

TABLE I
Manual CPE Assessment Results

		Scout's Assessment	
		Match	No Match
Manual Assessment	Match	17	1
	No Match	12	22

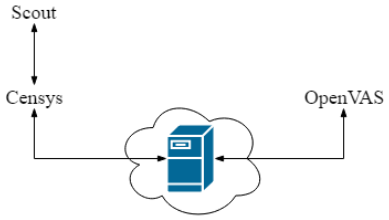

Fig. 5. Experiment Design

True positives in this assessment are defined as entries in which a CPE is assigned correctly by Scout, while true negatives are defined as entries in which a CPE is not assigned as it does not exist. False positives in this assessment are defined as entries in which a common platform enumeration is assigned incorrectly by Scout, while false negatives are defined as entries in which a common platform enumeration exists, but cannot be determined. Results for this experiment can be found in Table I.

B. Experiment 2 Comparative CPE Assessment

In previous research, the CPE creation accuracy is compared with industry tools which perform the same procedure [11], [24]. This involves scanning the same service with both an active and contactless active vulnerability assessment tool. Ideally, this would be the method in which Scout's CPE creation would be evaluated, however, permission could not be gained to use the active scanning tools on a large sample size of live target machines. Therefore, an experiment was conducted on targets created in Amazon Web Services, which were configured to run a small sample size of the same services found by the Scout tool. For the industry tool, OpenVAS was used for comparison. The topology for this experiment can be found in Fig. 5 while the results of the experiment can be found in Table II. In Table II '•' is used to denote a successful match whereas ' ' is used to denote an unsuccessful match.

C. Experiment 3 CVE Assignment

In related work, CVE assignment is compared against industry tools which perform a similar function [11], [24], so again active and contactless vulnerability assessment tools were run against the same services. Metrics used in work to assess web vulnerability scanning, such as Precision, Recall, and F1 measure are used to quantify accuracy, comprehensiveness and effectiveness [28]–[30]. These metrics seemed appropriate to be used in the evaluation due to their test data having a

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

TABLE II
COMPARATIVE CPE ASSESSMENT RESULTS

Advertised Service	Correct CPE	OpenVAS	Scout
Apache 2.4.7	apache:http_server:2.4.7	•	•
Apache 2.4.10	apache:http_server:2.4.27	•	•
Apache 2	apache:http_server:2.4.27		
Apache httpd	apache:http_server:2.4.27		
lighttpd 1.4.18	lighttpd:lighttpd:1.4.18	•	•
1.4.18 lighttpd	lighttpd:lighttpd:1.4.18		•
lighttpd 1.4.19	lighttpd:lighttpd:1.4.18		
lighttpd 1.4.33	lighttpd:lighttpd:1.4.33	•	•
lighttpd	lighttpd:lighttpd:1.4.33		
nginx 1.4.6	nginix:nginx:1.4.6	•	•
nginx	nginix:nginx:1.4.6		

TABLE III
CVE ASSIGNMENT RESULTS

Service	NVD	OpenVAS	Scout
Apache httpd 2.4.7	18	18	15
Apache httpd 2.4.10	16	14	13
Lighttpd 1.4.18	14	6	12
Lighttpd 1.4.33	5	4	3

similar property of a ground truth with a binary state of being susceptible to a vulnerability or not.

To distinguish between potential vulnerabilities and those that could be exploited by an attacker, the National Vulnerability Database was included in the comparison to provide a reference to potential vulnerabilities, while OpenVAS was used to provide a comparison to vulnerabilities the service is susceptible to. A service may have more potential vulnerabilities that are susceptible to due to necessary features being enabled or required configurations. There is also the possibility of the vulnerability have being patched through the process of backporting.

Table 3 presents an overview of outputs from the National Vulnerability Database, OpenVAS, and Scout. The NVD column has been created by manual investigation of the Censys output to NVD CVE information. The OpenVAS and Scout columns shows CVE's output by the tools for the specific services. The specific OpenVAS reports can be found along with the test data [27]. The National Vulnerability Database information used in the process was taken from the website variant on the 19th of March 2018. The differences between the manual NVD data analysis and the Scout output highlights possible issues with the data being returned from the NVD API compared to what is available on the website.

