Information Security

Vulnerability Prioritization

Ranking Vulnerabilities by their Environment

Diploma Thesis

For attainment of the academic degree of

Diplom-Ingenieur/in

submitted by

Alexander Halbarth, BSc

is191810

in the

University Course Information Security at St. Pölten University of Applied Sciences

The interior of this work has been composed in LaTeX.

Supervision

Advisor: FH-Prof. Mag. Dr. Simon Tjoa

Assistance:

St. Pölten, September 19, 2021 _____ _____

(Signature author) (Signature advisor)

Ehrenwörtliche Erklärung

Ich versichere, dass

- ich diese Diplomarbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich sonst keiner unerlaubten Hilfe bedient habe.
- ich dieses Diplomarbeitsthema bisher weder im Inland noch im Ausland einem Begutachter/einer Begutachterin zur Beurteilung oder in irgendeiner Form als Prüfungsarbeit vorgelegt habe.
- diese Arbeit mit der vom Begutachter/von der Begutachterin beurteilten Arbeit übereinstimmt.

Der Studierende/Absolvent räumt der FH St. Pölten das Recht ein, die Diplomarbeit für Lehre- und Forschungstätigkeiten zu verwenden und damit zu werben (z.B. bei der Projektevernissage, in Publikationen, auf der Homepage), wobei der Absolvent als Urheber zu nennen ist. Jegliche kommerzielle Verwertung/Nutzung bedarf einer weiteren Vereinbarung zwischen dem Studierenden/Absolventen und der FH St. Pölten.

Ort, Datum *Unterschrift*

Kurzfassung

Schwachstellen in verschiedenen Systemen und Software haben in den letzten Monaten viele Schlagzeilen verursacht und einige der größten Cybersecurity-Vorfälle sind von diesen hervorgerufen worden. Schwachstellenmanagement versucht Probleme mit diesen zu verhindern, indem Schwachstellen effizient geschlossen werden. Um diese Effizienz beizubehalten, obwohl die Anzahl der Systeme in Unternehmen und die Anzahl an Schwachstellen jedes Jahr steigen, sind Priorisierungsmethoden nötig. Der derzeitige Standard für die Bewertung, den Vergleich und die Priorisierung von Schwachstellen ist das Common Vulnerability Scoring System (CVSS). In der üblichen Verwendung dieses Systems fehlen jedoch wichtige Aspekte von Schwachstellen in Unternehmen, da es nur den Schweregrad der Auswirkung der technischen Schwachstelle misst. Die Inklusion der Informationen der Umwelt einer Schwachstelle in die Bewertung kann derzeit nur manuell für jede Schwachstelleninstanz durchgeführt werden, was einen seltenen Einsatz dieser Möglichkeiten verursacht.

Durch eine Analyse der Anforderungen von Schwachstellenpriorisierung in Unternehmen, basierend auf den Problemstellungen im Schwachstellenmanagement und der darin eingesetzten Komponenten, werden Priorisierungsmethoden verglichen. Außerdem werden Datenquellen identifiziert und auf die Verwendung für die Ableitung der CVSS Environmental Metrics und die Automatisierung dieser, analysiert. Durch das Definieren von Eigenschaften von Systemen mit Schwachstellen, ist es möglich diese Kontextinformationen abzuleiten. Diese Eigenschaften werden verwendet, um ein Regelwerk zu entwickeln, welches automatisch die nötigen Metriken von Daten in Schwachstellenscannern ableiten kann.

Das Modell, basierend auf diesen offen verfügbaren Regeln, besteht aus mehreren Schritten mit einer automatisierten Basis, welche einfach an Unternehmensanforderungen angepasst werden kann. Indem Priorisierungsmethoden zum resultierenden CVSS Environmental Score hinzugefügt werden, kann die Schwachstellenpriorisierung Limitierungen von auswirkungsbasierenden Bewertungen umgehen. Die Evaluierung des automatischen Modells zeigt eine erhöhte Diversität der resultierenden Bewertung im Vergleich zu den zugrundeliegenden Ergebnissen. Die Aufgabe der Schwachstellenpriorisierung wird mit den vorgestellten Methoden vereinfacht, während Kontextinformationen und unternehmensspezifische Anforderungen inkludiert werden. Die Kombination mit anderen Bewertungen verbessert in weiterer Folge die Priorisierung, außerdem vereinfacht die Offenheit des Modells die Interpretation und Argumentation von Ergebnissen.

