applied sciences

MDPI

Article

Efficient Algorithm for Providing Live Vulnerability Assessment in Corporate Network Environment

Michał Walkowski *,†, Maciej Krakowiak †, Jacek Oko † and Sławomir Sujecki †

Department of Telecommunications and Teleinformatics, Wroclaw University of Science and Technology, 50-370 Wroclaw, Poland; maciej.krakowiak@dsecure.me (M.K.); jacek.oko@pwr.edu.pl (J.O.); slawomir.sujecki@pwr.edu.pl (S.S.)

* Correspondence: michal.walkowski@pwr.edu.pl

† These authors contributed equally to this work.

Received: 15 September 2020; Accepted: 5 November 2020; Published: 9 November 2020

Featured Application: Vulnerability management center allows for the improvement of the quality and efficiency of operation for security operation centers.

Abstract: The time gap between public announcement of a vulnerability—its detection and reporting to stakeholders—is an important factor for cybersecurity of corporate networks. A large delay preceding an elimination of a critical vulnerability presents a significant risk to the network security and increases the probability of a sustained damage. Thus, accelerating the process of vulnerability identification and prioritization helps to red the probability of a successful cyberattack. This work introduces a flexible system that collects information about all known vulnerabilities present in the system, gathers data from organizational inventory database, and finally integrates and processes all collected information. Thanks to application of parallel processing and non relational databases, the results of this process are available subject to a negligible delay. The subsequent vulnerability prioritization is performed automatically on the basis of the calculated CVSS 2.0 and 3.1 scores for all scanned assets. The environmental CVSS vector component is evaluated accurately thanks to the fact that the environmental data is imported directly from the organizational inventory database.

Keywords: big data; complex system; cybersecurity; risk-based vulnerability management; data lifecycle; DLC; smart data; smart DLC

1. Introduction

Companies, governments and ordinary citizens notice the increasing importance of cybersecurity in daily life. During the first half of 2020, about 9799 new vulnerabilities were reported [1]. This indicates the increase of 34% in comparison with the same time span in 2019. The researchers active in the field of cybersecurity believe that in 2020 the number of detected vulnerabilities will reach another record value. Furthermore, due to the ongoing Covid-19 pandemic [1] and related increased use of internet services, the cybersecurity issues have the potential to affect a much larger, than in previous years, part of the human population. Consequently, the identification and prioritization of vulnerabilities becomes a critical issue for a company that offers internet services [1,2].

Vulnerability Management (VM) and Vulnerability Assessment (VA) are the proactive security layers against threats which commercial companies may face. In addition, they present a challenge for many organizations [3]. The first issue related to vulnerability management is the time passing between vulnerability identification and elimination. As Gartner explains, "most organizations follow a philosophy of gradual risk reduction, with vulnerability and patch management policies focused on mitigating and patching a percentage of vulnerabilities in a given time frame, for example, remediate

90% of high severity vulnerabilities within two weeks of discovery. This reduces vulnerability management to a pure metrics exercise, where risk is expressed as a numerical value that can be reduced." [4]. The fact is that in 90% of all cases a potential adversary is not going to be focused on the patched or remediated vulnerabilities but on the remaining 10%. Haldar and Mishra discuss in [5] the importance of the time reaction to new threats, stress the importance of quick vulnerability prioritization and patch, and show how short reaction time results in maximization of the effort required to breach the lines of defense. Other important points related to VM are [3]:

- there is no successful VM without effective communication,
- insufficient resources allocated to remediate the detected vulnerabilities, will cause vulnerability accumulation,
- fixing only "high" and "critical" vulnerabilities is not enough.

The above points stem from the fact that all vendors [6–9] provide their own methods of vulnerability prioritization without informing the end-user about the details of the decision making process. The unfamiliarity with prioritization algorithms may adversely affect the process of fixing the vulnerabilities since the companies that use particular software suite are left relying on unknown prioritization algorithms.

In this contribution authors have developed a distributed system—Vulnerability Management Centre (VMC) [10]—operating in a scalable, containerized environment. VMC allows the CVSS Environmental score to be calculated in an automatic manner. The developed VMC collects automatically information on vulnerabilities from publicly accessible sources. Then, VMC gathers information regarding vulnerabilities present in the system and integrates this data with the data obtained from the inventory database. Thus VMC is able to normalize accrued information and perform environmental calculations taking the relevant variables into consideration, e.g., Target Distribution (T_D) or Confidentiality, Integrity and Accessibility (CIA) triad. In addition, due to application of Smart Data algorithms [11,12] the developed VMC is capable of presenting results almost in real time to stakeholders. The small delay depends on computational resources available and the amount of data coming from a vulnerability scan. Thus, the developed VMC is vastly superior to standard systems whereby a monthly report is sent to the stakeholders each month based on previous month's data. Further, the developed VMC operates on normalized data, which renders the system independent of either the specific vulnerability scanner or asset management solution. A novel contribution of this work consists in performing automatic calculations of the environmental component of CVSS score vector by combining the data obtained from vulnerability scanner with the data retrieved from the inventory database. To the best of the authors' knowledge, such an approach has not been presented yet in the publicly available literature.

An additional novel aspect of the present paper unfolds in the context of data life cycles presented in [11], which pertains to Smart Data and consists in retrieving knowledge from "the mass of initially unstructured data" [12] collected by VMCs, i.e., the results of vulnerability scanning, vulnerability related classification of data stemming from several publicly accessible databases and integration with information on organization's assets within the organization-specific context. In this contribution, the initially completely unrelated data, gathered by VMC is structured, normalized and filtered to bring in value and novel knowledge relevant specifically to vulnerability management.

This article is organized as follows:

- Background—section describes foundations of this research, introduces the problems, processes and trades off that are present in vulnerability management research.
- Related Work—section presents other work related to the present topic. It is a brief description of work related vulnerability management.
- System Data Life Cycle and Analysis—section presents data life cycle and analysis of the proposed framework.

- VMC Implementation and Experiment Design—section shows an experiment design for conducted research.
- Results—section starts the discussion about the results illustrating the advantages of the proposed VMC system, shows a summary of the presented work.
- Conclusions—section gives the summary, starts critical discussion about the presented solution, and introduces fields for further research.

2. Background

In order to understand the concept of vulnerability management, one should begin with learning what the vulnerability in a computer system is and how it is marked using the Common Vulnerability Enumeration (CVE) [13] and assessed using Common Vulnerability Scoring System (CVSS) [14].

According to [15], software vulnerabilities "are software bugs that expose weaknesses in software systems". Consequently, software vulnerabilities are directly linked to the software development process. Authors in [16] point out that discovering bugs, problems, and vulnerabilities during the software development process is time consuming. Moreover, the insights regarding vulnerabilities and defects are frequently not pointed out by the development team but usually come from independent researchers.