V. ANALYSIS AND EVALUATION

A. Experiment 1 Manual CPE Assessment

Table I suggests that Scout can correctly assign CPEs from Internet-wide scanning data with a success rate of 75%. Of the 13 false results, only 1 was a false positive. This false positive is for the advertised service 'Apache httpd 2.4' where Scout failed to assign the correct CPE - 'cpe:/a:apache:http_server:2.4.0'. Due to lack of consideration for the potential '.0' to be amended on a CPE version field. This led Scout to incorrectly 'cpe:2.3:a:apache:mod_python:2.4'. The 12 false negatives highlight several different problems with the approach employed. 8 of the false negatives come from Apache products stemming from 2 problems. The first problem is due to identifying apache products without a corresponding CPE. This is a particular problem within the NVD, and is not limited to apache products as this problem was also responsible for another 2 false negatives. The second problem stems from when the advertised service does not follow a regular formatting. For the advertised service 'Apache httpd2.2.11', Scout was unable to correctly identify the service due to the lack of a space character separating the version number from the product. The final problem encountered is specific to Microsoft IIS products as the CPE syntax changes from 'cpe:/a:microsoft:iis:7.5' to 'cpe:/a:microsoft:internet_information_services:8.5'. A solution to this problem could be to write a direct rule to change it accordingly, however, doing so would not align with the non-service specific approach aimed at with Scout.

B. Experiment 2 Comparative CPE Assessment

The purpose of this comparative assessment was to analyse how well Scout can manufacture Common Platform Enumerations in comparison to OpenVAS, which performs the same function. Table II presents an overview of the experiment results. Interestingly with the data used, Scout outperforms OpenVAS as Scout is able to identify 1 more CPE than OpenVAS. This is a rather unexpected result due to Scout having the limitation of not being able to interact with the target. Both tools only correctly matched the CPE when enough correct information was supplied. The caveat for OpenVAS which led to Scout outperforming was the need for the advertised service information to be given in the correct order. Due to Table II not displaying information on false positives, a false positive which Scout returned is not highlighted. Scout returned this result due to the advertised service being false, but still having a corresponding CPE. The advertised service was 'lighttpd 1.4.19', Scout returns 'cpe:2.3:a:lighttpd:lighttpd:1.4.19', which although a valid CPE, is not the correct CPE for the 'lighttpd 1.4.18' service. Scout only gathers information from one source and can not corroborate information from other methods, such as an active approach used in OpenVAS could. These results suggest that Scout is able to perform CPE assignments in a more diverse dataset than OpenVAS, although does likely return a false positive in situations where there is false advertisement.

C. Experiment 3 CVE Assignment

Experiment 3 sought to compare the contactless active approach employed in Scout to the active approach found in industry tools such as OpenVAS. An overview of this experiment can be found in Table III, which shows the NVD contains more vulnerabilities than is reported by OpenVAS and Scout. This is expected due to vulnerabilities being tied to specific configurations. However, this is unexpected in regards to Scout as Scout is supposed to return the same NVD data. This suggests there is a discrepancy between the data used

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

TABLE IV
CVE ASSIGNEMENT FOR APACHE 2.4.7 SERVICE

NVD	OpenVAS	Scout
	CVE-2004-0230	
CVE-2013-6438		CVE-2013-6438
CVE-2014-0098		CVE-2014-0098
CVE-2014-0117	CVE-2014-0117	CVE-2014-0117
CVE-2014-0118	CVE-2014-0118	CVE-2014-0118
CVE-2014-0226	CVE-2014-0226	CVE-2014-0226
CVE-2014-0231	CVE-2014-0231	CVE-2014-0231
CVE-2014-3523	CVE-2014-3523	CVE-2014-3523
CVE-2014-8109	CVE-2014-8109	CVE-2014-8109
CVE-2015-0228	CVE-2015-0228	
CVE-2015-3183	CVE-2015-3183	
CVE-2015-3184		CVE-2015-3184
CVE-2015-3185	CVE-2015-3185	CVE-2015-3185
CVE-2016-0736		CVE-2016-0736
CVE-2016-2161	CVE-2016-2161	CVE-2016-2161
CVE-2016-5387	CVE-2016-5387	
CVE-2016-8743	CVE-2016-8743	CVE-2016-8743
	CVE-2017-3167	
	CVE-2017-3169	
	CVE-2017-7679	
CVE-2017-9788	CVE-2017-9788	CVE-2017-9788
CVE-2017-9798	CVE-2017-9798	CVE-2017-9798
Differences	**+4/-4**	**+0/-3**
		+4/-7