Abstract

Vulnerabilities in different systems and software caused many headlines in the last months. Some of the biggest, in terms of scale, cybersecurity incidents originated from known vulnerabilities in common systems. Vulnerability management tries to stay ahead of problems affecting the cybersecurity of organizations by eliminating these vulnerabilities in an efficient matter.

To stay efficient, even though the number of systems in organizations and the number of vulnerabilities rise every year, prioritization methods are required in vulnerability management. The current standard in scoring, comparing and prioritizing vulnerabilities is the Common Vulnerability Scoring System (CVSS), but its common practice lacks some important aspects of vulnerabilities in organizations, since it is a measure of the severity of a technical vulnerability only. Incorporating the environmental information surrounding a technical vulnerability into this rating is currently only possible manually on a per-instance basis, causing them to not be employed often.

By analyzing the requirements for vulnerability prioritization in organizations, based on challenges involved in vulnerability management and the used components in this process, methods for vulnerability prioritization are compared. Furthermore, data sources are identified and it is examined which can be used to infer the CVSS Environmental Metrics and automate the prioritization task. By extracting features from systems, on which vulnerabilities reside, context is being created. These features are used to develop rules that can automatically suggest these metrics from data in vulnerability scanners.

The model, based upon these open rules, consists of multiple steps to provide an automatic base, which can be customized to unique needs of organizations with little effort. By providing prioritization methods that can be added to the resulting CVSS Environmental Score of the model, vulnerability prioritization can be taken beyond known limitations of severity-based scoring. The evaluation of the automatic model shows an increased diversity in the scores as opposed to the underlying technical vulnerability rating. The suggested customizations, that organizations can incorporate into the model, would further enhance these results. The task of prioritization of vulnerabilities is simplified with the proposed methods, while incorporating context of vulnerabilities and organization specific requirements. By combination with other leading scoring systems, the model can be further enhanced and its openness makes interpretation of the results and argumentation on the actions derived from the model easy for its users.

Contents

1 Introduction

Due to the increasing risk of cyber-attacks, which has been seen lately in the recent attacks on Microsoft Exchange Servers[1], it is important to quickly close attack vectors which are related to software vulnerabilities and misconfigurations to prevent further damage. To efficiently close these vulnerabilities in organizations' systems, it is important to work on them in an organized manner. This requires implementing a vulnerability management process and is often accompanied by using technology like vulnerability scanners. Implementing such a vulnerability management process into existing infrastructures composes many challenges such as described in [2].

One of these challenges is the difficult task of prioritization of vulnerabilities despite having limited resources to remediate them. Since the number of vulnerabilities is growing faster every year, this task is getting more important every day. (See figure 1.1 with data from cvedetails.com[3] for reference)

A scoring system to aid prioritization, should be easy to interpret and be flexible in customization to fit the organization. A good acceptance for such a model ca be achieved, by having a low entry threshold. To use the resources efficiently, such a system should be mostly automated and should only need a small amount of manual input data, to quickly customize the outcome.

There already exist scoring systems for technical details of vulnerabilities, which offer a first level of prioritization on a per vulnerability basis. But if the same vulnerability is present on different systems in an environment, it is possible that it has a completely different business impact, depending on the systems. Therefore, these instances of the same technical vulnerability must not be ranked equally, to use the limited resources of Information Technology (IT) personnel in an efficient matter. This discrepancy of the technical impact of a vulnerability and the business impact needs to be addressed to lower the time spent at prioritization.

The Common Vulnerability Scoring System (CVSS)[4] as defined by Forum of Incident Response and Security Teams Inc. (FIRST), a US-based non-profit organization, is a standard in quantifying the technical characteristics of a vulnerability. The CVSS is described thoroughly in section 2.3. It incorporates three numerical scores from zero to ten. These three scores compose a hierarchical scoring system where the Base