The idea for creating a consistent standard for marking the vulnerabilities was presented in [13]. The authors of the CVE concept discovered the need for introducing a consistent way of marking vulnerabilities in order to improve the internal communication within the organization. Each security system that was used and attempted to be integrated, consisted of its public and nonpublic vulnerability database [13,17,18]. The largest problem faced was the lack of common naming convention and vulnerability identification. As a result, the data comparison for different providers and linking this data with other security systems to minimize the risks, was very time consuming [13,17]. The Common Vulnerability Enumeration (CVE) concept was accepted by the industry and literature to such an extent that it has found its usage not only in Intrusion Detection Systems (IDSs) [19] but in every cybersecurity related field [20–22]. Further, since 1999 many computer security providers and nonprofit organizations have been developing, promoting, and implementing the diverse systems of vulnerability assessment: X-Force [23], Symantec [24], Microsoft [25], Redhat [26], Mozilla [27], Secunia [28], Vulpen [29], Google [30], VRSS [29], CVSS [31]. Currently [23–27,30], many computer security providers are still maintaining research departments; however, support for many past solutions has been discontinued [31]. For instance,[28] is no longer developed while [29] has not been adopted. The Common Vulnerability Scoring System (CVSS) on the other hand, was introduced for the first time as a research project by the US National Infrastructure Advisory Council (NIAC) in 2005 [31] and adopted subsequently by other organizations. The CVSS 2.0 and CVSS 3.1 versions [32,33] are divided into three categories:

- Base
- Temporal
- Environmental

Base category represents properties of the vulnerability that do not change in time. These properties consist of access complexity, access vector, and assess the degree to which a vulnerability compromises the confidentiality, integrity, and availability of the system. Temporal category describes properties that may change over time. In particular, the temporal category refers to the existence of a public exploit and a patch or fix availability. The temporal characteristics of CVE were studied specifically by Ruohonen in [15] while in [19] researchers aimed to express the value risk, potential loss, and prevalence of affected systems in the considered environment.

Beyond any doubt, the Vulnerability Management (VM), an essential part of maintaining the security of an organization [34,35], is threatened by growing cybercrime [36]. The identification and mitigation of vulnerabilities in specific or critical systems reduces the risk of exploitation impact during

a potential attack [37]. Therefore, it is crucial that leading Information Technology (IT) organizations and network administrators aim for zero vulnerabilities in managed systems. The VM process should be implemented practically in every organization that uses IT infrastructure. For instance, current corporate networks consist of thousands of devices and applications, without which business processes cannot function, whilst even a temporary unavailability of services rendered may result in large financial losses and reputation damage [35]. However, the critical importance of VM also applies to other entities ranging from office networks, to financial and personnel systems, to very specialized systems (e.g., industrial/process control, weapons, telecommunications, and environmental control systems) [38]. Thus, due to increasing threats and known vulnerabilities, an organization must have a vulnerability management system or a process that provides the latest security patches and updates to the organization's network [39]. In general, the purpose of VM is to monitor and identify new threats and vulnerabilities (hardware and software) that may affect the confidentiality, integrity, or availability of an organization's IT resources. In addition, VM should help system administrators to identify existing and known vulnerabilities and apply appropriate actions to reduce the risk of vulnerability exposure [40]. The undertaken actions may consist of patching vulnerabilities or taking other actions, if a vulnerable system cannot be repaired due to operational constraints, or the patch causes key services to be unavailable. If a vulnerable system cannot be repaired, system/network administrators should create a plan to mitigate every vulnerability that cannot be eliminated [41]. Mitigation plans may consist of blocking rules on Intrusion Prevention system (IPS)/Intrusion Detection System (IDS), moving the system to a separate Virtual Local-Area Network (VLAN), significantly restricting or blocking ports on the firewall, or even removing the system from the publicly available part of the network until appropriate corrective measures are implemented.

Further, from a practical point of view it is important to scan as much of the network as possible (preferably the whole network each time). Using the VM system presented in this contribution, if only a fraction of the network is scanned within the VM process then by viewing the scan results and comparing them with the assets via asset management tool, one can estimate the percentage of scanned devices and determine the overall network status and network hygiene level. Finally, in almost every environment it is presumed that devices will be turned off, disconnected from the network or put in a transient state during the scan, so scans should take place regularly (as often as possible) and maintain a vulnerability history.

3. Related Work

Many authors [42,43] stress the fact that in order to prioritize the vulnerability correctly, organizations should consider an asset value and a vulnerability importance in a standardized way. The problem of vulnerability prioritization has been discussed in available literature for a long time [44–47]. Most established companies that take cybersecurity seriously into consideration have a vulnerability management process implemented to a greater or lesser extent [3,42,44]. However, as noticed in [3,44–47] each organization approaches the problem differently. The solutions enlisted in the report [48] F-Secure [6], Qualys [7], Rapid7 [8], or Tenable [9], on the one hand help the organizations to overcome the vulnerability management problem but on the other one, they have two drawbacks. Namely, they are very expensive and they do not inform users on the details of the prioritization procedure. For instance, Qualys uses a 7-point scale [7], Rapid7 performs the prioritization in the range from 1 to 1000 [8], whereas Tenable named its prioritization method VPR and the provided levels range from 1 to 10 [9]. Next to commercial solutions, it is possible to find in literature, other solutions, i.e.,: PatchRank [49], SecureRank [50], VULCON [37], or VEST [51]. The PatchRank solution focuses only on the updates prioritization for SCADA systems [49]. The SecureRank uses the network topology and the potential interaction between servers to calculate their risk [50]. The Vulcon's software strategy is based on two elementary pointers: the vulnerability appearance time (TVR) and total vulnerability exposure time [37]. VEST, however, focuses on verifying whether the vulnerability is exploited and how quickly it can be used by an attacker [51]. Other solutions, e.g., [37,49–51]

do not take into consideration the value of assets and are not adjusted to the increasing amount of data in cloud computing environment and therefore they cannot be applied in every network infrastructure. Additionally, none of the presented solutions offer prioritization for CVSS 2.0 and CVSS 3.1 simultaneously. This is due to the fact that not all vulnerabilities were converted from CVSS 2.0 to CVSS 3.1, even though CVSS 3.1 assesses the essence of vulnerability in a better way and estimates threats more efficiently [18].