TABLE V
CVE ASSIGNEMENT FOR APACHE 2.4.10 SERVICE

NVD	OpenVAS	Scout
	CVE-2004-0230	
CVE-2014-3583	CVE-2014-3583	CVE-2014-3583
CVE-2014-8109	CVE-2014-8109	CVE-2014-8109
CVE-2015-0228	CVE-2015-0228	
CVE-2015-3183	CVE-2015-3183	
CVE-2015-3184		CVE-2015-3184
CVE-2015-3185	CVE-2015-3185	CVE-2015-3185
CVE-2016-0736		CVE-2016-0736
CVE-2016-2161	CVE-2016-2161	CVE-2016-2161
CVE-2016-5387	CVE-2016-5387	
CVE-2016-8743	CVE-2016-8743	CVE-2016-8743
CVE-2017-3167	CVE-2017-3167	CVE-2017-3167
CVE-2017-3169	CVE-2017-3169	CVE-2017-3169
CVE-2017-7668		CVE-2017-7668
CVE-2017-7679	CVE-2017-7679	CVE-2017-7679
CVE-2017-9788	CVE-2017-9788	CVE-2017-9788
CVE-2017-9798	CVE-2017-9798	CVE-2017-9798
Differences	**+1/-3**	**+0/-3**
		+3/-4

in the NVD for this experiment and the data used by Scout. To explore the reasoning and further assess the usefulness of Scout in comparsion to OpenVAS, a dissection of the results reported in Table III is needed. What follows is a detailed examination of the individual results, found in Table IV, V, VI and VII, comparing the vulnerability assessment tools to the NVD and between the two tools.

Table IV contains the results obtained from further analysis of vulnerability assessment reports on an Apache 2.4.7 service. This table shows the differences in results across the three collections. In comparison to the NVD, Scout returned all but 3 vulnerabilities, again suggesting there is a discrepency between the versions of NVD used. Table IV also shows OpenVAS reporting 4 vulnerabilities not found in the associated NVD entry. These vulnerabilities are CVE-2004-0230, CVE-2017-3167, CVE-2017-3169, and CVE-2017-7679. CVE-2004-0230 is associated with TCP/IPv4 methods rather than the Apache web service itself, due to this Scout does not report this vulnerability. The remaining 3 vulnerabilities were all vulnerabilities tied to the Apache service however, only CVE-2017-7679 has an explicit reference to the Apache 2.4.7 CPE. This seems to be an issue with NVD as the description of the vulnerabilities mentions Apache 2.4.7 being susceptible. In comparison to the NVD, OpenVAS did not identify 4 vulnerabilities which are associated with the service, CVE-2013-6438, CVE-2014-0098, CVE-2015-3184, and CVE-2016-0736. The explanation for which seems to be due to these vulnerabilities being tied to optional configurations.

When comparing Scout to OpenVAS it can be observed that Scout reports 4 vulnerabilities which OpenVAS does not, but omitted 7 vulnerabilities which OpenVAS reported. Using OpenVAS results as the ground truth in comparison to Scout's results allows for the use of Precision, Recall and F1 score to quantify accuracy, comprehensiveness and effectiveness [28]–[30]. To calculate these metrics, true positive, false positives and false negatives must be defined. True positives are results which appear in the output of both OpenVAS and Scout, false positives are results which are present in to only Scout's results and false negatives are results which OpenVAS returns but Scout does not. The workings and results for the metrics can

TABLE VI
CVE ASSIGNMENT FOR A LIGHTTPD 1.4.18 SERVICE

NVD	OpenVAS	Scout
	CVE-2004-0230	
CVE-2008-0983	CVE-2008-0983	CVE-2008-0983
CVE-2008-1111		CVE-2008-1111
CVE-2008-1270		CVE-2008-1270
CVE-2008-1531		
CVE-2008-4298		CVE-2008-4298
CVE-2008-4360	CVE-2008-4360	CVE-2008-4360
CVE-2010-0295	CVE-2010-0295	CVE-2010-0295
CVE-2011-4362		CVE-2011-4362
CVE-2013-1427		CVE-2013-1427
CVE-2013-4559		CVE-2013-4559
CVE-2013-4560		CVE-2013-4560
CVE-2014-2323	CVE-2014-2323	CVE-2014-2323
CVE-2014-2324	CVE-2014-2324	CVE-2014-2324
CVE-2015-3200	CVE-2015-3200	
Differences	**+1/-9**	**+0/-2**
		+8/-2