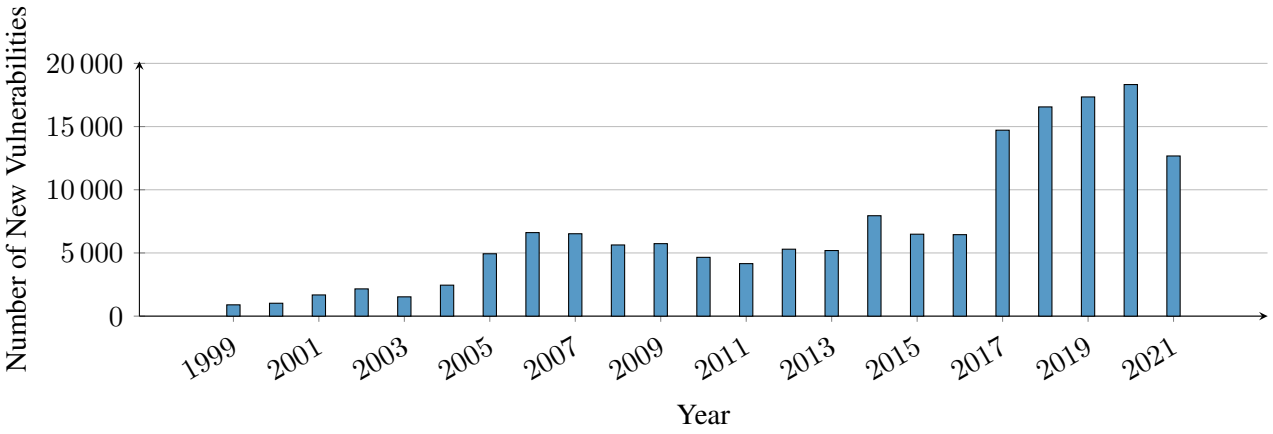


Figure 1.1: Number of new Vulnerabilities per Year retrieved on August 26, 2021 [3]

Score contains the least information, the impact and exploitability of a vulnerability. The Temporal Score contains information that changes over time and the Environmental Score can be used to alter the rating to represent the actual instance of the vulnerability on a particular system. Since the Environmental Metrics of CVSS currently need to be manually defined at least on a system level, it is not used very often [5] and can also be error prone as concluded in[6].

To aid the prioritization of remediation actions to vulnerabilities on a per-system basis, a quantified scoring can be used. CVSS is able to create such a scoring, but it lacks the possibility of automatically scoring a huge number of systems. Organizations performing vulnerability management are overwhelmed by the amount of work that is required to close their vulnerabilities. In fact, the Cyentia Institute shows, based on data from 300 organizations, “half of firms falling behind, one in six keeping up, and one-third gaining ground”[7, p. 20], when comparing the number of vulnerabilities closed and the number of high-risk vulnerabilities open. They most likely do not have the time to rate each vulnerability on each system manually, to infer a prioritization using the CVSS Environment Score, when they could use their time to actually remediate or mitigate vulnerabilities. On the other hand, organizations which already employ vulnerability management and need to comply to different standards and norms, can be assisted by giving them tools, to adapt the vulnerability ranking to their environment and to the measurements they have already taken to minimize risks after vulnerabilities have been exploited, like segmentation. By adapting the rating to the risk minimizing measurements of an organization, risk acceptance of vulnerabilities can be described more easily and resources can be assigned to vulnerabilities that actually impose the biggest risk to the organization.

The described challenges in vulnerability management and prioritization of remediation of vulnerabilities lead to the following research questions of this thesis.

- Which data is required to estimate the Environmental Score of the CVSS?
- What data sources can be used to acquire this data and which of them can be deduced from available data in vulnerability scanners?
- How is it possible to design a model that infers the Environmental Score of CVSS, that combines automatic and manual data.

The major contribution of this thesis will be a definition of an iterative model to rank vulnerabilities in existing environments. The iterative way of implementing this system is important, to be able to incorporate this strategy to organizations with vastly differing maturity in security. The input data to this system will start with data that is automatically collected by vulnerability scanners, which is not unique to the organization like Operating System (OS). In higher levels it should incorporate organization specific data to improve on the initial ranking, like input from Information Security Management System (ISMS) documentation. By having a flexible model, that can be used with or without manual interaction, organizations with very little maturity in security and therefore also fewer human resources dedicated to security, can use the model. Organizations with better capabilities, with dedicated personnel for security tasks, can input information into the model and improve the results even further.