An important characteristic of the VMC developed in this contribution is solving the problem of scalability. Thus, allowing to adjust the tool easily to the increasing amount of incoming data. In comparison to [37,49–51] the developed VMC uses the information collected from the asset database. Unlike [6–9] the prioritization procedure has been outlined in detail and hence can be analyzed using FIRST standard [14]. Another aspect of the developed VMC's operational activities is handling the big sets of data. According to results and discussions presented in available literature, a significant element of managing big sets of data is their life cycle. In [52] the authors examined over a dozen different data life cycles and their phases aiming to find the ones that "makes data Smart and thus facilitate their management in the Big Data context". "Smart" meaning the aforementioned knowledge obtained from the big and initially unstructured data set. According to [11] "the phases which constitute the data life cycle in a Big Data context are very complex. Each phase is considered as one or more complex, operational, and independent processes, but these processes are linked to one another and to make data management more flexible and smart". Thus, presenting the data life cycle is not a trivial task. The developed VMC as an open source product is characterized by transparency of the used methods and techniques. The developed VMC has been based on, so-called, Smart Data Lifecycle presented in [11]. Its aim is to present the data life cycle as a process, in order to increase its flexibility and ability of adjusting to different cases. The description of the particular life cycle phases is presented in the following sections.

4. System Data Life Cycle and Analysis

In this section, the system data life cycle and analysis of the proposed Vulnerability Managements Centre (VMC) is described. VMC consists of four core modules: Knowledge Collector, Asset Collector, Vulnerability Collector, and Processing Module. The first three modules are responsible for data collection, integration, and filtering while the Processing Module enriches the data with the CVSS environmental results.

All modules work independently [53], communicating asynchronously via a queue system. Consequently, the software is vertically scalable and the system is configured from the administrator panel. VMC also has two administrational modules, i.e., Scheduler—which controls the sequence and the time of data collection from particular sources and Task Monitor—to provide a preview of the current state of the system.

The software has been prepared to operate in the cloud computing environment and is based on Docker container technology [54]. All data is stored in the form of documents in Elasticsearch [55] that enables its processing in full-text mode, while the Kibana tool [56] is used to analyze and present the results. The software was implemented in Python due to its flexibility and ability to process data on the server side efficiently. The whole project includes multi tenancy support allowing for comprehensive data separation between the documents. Additionally, in VMC there are two data life cycles: operational and historical. Due to this fact, an operating engineer is able to see all changes immediately. The developed VMC allows previewing data and historical calculations, facilitating, at the same time, the analysis of the events occurring in the system. The last step, visualization, is done by Kibana (Figure 1). In order to simplify the principle of the data flow and VMC architecture, the Vulnerability Management Center is presented in the form of separate modules.

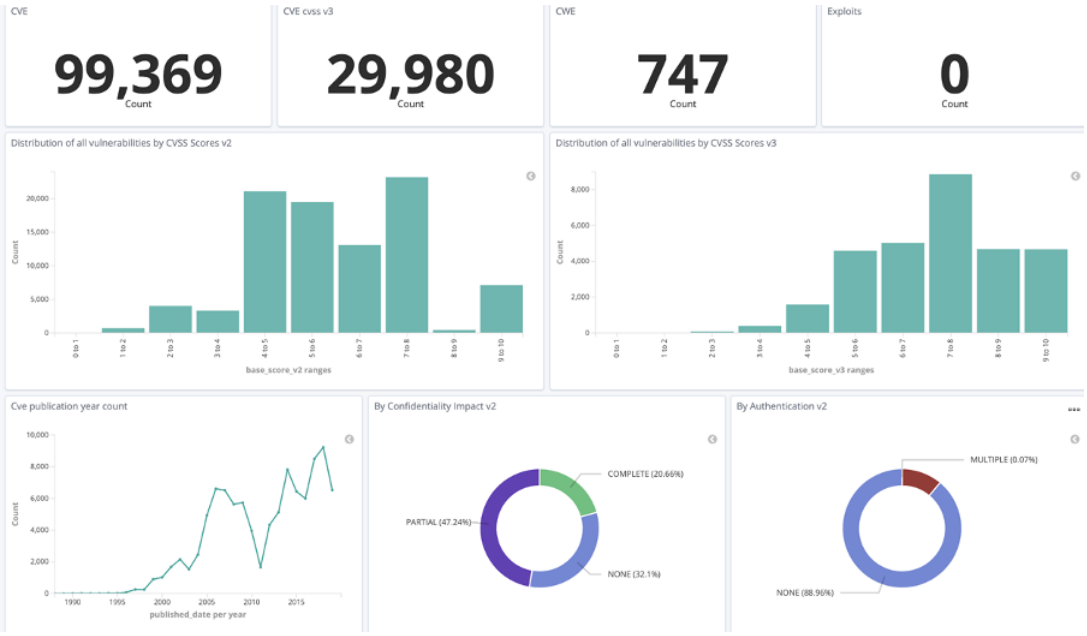


Figure 1. An example of data gathered by knowledge collector module.

4.1. Knowledge Collector Module

Knowledge Collector Module is responsible for collecting, integrating, and filtering data from publicly available databases regarding known exploits and Common Vulnerabilities and Exposures (CVE) [14], among others, such as National Vulnerabilities Database (NVD) [57] and Exploits Database [58]. Collecting is done with publicly accessible API, files, and web scraping. Then data has to be integrated, CVE and Exploits are matched, and previous existing data has to be updated. In the filtering phase all rejected [14] CVE or Exports are skipped to remove unnecessary information. The proposed document [55] stores the complete vector for CVSS 2.0 and CVSS 3.1 in the separate fields constituting the vulnerability assessment [14], in order to accelerate the calculations and facilitate easier vulnerability search, taking in that way its characteristic, e.g., remote usage. Using vertical scaling and triggering integration only for new or changed CVEs and Exploits reduces the time gap between the data collection and presentation of information on critical vulnerabilities to stakeholders [39].

4.2. Asset Collector Module

Asset Collector Module is responsible for collecting, integrating, and filtering data that involve detected and defined assets for monitored network. The module collects data from two data sources. The first source is Configuration Management Database (CMDB) [59]. The second source is vulnerability scanner [60,61] that is not only able to scan but also has the functionality of detecting the components. In consequence, it is possible for VMC to inform the operator about data incompatibility between CMDB and scanning results. That is one of the first knowledge enrichment manifestation the VMC can present while analyzing collected data. In order to explicitly determine an asset identifier (id), the universally unique identifier (uuid) generator version 3 [62] was used, based on the asset IP address and id value received from CMDB. The proposed document, except for fields needed by CVSS standards, contains also business and technical owner fields, which hold information about the person responsible for the monitored asset.

4.3. Vulnerability Collector Module

Vulnerability Collector Module collects data via accessible API from a vulnerability scanner [60,61]. During the filtering phase, the vulnerabilities classified as informational, i.e., with base CVSS 2.0 and

3.1 equal to 0, are excluded. This phenomenon is caused by the fact that informational findings do not provide any additional value to the vulnerability assessment and represent considerable volume of data (even 85% of all reported findings per host). In the integration phase, vulnerability collector module updates all existing vulnerabilities and assets received from previous scans. In order to explicitly determine the vulnerability identifier, the uuid was used, based on the IP address of the scanned machine and plugin id received from the corresponding scanner. The prepared document also contains an environmental score vector field that includes an explicit description of CVSS components for the calculated score.