TABLE VII
CVE ASSIGNMENT FOR A LIGHTTPD 1.4.33 SERVICE

NVD	OpenVAS	Scout
	CVE-2004-0230	
CVE-2011-4362		
CVE-2013-4508		CVE-2013-4508
CVE-2014-2323	CVE-2014-2323	CVE-2014-2323
CVE-2014-2324	CVE-2014-2324	CVE-2014-2324
CVE-2015-3200	CVE-2015-3200	
Differences	**+1/-2**	**+0/-2**
		+1/-2

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

TABLE VIII
EQUATIONS RESULTS

Service	Precision	Recall	F1 Score
Apache 2.4.7	0.73	0.61	0.66
Apache 2.4.10	0.77	0.71	0.74
Lighttpd 1.4.18	0.33	0.67	0.44
Lighttpd 1.4.33	0.67	0.50	0.57

be observed in the following equations.

$$Precision = \frac{TP}{TP + FP} = \frac{11}{11 + 4} = \frac{11}{15} = 0.73 \quad (1)$$

(1) shows Scout has a precision of 73%. This is a significant result as this shows that contactless active reconnaissance can identifying vulnerabilities, usually found through active reconnaissance. Although as can be seen by the precision, these results may be inaccurate.

$$Recall = \frac{TP}{TP + FN} = \frac{11}{11 + 7} = \frac{11}{18} = 0.61 \quad (2)$$

(2) shows Scout has a recall of 61%. This is an important result as this quantifies the comprehensiveness of the results Scout. With a recall of 61% Scout is able to identify more than half, close to two thirds, of the applicable vulnerabilities for this Apache service.

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} = \frac{2 \times 0.73 \times 0.61}{0.73 + 0.61} = 0.66 \quad (3)$$

(3) shows Scout has an F-Measure of 66%, the effectiveness of Scout can be quantified as nearly two-thirds of OpenVAS however, this metric only applies to this configuration of an Apache 2.4.7 service.

The results of the equations associated with data provided in Table IV-VII are presented in Table VIII. Table VIII presents a variance in results. In overall effectiveness, measured by F1 score, the effectiveness ranged from a high of 74% for Apache 2.4.10 to a low of 44% for Lighttpd 1.4.18. Closer inspection of the data shows that reason for the moderate range of F1 score is due to the significant range in precision. Precision ranges from 33% to 77%, this 40% range is greater that of the range for Recall, which is 29%. Overall, these results indicate that Scout varies moderately in effectiveness as vulnerability assessment tool, this variance is due to the deviation in accuracy more so than the comprehensiveness.

D. Evaluation

From the prior experiments, the subsequent analysis and comparison, Scout has shown to be a useful vulnerability assessment tool. This can be supported through the comparative assessment where Scout performed better than OpenVAS by identifying more services. In the limited experiments in comparison to OpenVAS, a significant finding to emerge from this comparison was the effectiveness of Scout to identify known vulnerabilities. In this aspect Scout performed worse than OpenVAS, however, the effectiveness of Scout measured in F1-score ranged from 33-74%. Notwithstanding the limitation to services located on port 80 and the lack of consideration for vulnerabilities associated to transport medium, this comparison suggests that there is a moderate to significant usefulness to the results provided. This usefulness is enhanced by the novel contactless approach which by design mitigates any impact on the target network seen in a typical active vulnerability assessment tool.

TABLE IX
TOOL COMPARISON

Tool	Active	Passive	Custom Banner	CPE/CVE
Masscan	•••	•	•	•
ZMap	•••	•	•	•
Shodan	•	•••	•	••
Censys	•	•••	•	••
Nessus	•••	•	••	•••
OpenVAS	•••	•	••	•••
ShoVAT	•	•••	••	•••
Scout	•	•••	••	•••

Although no interaction takes place between the Scout user and the target systems, the information gained through the operation of Scout may be used in conjunction with more sophisticated tools and malicious intent [9]. The possibility of this is perhaps increased through the lack of complexity and knowledge required to operate Scout. Despite this, Scout has the potential to be a useful security tool for both network management and security auditing. Scout could be used in reaction to a high-profile vulnerability's release into the NVD to locate any possible exploitation on an administered network. Scout could be used in to demonstrate to organizations what a potential malicious actor can see and what they may try to exploit.