4.4. Processing Module

The Processing Module is responsible for data enrichment when new data occurs. Using the system architecture designed to operate in the cloud computing network, an algorithm was implemented to process large amounts of data that depends on processing module configuration. Firstly, the algorithm downloads the number of vulnerabilities which have not been fixed or have been marked as removed. Then, the algorithm retrieves information stored in the system to assess the amount of available resources which can be used for calculations. The task division subject to available resources is assigned according to Equation (1).

$$d = \begin{cases} v//t & \text{if } v//t \leq t \\ t & \text{other} \end{cases} \quad (1)$$

where:

d the number of data processed by one processing module
t the number of available processing modules
v the number of vulnerabilities that are not fixed or removed

Each time before calculations begin, the verification of the Equation (1) allows for vertical scaling without restarting the system. In order to accelerate the target distribution [33] value calculations, every CVE query is stored in the cache that is shared by all counting modules. As a result, only one counting module sends the query to the database at a time and the rest of the modules retrieve this information from cache. The vulnerability scores that have already been calculated are saved in a separate thread by bulk method [63]. As a result, the calculating loop does not require waiting for the result of the save operation.

5. VMC Implementation and Experiment Design

The VMC system was implemented on Microsoft Azure® Free Tier subscription [64]. Due to encountered limitations, the software was launched in two regions: US West and US West 2 that demonstrated the average latency of 22 ms during tests [65]. In region US West 2 the kubernetes cluster (k8s) was launched [66].

The US West 2 cluster consists of 2 servers each with 2× CPU Intel Xeon® E5-2673 v3 (Haswell) 2.4 GHz processors and 7 GB RAM memory and supports the following services:

- VMC processing module
- PostgreSQL database—storing VMC configurations
- MariaDB database—storing CMDB information
- Ralph—CMDB administration panel
- Rabbitmq—queue system used for communication between VMC modules
- Redis—in-memory base used for partial calculations storage and mutex support in VMC modules
- VMC monitor—the monitoring of tasks performed by VMC
- VMC admin panel—module for VMC management

In the region of US West Elasticsearch cluster was launched that consists of 2 servers with 1× CPU Intel Xeon® E5-2673 v3 (Haswell) 2.4 GHz processor and 3.5 GB RAM memory. Between the regions US West and US West 2 ,the network type virtual-network to virtual-network (VLan) has been created.

Thus, in summary, all components of the developed VMC run autonomously within a computer cloud environment. This approach allows for performing the vulnerability prioritization in a fully automatic manner. In the following part of this section the numerical experiments are described that show the relevance of each component of the system and the advantage of parallel processing. The application of parallel processing shortens the processing time and allows for an elastic response to increased data processing demand.

Thus, in order to show the relevance of automatic integration of the asset collector module, a network model with the distribution of operating systems was used as described in [67]. In order to investigate the behavior and the execution time of the proposed algorithms in the context of smart data, the network model was created containing 2110 IP addresses and had a simplified distribution of operating systems described in Table 1. The network contained 168,940 vulnerabilities of which 3008 vulnerabilities are unique with the distribution presented in Table 2.

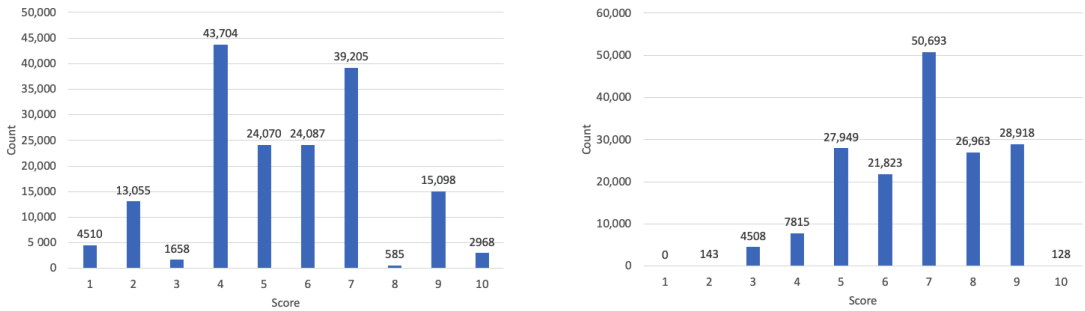
Table 1. Distribution of operating systems.

Name	Value	Name	Value
Redhat 5	18.48%	IBM AIX 6	4.27%
Redhat 6	19.67%	IBM AIX 5	5.69%
Redhat 7	18.96%	IBM AIX 7	4.03%
Windows Server 2016	7.82%	Debian 8	2.13%
Windows Server 2019	8.06%	Debian 9	1.41%
Windows Server 2012	7.58%	Debian 10	1.9%

Table 2. Distribution of vulnerabilities with operating systems division.

Name	Value	Name	Value
Redhat 5	6.53%	IBM AIX 6	0.8%
Redhat 6	27.33%	IBM AIX 5	1.11%
Redhat 7	26.46%	IBM AIX 7	1.01%
Windows Server 2016	13.11%	Debian 8	2.32%
Windows Server 2019	8.6%	Debian 9	1.8%
Windows Server 2012	10.03%	Debian 10	0.9%

Figure 2 shows CVSS Base histograms for the proposed model. With this set of vulnerabilities, three configurations of the CIA distributions as described in Tables 3–5 were studied.



(**a**) CVSS Base 2.0 histogram.

(**b**) CVSS Base 3.1 histogram.

Figure 2. CVSS Base histograms.

Table 3. Distribution of CIA requirements for configuration I.

Name	Low	Medium	High	N.D.
Confidentiality	25.36%	22.99%	23.7%	27.96%
Integrity	22.99%	25.12%	25.12%	26.78%
Availability	23.93%	25.83%	30.33%	19.90%

Table 4. Distribution of CIA requirements for configuration II.

Name	Low	Medium	High
Confidentiality	10.19%	7.82%	81.99%
Integrity	8.29%	10.9%	80.81%
Availability	9.25%	10.66%	80.09%

Table 5. Distribution of CIA requirements for configuration III.

Name	Low	Medium	High	N.D.
Confidentiality	76.3%	9%	6.88%	7.82%
Integrity	74.64%	8.06%	8.77%	8.53%
Availability	73.22%	9.48%	8.53%	8.77%

Then, to test the advantages of vertical scaling for the processing module, the time gap between an occurrence of a case (P) and obtaining the final results concerning the CVSS environmental assessment was measured. For this purpose 12 test cases were considered:

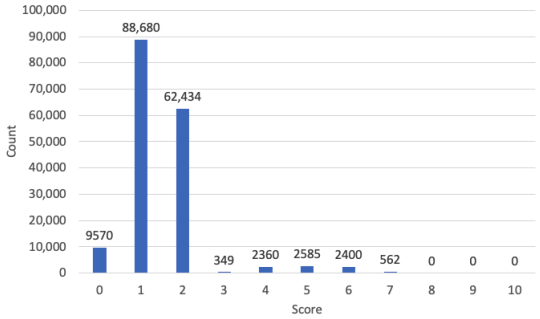
- P_0—prioritization for the initial state,
- P_1—CIA value change for 10% of assets,
- P_2—CIA value change for 20% of assets,
- P_3—CIA value change for 30% of assets,
- P_4—10% of assets marked as DELETED,
- P_5—20% of assets marked as DELETED,
- P_6—30% of assets marked as DELETED,
- P_7—the increase of new vulnerabilities by 10%,
- P_8—the increase of new vulnerabilities by 20%,
- P_9—the increase of new vulnerabilities by 30%,
- P_{10}—10% of vulnerabilities marked as FIXED (fixing the vulnerability),
- P_{11}—20% of vulnerabilities marked as FIXED (fixing the vulnerability),
- P_{12}—30% of vulnerabilities marked as FIXED (fixing the vulnerability).

The simulations were repeated three times and afterwards the average value of the simulation time was calculated for each P. Each time simulations were repeated the VMC software was restarted in order to exclude the influence of optimizations performed automatically by autonomous VMC components, which are not the subject of the presented research, e.g., each time Elasticsearch handles the same request it uses cache to speed up operation. Such optimization of course influences the measured CPU time and thus distorts the calculated results and hence should be prevented from taking place.

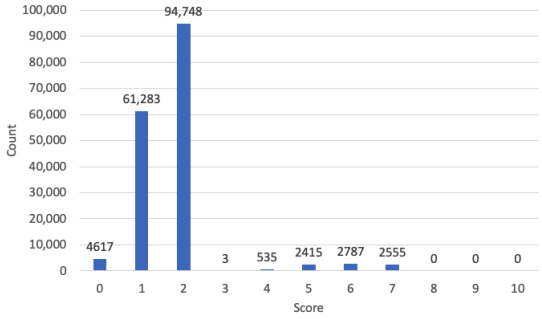
6. Results

In this section, the main purpose is to verify the operation of the proposed software and for this purpose the test cases described in the previous section were used. First, the influence of the asset collector module on the CVSS scores is discussed. Figure 3 presents the CVSS 2.0 scores obtained for all 3 considered configurations (Tables 3–5). When compared with Figure 2a, significant

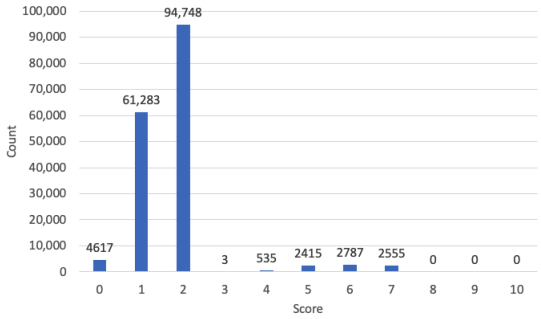
difference in calculated CVSS scores is observed for all considered configurations. The highest CVSS 2.0 environmental assessments received for the tested configurations are 7.5 (High) [14]. The next step in analyzing the impact of CVSS factors on the threat assessment generated by the vulnerability is considering the confidentiality, integrity, and availability requirements. Figure 3a shows the result of vulnerability prioritization including CVSS environmental 2.0 vector element for an equal distribution of all CIA components (Table 3). The obtained results indicate that only 0.33% of vulnerabilities have high priority in comparison to 34% obtained for CVSS base 2.0 score (Figure 2a). Figure 3b shows that 1.51% of vulnerabilities receive a high priority score (the scoring higher or equal to 7) (Table 4). For configuration III (Table 5), with 70% of the CIA components having LOW level, the results indicate only 0.11% of vulnerabilities with high CVSS score.



(**a**) Configuration I.



(**b**) Configuration II.



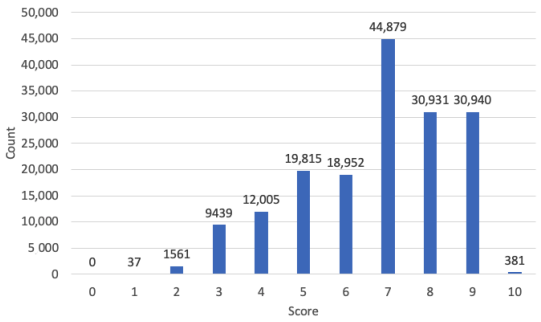
(**c**) Configuration III.

Figure 3. CVSS Environmental 2.0 histograms.

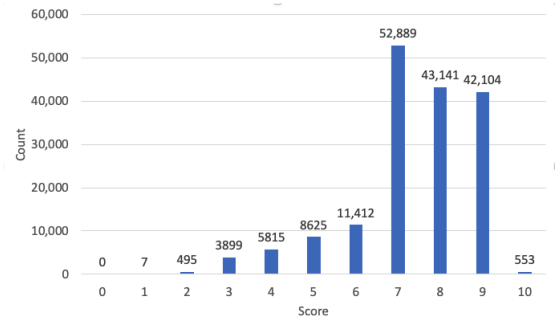
For CVSS environmental 3.1 scoring the observed changes in prioritization in the applied configurations are:

- configuration I, the decrease of critical and high vulnerabilities by 30% (Figure 4a) in comparison to CVSS base 3.1 (Figure 2b),
- configuration II, the increase of critical and high vulnerabilities by 30% (Figure 4b) in comparison to CVSS base 3.1 (Figure 2b),
- configuration 3, the decrease of critical and high vulnerabilities by 50% (Figure 4c) in comparison to CVSS base 3.1 (Figure 2b).

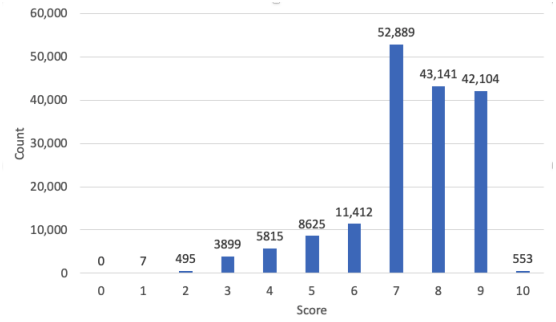
Thus the impact of integrating CVSS base scores with information available from CMDB is large so that a significant reprioritizing after including CVSS environmental information has to take place. It is noted, however, that depending on the nature of the monitored infrastructure, the distribution of CIA values may differ from the one adopted in the research. Nonetheless, results obtained confirm that maintaining an up-to-date CMDB database and its integration with the vulnerability scan results increases the level of security services.