A natural progression of this project is to further develop Scout in an effort to improve effectiveness and usefulness. Greater effectiveness could be achieved through performing text analysis on the NVD to identify configuration specific vulnerabilities [32]. Greater usefulness could be achieved by allowing a larger number of data sources to be cross-referenced with a user-specified query, this could include incorporating more ports provided by Censys.

To show the nature of Scout in comparison to other tools, Table 9 contains a comparison comprising of tools mentioned previously.

VI. RELATED RESEARCH

Through a series of published works, Genge and Enăchescu introduced the novel idea of identifying vulnerabilities passively through Internet-wide scanning data, culminating in their tool known as ShoVAT [24], [33]–[35]. Until now, ShoVAT is the only published work about a tool relying solely upon an internet scanning project to identify known vulnerabilities. ShoVAT takes Shodan input, hence the name, and creates CPEs then associates them with known vulnerabilities. The methodology implemented utilizes a vital dependency on identifying version numbers, to correspond with an entry in a hash table containing possible CPEs. Scout

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

differs from ShoVAT by using Censys over Shodan, this effects the CPE manufacturing process as the banner for the service is already processed by Censys. This redirects the dependency from version numbers to vendor and product information. Which results in consideration for the erroneous NVD and incomplete service banner information. In the evaluation of ShoVAT, several experiments were undertaken however, these experiments could not be produced in this research due to the lack of data given. Therefore, for the following experiments, the data used is available via GitHub [27]. A criticism of research published about ShoVAT is that it focuses too heavily on the performance aspect of the tool over the accuracy [11]. As the performance of Scout is not within the scope of this paper, Scout was not evaluated against ShoVAT in this aspect. Williams, McMahon, Samtani, Patton and Chen use Shodan in conjunction with the vulnerability assessment tool Nessus to produce a scalable approach for identifying vulnerabilities in IoT devices [36]. Their approach employed the use of 3 resourceful computers which took IP addresses provided by IoT related Shodan queries which are then passed to Nessus. In similar research conducted by Al-Alami, Hadi and Al-Bahadili, their approach only employed the use of tailored queries and did not make use of vulnerability assessment tools [37].

They did not use an external vulnerability assessment tool.

VII. Conclusion

The aim of this research was to explore and evaluate how well Internet-wide scanning projects could be used to identify potential known-vulnerabilities. To accomplish this aim, the creation of a novel tool, Scout, was undertaken. Scout utilizes the Internet-wide scanning project Censys to source publicly available services and the NVD in an attempt to assign these services with applicable known vulnerabilities. The contactless active approach employed by Scout has shown to be successful by correctly identifying vulnerabilities in Internet services.

By investigating literature surrounding Internet-wide Scanning, the main tools Censys and Shodan were identified. Censys' was selected as the structured output data from the Internet-wide scanning it provided was more useful than the more extensive but raw Shodan information. Other instrumental findings to emerge from literature included that the NVD is in fact rather flawed, with many inconsistencies which impacted its usability for this type of work. To mitigate these the issues when using the NVD for service banner analysis, an approach from related work which utilized the Levenshtein distance algorithm to perform approximate matching was attempted [25].

The experiments undertook had a basis in research cited throughout the methodology, with alterations made to create a clear reproducible process [11], [24]. 3 distinct experiments were conducted. 2 CPE creations experiments, one focused on the accuracy and one experiment focused on obfuscated banners. The last experiment focused on the accuracy, comprehensiveness, and effectiveness in relation to CVE assignment. The reproducibility was furthered through the use of applicable scientific measurements to quantify results. These metrics allow for evaluation of similar tools such as OpenVAS. When comparing Scout to OpenVAS at the ability to identify known-vulnerabilities, Scout obtained an effectiveness of 74% for a specific Apache service, this is a remarkable result considering Scout does not correspond directly with the service. However, this effectiveness ranges across results with a 44% effectiveness observed against OpenVAS'findings on a Lighttpd 1.4.18. The reason for this difference in effectiveness is mostly due to the accuracy from which it is derived from, although the comprehensiveness of the results also plays a part. These statistics can also be used to evaluate the project as a whole against the aim of the project, which is to evaluate how well Internet-wide scanning results could be used to identify potential known-vulnerabilities in Internet-connected systems. In which it was successfully able to.

In future work, Scout will be updated with the ability to analyse more diverse services through the use of a combination of Internet-wide scanners including Shodan and ZoomEye. For avenues of future research, experiments could be repeated to further analyse the effectiveness of contactless active reconnaissance against alternative vulnerability assessment methods. This research could look to compare these tools across a diverse test bed, including more diverse services including IoT devices.

Scout is currently available to the public through the author's personal Github [27].

References

[1] K. Panetta, *Gartner's top 10 security predictions 2016*, Gartner blog 2016, available at https://www.gartner.com/smarterwithgartner/top-10-security-predictions-2016.

[2] N. Perlroth, and M. Scott, and S. Frenkel, *Cyberattack hits ukraine then spreads internationally*, The New York Times, 2017, available at https://www.nytimes.com/2017/06/27/technology/ransomwarehackers.html.

[3] Greenbone Networks, *OpenVAS*, 2005.

[4] Tenable, *Nessus*, 1998.

[5] Shodan.io, *Shodan*, 2009. [Online] Available: https://shodan.io. [Accessed 17- Mar- 2018].

[6] Censys.io, *Censys*, 2015. [Online] Available: https://censys.io. [Accessed 17- Mar- 2018].

[7] Z. Durumeric, and D. Adrian, and A. Mirian, and M. Bailey, and JA. Halderman, *A search engine backed by Internet-Wide scanning*, Proceedings of the 22nd ACM Conference on Computer and Communications, 2015, pp.542–553.

[8] J. Matherly, *Complete guide to Shodan*. Leanpub, 2017, p.3.

[9] A. Tundis, and W. Mazurczyk, and M. Mühlhäuser *A Review of Network Vulnerabilities Scanning Tools: Types, Capabilities and Functioning*, Proceedings of the 13th International Conference on Availability, Reliability and Security, 2018, pp65:1–65:10.

[10] Z. Durumeric, and F. Li, and J. Kasten, and J. Amann, and J. Beekmen, and M. Payer, and N. Weaver, and D. Adrian, and V. Paxson, and M. Bailey, and JA. Halderman, *The Matter of Heartbleed*, Proceedings of the 2014 Conference on Internet Measurement Conference, 2014, pp.475–488.

[11] K. Simon, and C. Moucha, and J. Keller, *Contactless vulnerability analysis using Google and Shodan*, Journal of Universal Computer Science, volume 23, issue 4, 2017, pp.404–430.

[12] ZoomEye.org, *ZoomEye - Cyberspace Search Engine*, 2011. [Online] Available: https://www.zoomeye.org. [Accessed 17- Mar- 2018].

[13] J. Matherly, *Inside the world's most dangerous search engine*, ShowMeCon, 2014.

[14] E. Bou-Harb, and M. Debbabi, and C. Assi, *Cyber scanning: a comprehensive survey*, IEEE Communications Surveys & Tutorials, volume 16, issue 3, 2014, pp.1496–1519.

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.