(**a**) Configuration I.



(**b**) Configuration II.



(**c**) Configuration III.

Figure 4. CVSS Environmental 3.1 histograms.

Next, the results concerning vertical scalability of the software are considered. Figure 5 shows the influence of changes on time consuming proposed test cases in section VMC Implementation and Experiment Design. P_0 measurements were conducted to relate to changes caused by P to the initial phase of the system. Test cases P_4, P_5, P_6, P_{10}, P_{11}, P_{12} reduce the amount of active vulnerabilities for which calculations must be made. Therefore, the reprioritization time is lower than the one obtained for P_0. For P_1, P_2, P_3 the calculation time is similar to P_0 and stems from the nature of CVSS 2.0 environmental assessment. The influence of the CIA changes in CMDB is unnoticeable and concerns only the additional overhead related to updating data related to individual IP addresses. A significant time increase compared with P_0 case was noticed for P_7, P_8, P_9 cases since these latter cases add new vulnerabilities to the modeled network. For all cases considered, the most significant decrease in the calculation time is observed after adding 2 processing module instances (Calculation time reduction circa 45%.) The optimal number of processing module instances of the cloud computational environment described in Experiment Design section totals 5 (the average calculation time reduction by 65%). The highest calculation time reduction of 70% was obtained for P_{12} and 6 processing module instances.

In comparison to solutions presented in literature [37,49–51] the developed VMC software package takes into consideration the value of assets and is adjusted to the increasing amount of data in cloud computing environment. Therefore, the developed VMC can be applied for every network infrastructure. Additionally, none of the presented solutions [37,49–51] and [6–9] offer prioritization for CVSS 2.0 and CVSS 3.1 simultaneously. Thanks to the use of CVSS environmental score, the stakeholders using the developed VMC fully understand the nature of received vulnerability prioritization and can trace back all steps for the all received scores.

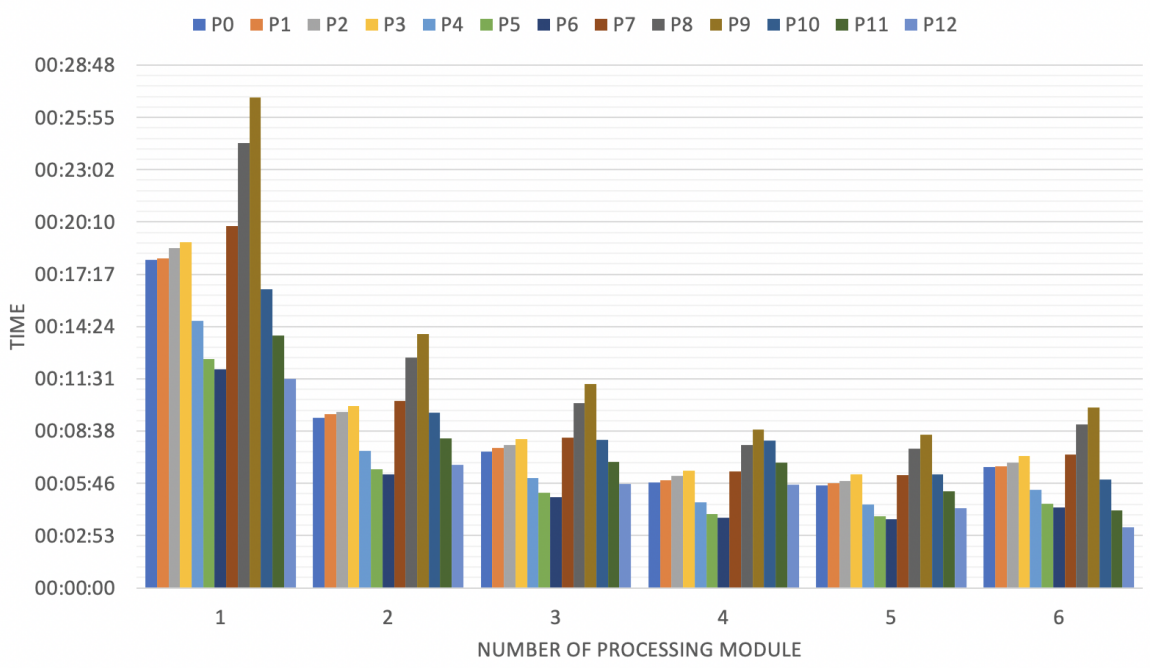


Figure 5. The time consumption in proportion to the number of processing module used and proposed *P*.

7. Conclusions

The obtained results present a fully automated vulnerability management platform. It was demonstrated that the inclusion of an asset collector module has a major impact on the CVSS scores. This fact shows that it is possible to maintain and operate with an open software platform that uses a well known open scoring system presenting reliable results. The vertical scalability of the developed software was also studied and demonstrated that the developed VMC is capable of processing the increasing amount of data. The results obtained show also that the developed VMC supports VM automation by:

- calculating CVSS Environmental vector component, reducing thereby the workload for stakeholders,
- fully scalable implementation, thus enabling processing large amounts of data in corporate environments [31],
- creating the dynamic metrics adjusted to corporate requirements.

In conclusion, comparing to solutions presented in literature [37,49–51] the developed VMC software package takes into consideration the value of assets and is adjusted to the increasing amount of data in cloud computing environment. Therefore, the developed VMC can be applied for every network infrastructure. Additionally, none of the presented [37,49–51] and [6–9] offer prioritization for CVSS 2.0 and CVSS 3.1 simultaneously.

The future work will focus on developing machine learning algorithms that predict each CVSS 3.1 vector component derived from public CVE databases. This issue is very important because all the vulnerabilities have not been evaluated using CVSS 3.1 metric and most of the available scores are based on CVSS 2.0 [18] metric only. However, the predicted vector components of CVSS 3.1 allow calculating the environmental score for all vulnerabilities coming from scanning software. This may further reduce time-to-vulnerability-remediation and total-vulnerability-exposure and facilitate the network administration by bringing focus to genuine deficiencies present within the infrastructure.

Author Contributions: All authors have read and agreed to the published version of the manuscript.

Funding: Wrocław University for Science and Technology, grant No CYBERSECIDENT/381690/ II/NCBR/2018 from National Centre for Research and Development within the CyberSecIdent-Cybersecurity and e-Identity program.