[15] R. Bodenheim, and J. Butts, and S. Dunlap, and B. Mullins, *Evaluation of the ability of the Shodan search engine to identify Internet-facing industrial control devices*, International Journal of Critical Infrastructure Protection, volume 7, issue 2, 2014, pp.114–123.
[16] V. J. Ercolani and M. W. Patton and H. Chen, Shodan visualized, 2016 IEEE Conference on Intelligence and Security Informatics (ISI), 2016, 193-195.
[17] greynoise.io, *Grey Noise Intelligence*, 2017. [Online] Available: https://greynoise.io. [Accessed 17- Mar- 2018].
[18] D. Adrian, and Z. Durumeric, and S. Gulshan, and JA. Halderman, *Zippier ZMap: Internet-wide scanning at 10 Gbps*, Proceedings of the 8th USENIX Conference on Offensive Technologies, USENIX, 2014.
[19] R. Graham, and P. McMillian, and D. Tentler, *Mass scanning the Internet*, DEFCON 22, 2014.
[20] R. Graham, *Masscan: designing my own crypto*, Errata Security Blog, 2013. [Online] Available: https://blog.erratasec.com/2013/12/masscan-designing-my-own-crypto.html. [Accessed 22- Feb- 2018].
[21] S. Na, and T. Kim, and H. Kim, *A Study on the Service Identification of Internet-Connected Devices Using Common Platform Enumeration*, Advanced Multimedia and Ubiquitous Engineering, 2017, pp.237–241.
[22] S. Na, and T. Kim, and H. Kim, *Service Identification of Internet-Connected Devices Based on Common Platform Enumeration*, Journal of Information Processing Systems, volume 14, issue 3, 2018, pp.740–750.
[23] M. Gawron, and F. Cheng, and C. Meinel, *PVD: passive vulnerability detection*, 8th International Conference on Information and Communication Systems (ICICS), 2017, pp.322-327.
[24] B. Genge, and C. Enăchescu, *ShoVAT: Shodan-based vulnerability assessment tool for Internet-facing services*, Security and Communication Networks, volume 19, issue 15, 2016, pp.2696–2719.
[25] LAB. Sanguino, and R. Uetz, *Software vulnerability analysis using CPE and CVE*, Computing Research Repository, 2017.
[26] A. Dulaunoy, and P. Moreels, and R. Vinot, *cve-search project*, 2016. [Online] Available: https://www.cve-search.org. [Accessed 29- Feb- 2018]
[27] J. O'Hare, *Scout*, 2018. [Online] Available: https://github.com/TheHairyJ/Scout. [Accessed 21- Aug- 2018]
[28] N. Antunes, and M. Vieira, *Assessing and Comparing Vulnerability Detection Tools for Web Services: Benchmarking Approach and Examples*, IEEE Transactions on Services Computing, volume 8, issue 2, 2015, pp.269–283.
[29] N. Antunes, and M. Vieira, *On the Metrics for Benchmarking Vulnerability Detection Tools*, 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2015, pp.505–516.
[30] P. Nunes, and I. Medeiros, and J. C. Fonseca, and N. Neves, and M. Correia, and M. Vieira, *Benchmarking Static Analysis Tools for Web Security*, IEEE Transactions on Reliability, volume 67, issue 3, 2018, pp.1159–1175.
[31] B. Cheikes, and D. Waltermire, and K. Scarfone *Common platform enumeration: Naming specification version 2.3. Technical Report NIST Inter-agency Report 7695*, NIST, 2011. [Online] Available: http:// csrc.nist.gov/publications/nistir/ ir7695/NISTIR-7695-CPE-Naming.pdf. [Accessed 29- Feb- 2018]
[32] S. Huang, and H. Tang, and M. Zhang, and J. Tian, *Text Clustering on National Vulnerability Database*, Second International Conference on Computer Engineering and Applications, 2010, pp.285–299.
[33] B. Genge, and F. Graur, and C. Enăchescu, *Non-intrusive Techniques for Vulnerability Assessment of Services in Distributed Systems*, Procedia Technology, volume 19, 2015, pp.12–19.
[34] B. Genge, and C. Enăchescu, *Non-Intrusive Historical Assessment of Internet-Facing Services in the Internet of Things*, 5th International Conference on Recent Achievements in Mechatronics, Automation, Computer Science and Robotics, volume 1, issue 1, 2015, pp.25–36.
[35] B. Genge, and P. Haller, and C. Enăchescu, *Beyond Internet Scanning: Non-Intrusive Vulnerability Assessment of Internet-Facing Services*, International Journal of Information Security Science, volume 4, issue 3, 2015.
[36] R. Williams and E. McMahon and S. Samtani and M. Patton and H. Chen, Identifying vulnerabilities of consumer Internet of Things (IoT) devices: A scalable approach, 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), 2017, 179-181.
[37] H. Al-Alami and A. Hadi and H. Al-Bahadili, Vulnerability scanning of IoT devices in Jordan using Shodan, 2017 2nd International Conference on the Applications of Information Technology in Developing Renewable Energy Processes Systems (IT-DREPS), 2017, 1-6.

Authorized licensed use limited to: DA IICT. Downloaded on August 06,2024 at 16:51:32 UTC from IEEE Xplore. Restrictions apply.