Acknowledgments: The authors wish to thank Wroclaw University of Science and Technology (statutory activity) for financial support and Agata Szewczyk for proofreading and translation. This publication was created as a part of the Regional Security Operations Center (RegSOC) project (Regionalne Centrum Bezpieczeństwa Cybernetycznego), cofinanced by the National Centre for Research and Development as part of the CyberSecIdent-Cybersecurity and e-Identity program.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

The following abbreviations are used in this manuscript:

API	Application Programming Interface
CIA	Confidentiality, Integrity and Accessibility
CMDB	Configuration Management Database
CVE	Common Vulnerabilities and Exposures
CVSS	Common Vulnerability Scoring System
DCIM	Data Center Infrastructure Management
ID	Identification
IDS	Intrusion Detection System
IoT	Internet of Things
IPS	Intrusion Prevention system
IT	Information Technology
K8S	Kubernetes cluster
NVD	National Vulnerabilities Database
OVM	Ontology for Vulnerability Management
P	Proposed Test cased
SOC	Security Operations Center
TD	Target Distribution
US	United States
VA	Vulnerability Assessment
VLAN	Virtual Local-Area Network
VM	Vulnerability Management
VMC	Vulnerability Management Centre

References

1. SkyboxR Research Lab. Vulnerability and Threat Trends; Technical Report. 2020. Available online: https://lp.skyboxsecurity.com/rs/440-MPQ-510/images/Skybox_Report_2020-VT_Trends.pdf (accessed on 15 October 2020).
2. Yang, H.; Park, S.; Yim, K.; Lee, M. Better Not to Use Vulnerability's Reference for Exploitability Prediction. *Appl. Sci.* **2020**, *10*, 2555. [CrossRef]
3. Gartner Research. A Guidance Framework for Developing and Implementing Vulnerability Management. Available online: https://www.gartner.com/en/documents/3747620 (accessed on 15 October 2020)
4. Rochford, O.; Threat-Centric, T. Vulnerability Remediation Prioritization. *J. Abbr.* **2008**, *10*, 142–149.
5. Haldar, K.; Mishra, B.K. Mathematical model on vulnerability characterization and its impact on network epidemics. *Int. J. Syst. Assur. Eng. Manag.* **2017**, *8*, 379–382. [CrossRef]
6. F-Secure. Vulnerability Management Tool. Available online: https://www.f-secure.com/us-en/business/solutions/vulnerability-management/radar (accessed on 15 October 2020)
7. Qualys. Vulnerability Management Tool. Available online: https://www.qualys.com/apps/vulnerability-management/ (accessed on 15 October 2020)
8. Rapid7. Vulnerability Management Tool. Available online: https://www.rapid7.com/products/nexpose/ (accessed on 15 October 2020)
9. Tenable. Vulnerability Management Tool. Available online: https://www.tenable.com/products/tenable-io (accessed on 15 October 2020)

10. VMC: A Scalable, Open Source and Free Vulnerability Management Platform. Available online: https://github.com/DSecureMe/vmc (accessed on 11 May 2020).
11. El Arass, M.; Souissi, N. Data Lifecycle: From Big Data to SmartData. In Proceedings of the 2018 IEEE 5th International Congress on Information Science and Technology (CiSt), Marrakech, Morocco, 21–27 October 2018; pp. 80–87. [CrossRef]
12. Lenk, A.; Bonorden, L.; Hellmanns, A.; Roedder, N.; Jaehnichen, S. Towards a taxonomy of standards in smart data. In Proceedings of the 2015 IEEE International Conference on Big Data (Big Data), Santa Clara, CA, USA, 29 October–1 November 2015; pp. 1749–1754.
13. Mann, D.E.; Christey, S.M. Towards a common enumeration of vulnerabilities. In Proceedings of the 2nd Workshop on Research with Security Vulnerability Databases, West Lafayette, Indiana, 21–22 January 1999.
14. Common Vulnerability Scoring System. Available online: http://www.first.org/cvss (accessed on 24 April 2020).
15. Ruohonen, J. A look at the time delays in CVSS vulnerability scoring. *Appl. Comput. Inform.* **2019**, *15*, 129. [CrossRef]
16. Morrison, P.J.; Pandita, R.; Xiao, X.; Chillarege, R.; Williams, L. Are vulnerabilities discovered and resolved like other defects? *Empir. Softw. Eng.* **2018**, *23*, 1383–1384. [CrossRef]
17. Martin, R.A. Managing vulnerabilities in networked systems. *Computer* **2001**, *34*, 32–38. [CrossRef]
18. Fall, D.; Kadobayashi, Y. The Common Vulnerability Scoring System vs. Rock Star Vulnerabilities: Why the Discrepancy? In Proceedings of the 5th International Conference on Information Systems Security and Privacy—Volume 1: ICISSP, Prague, Czech Republic, 23–25 February 2019; pp. 405–411.
19. Mell, P.M. An Overview of Issues in Testing Intrusion Detection Systems. NIST Internal Report 7007. Available online: https://nvlpubs.nist.gov/nistpubs/Legacy/IR/nistir7007.pdf (accessed on 24 April 2020)
20. Kaya, K. A Study of Vulnerabilities and Weaknesses in Connected Cars. Bachelor's Thesis, KTH, School of Electrical Engineering and Computer Science (EECS), Stockholm, Sweden, 2019.
21. U.S. Food and Drug Administration. *Postmarket Management of Cybersecurity in Medical Devices: Guidance for Industry and Food and Drug Administration Staff*; U.S. Food and Drug Administration: Silver Spring, MD, USA, 2016.
22. Wang, W.; Gupta, A.; Niu, N. Mining Security Requirements from Common Vulnerabilities and Exposures for Agile Projects. In Proceedings of the 2018 IEEE 1st International Workshop on Quality Requirements in Agile Projects (QuaRAP), Banff, AB, Canada, 21 August 2018; pp. 6–9,
23. IBM X-Force Threat Intelligence. Available online: https://www.ibm.com/security/xforce (accessed on 15 October 2020)
24. Symantec Security Center. Available online: https://www.broadcom.com/support/security-center (accessed on 15 October 2020)
25. Microsoft Security Response Center. Available online: https://www.microsoft.com/en-us/msrc?rtc=1 (accessed on 15 October 2020)
26. Redhat Product Security Center. Available online: https://access.redhat.com/security (accessed on 15 October 2020)
27. Mozilla Foundation Security Advisories. Available online: https://www.mozilla.org/en-US/security/advisories/ (accessed on 15 October 2020)
28. Secunia Research. Available online: http://secunia.com/advisories/historic/ (accessed on 15 October 2020)
29. Liu, Q.; Zhang, Y.; Kong, Y.; Wu, Q. Improving VRSS-based vulnerability prioritization using analytic hierarchy process. *J. Syst. Softw.* **2012**, *85*, 1699–1708, [CrossRef]
30. Google. Severity Guidelines for Security Issues. Available online: http://dev.chromium.org/developers/severity-guidelines (accessed on 15 October 2020)
31. Mell, K.P.; Scarfone, S.; Romanosky, T. Common Vulnerability Scoring System. *IEEE Secur. Privacy. J. Abbr.* **2006**, *4*, 456–461. [CrossRef]
32. Common Vulnerability Scoring System v3.1: Specification Document. Available online: https://www.first.org/cvss/v3.1/specification-document (accessed on 7 May 2020).
33. Common Vulnerability Scoring System v2.0: Specification Document. Available online: https://www.first.org/cvss/v2/guide (accessed on 7 May 2020).

34. Trevor, J. *Enterprise Vulnerability Management*; ISACA Journal 2017. Available online: https://www.isaca.org/resources/isaca-journal/issues/2017/volume-2/enterprise-vulnerability-management (accessed on 8 May 2020).
35. Nyanchama, M. Enterprise Vulnerability Management and Its Role in Information Security Management. *Inf. Syst. Secur.* **2005**, *14*, 29–56. [CrossRef]
36. Skaggs, B.; Blackburn, B.; Manes, G.; Shenoi, S. Network vulnerability analysis. In Proceedings of the 2002 45th Midwest Symposium on Circuits and Systems, Tulsa, OK, USA, 4–7 August 2002; p. III-493.
37. Farris, K.A.; Shah, A.; Cybenko, G.; Ganesan, R.; Jajodia, S. Vulcon: A System for Vulnerability Prioritization, Mitigation, and Management. *ACM Trans. Priv. Secur.* **2018**, *21*, 1–28. [CrossRef]
38. NIST. *Guide for Conducting Risk Assessments*; NIST Special Publication 800-30 Revision 1; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2012; p. 1.
39. Walkowski, M.; Biskup, M.; Szewczyk, A.; Oko, J.; Sujecki, S. Container Based Analysis Tool for Vulnerability Prioritization in Cyber Security Systems. In Proceedings of the 2019 21st International Conference on Transparent Optical Networks (ICTON), Angers, France, 9–13 July 2019; pp. 1–4.
40. Barrett, M.P. *Framework for Improving Critical Infrastructure Cybersecurity*; National Institute of Standards and Technology: Gaithersburg, MD, USA, 2018
41. Allodi, L. Risk-Based Vulnerability Management Exploiting the Economic Nature of the Attacker to Build Sound and Measurable Vulnerability Mitigation Strategies. Ph.D. Thesis, University of Trento, Trento, Italy, 2015; p. 8
42. Fruhwirth, C.; Mannisto, T. Improving CVSS-based vulnerability prioritization and response with context information. In Proceedings of the 2009 3rd International Symposium on Empirical Software Engineering and Measurement, Lake Buena Vista, FL, USA, 15–16 October 2009; pp. 535–544
43. Ali, A.; Zavarsky, P.; Lindskog, D.; Ruhl, R. A software application to analyze the effects of temporal and environmental metrics on overall CVSS v2 score. In Proceedings of the 2011 World Congress on Internet Security (WorldCIS-2011), London, UK, 21–23 February 2011; pp. 109–113.
44. Chen, Y. Stakeholder Value Driven Threat Modeling for Off The Shelf Based Systems. In Proceedings of the International Conference on Software Engineering, Washington, DC, USA, 6–8 November 2007; pp. 91–92.
45. Eschelbeck, G. The Laws of Vulnerabilities: Which security vulnerabilities really matter? *Inf. Secur. Tech. Rep.* **2005**, *10*, 213–219. [CrossRef]
46. Lai, Y.; Hsia, P. Using the vulnerability information of computer systems to improve the network security. *Comput. Commun.* **2007**, *30*, 2032–2047. [CrossRef]
47. Rieke, R. Modelling and Analysing Network Security Policies in a Given Vulnerability Setting. In Proceedings of the Critical Information Infrastructures Security, Samos Island, Greece, 31 August–1 September 2006; pp. 67–78.
48. Gartner Peer Insights 'Voice of the Customer': Vulnerability Assessment. Available online: https://www.gartner.com/doc/reprints?id=1-1Z87ZU8K&ct=200611&st=sb (accessed on 15 October 2020)
49. Yadav, G.; Paul, K. PatchRank: Ordering updates for SCADA systems. In Proceedings of the 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 10–13 September 2019; pp. 110–117.
50. Miura-Ko, R.A.; Bambos, N. SecureRank: A Risk-Based Vulnerability Management Scheme for Computing Infrastructures. In Proceedings of the 2007 IEEE International Conference on Communications, Glasgow, UK, 24–28 June 2007; pp. 1455–1460.
51. Chen, H.; Liu, J.; Liu, R.; Park, N.; Subrahmanian, V. VEST: A System for Vulnerability Exploit Scoring & Timing. In Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, Macao, China, 10–16 August 2019; pp. 6503–6505.
52. El Arass, M.; Tikito, I.; Souissi, N. Data lifecycles analysis: Towards intelligent cycle. In Proceedings of the 2017 Intelligent Systems and Computer Vision (ISCV), Fez, Morocco, 17–19 April 2017.
53. El Alaoui, I.; Youssef, G. Network Security Strategies in Big Data Context. *Procedia Comput. Sci.* **2020**, *175*, 730–736. [CrossRef]
54. Docker Home Page. Available online: http://www.docker.com (accessed on 24 April 2020).
55. Elasticsearch Home Page. Available online: http://www.elastic.co/elasticsearch/ (accessed on 24 April 2020).
56. Kibana Home Page. Available online: http://www.elastic.co/kibana (accessed on 24 April 2020).

57. National Vulnerability Database. Available online: http://nvd.nist.gov/ (accessed on 24 April 2020).
58. Exploit Database. Available online: http://www.exploit-db.com/ (accessed on 24 April 2020).
59. Baron, A. Configuration Mmanagement Database State Model. U.S. Patent No. 7,756,828, 13 July 2010.
60. Nessus Home Page. Available online: https://www.tenable.com/products/nessus (accessed on 24 April 2020).
61. OpenVas Scanner Home Page. Available online: https://www.openvas.org/ (accessed on 24 April 2020).
62. A Universally Unique IDentifier (UUID). Available online: http://www.ietf.org/rfc/rfc4122.txt (accessed on 24 April 2020).
63. Elasticsearch DSL. Available online: https://elasticsearch-dsl.readthedocs.io/en/latest/ (accessed on 18 May 2020).
64. Microsoft Azure Free Tier. Available online: https://azure.microsoft.com/free/ (accessed on 18 May 2020).
65. Azure Network Round Trip Latency Statistics. Available online: https://docs.microsoft.com/en-us/azure/networking/azure-network-latency (accessed on 18 May 2020).
66. What Is Kubernetes. Available online: https://kubernetes.io/pl/docs/concepts/overview/what-is-kubernetes (accessed on 18 May 2020).
67. Peng, C.; Kim, M.; Zhang, Z.; Lei, H. VDN: Virtual machine image distribution network for cloud data centers. In Proceedings of the 2012 Proceedings IEEE INFOCOM, Orlando, FL, USA, 25–30 March 2012; pp. 181–189.