The already existing standard in the industry of quantifying vulnerabilities, CVSS, will be improved by the proposed model which uses the CVSS Environmental Score to easily ingest the system ranking into the resulting CVSS total vulnerability score. By inferring this part of the score, ranking of vulnerabilities on a risk-based level is made available very easily. When incorporating more information to the model, like risk-reducing measurements and information on systems and networks, the model should improve even further and converge with the real risk imposed by vulnerabilities. To further prioritize vulnerabilities and factor in the risk that they impose on systems in an organization, the proposed framework should improve upon different existing ideas and contributions as described in chapter 3. By combining different aspects that can be used to increase the granularity of vulnerability scores and staging them, the model should be usable for organizations in different maturity levels of their security system.

The scope of the thesis is defining the different stages in the model, assessing input data for them and providing a means to calculate the resulting score and prioritization on a granular basis. Furthermore, the elements used in the model will be documented, to complement them with additional information which is not added in this work. Different contributions of related work in chapter 3 will be used to aid the different approaches of the model and supplement the stages with insight.

1.1 Thesis Outline

This thesis is structured into the following parts:

Chapter 2 is defining important terms in Vulnerability Management (VM) in section 2.1.1, as well as explaining how VM can be performed in organizations based on standards in section 2.1.2. Important specifications used for vulnerabilities are explained in the following sections of the chapter. The Security Content Automation Protocol (SCAP), and its various definitions for VM are explained in section 2.2. The Common Vulnerability Scoring System (CVSS) is discussed and explained thoroughly in its current version 3.1 in section 2.3. In the last section of the chapter other scoring systems for vulnerabilities used in the industry are touched upon in section 2.4, with a summary and comparison of them in section 2.4.7.

Chapter 3 is discussing different related work based on, criticizing or improving CVSS and other vulnerability prioritization methods and rankings. Section 3.4 concludes the chapter with an overview on topics covered by related work and a brief discussion about requirements, that can be deducted from related work for this thesis. Furthermore, a comparison between the presented prioritization methods and rankings is performed.

Chapter 4 is defining the approach used to create the model for vulnerability prioritization, which is the main contribution of this thesis. The requirements for the model are discussed in section 4.1, which are used in the following creation of the model.

Chapter 5 explains how the input data for the model is retrieved, analyzed and processed, according to the requirements described in the approach chapter. Two different depths of classifications for systems, that can be retrieved automatically from vulnerability scanners are discussed in detail. The first classification is Operating System and Device Type in section 5.1, followed by Services, Applications and Roles in section 5.2. The chapter is concluded by a final listing of classifications for the implementation of the models prototype in section 5.3.

Chapter 6 contains the main contribution of this thesis. The three steps of the model are explained, which is able to infer the CVSS Version 3.1 (CVSSv3.1) Environmental Metrics using a staged approach. The first automatic step is described in section 6.1 and uses only data from vulnerability scanners. The second step in section 6.2 improves upon the first by adding organizational specific customizations to the assignments in the first step. The third step adds custom rules into the model, depending on the organizations security system and is described in section 6.3. To create a prioritization of vulnerabilities that goes beyond the limitations of CVSS, prioritization using exploitation is added to the result in section 6.4 and other prioritization methods are added to the inferred Environmental Score in section 6.5.

Chapter 7 describes an evaluation of the automatically inferred CVSS Environmental Score and other prioritization methods on a real dataset, to illustrate the improvements made with the added scoring from the model.
Chapter 8 draws a conclusion to the thesis and discusses the results created with the model. The thesis closes with an overview on future work and possible improvements to the created model in section 8.1.

2 Prerequisites

This chapter contains a brief introduction to VM in section 2.1. In this section the terms used in VM are defined in section 2.1.1. An example process for VM, that is used to implement the results of this thesis, is presented in section 2.1.2 as well as general VM practices in organizations. In following sections different standardized constructs that are commonly used in VM are described. SCAP and its accompanying specifications are described in section 2.2. Specifications used in SCAP, which are used in this thesis are described further in the subsections. These are Common Weakness Enumeration (CWE) in section 2.2.1, Common Platform Enumeration (CPE) in section 2.2.2, Software Identification Tag (SWID) in section 2.2.3 and Common Vulnerabilities and Exposures (CVE) in section 2.2.4. The CVSS is described thoroughly in section 2.3, since it is detrimental to this thesis work. Finally, the end of this chapter in section 2.4 describes other scoring systems, that are used in addition to CVSS in the VM industry, like the Exploit Prediction Scoring System (EPSS) and VM vendor scores, as well as scores employed by big software companies for their advisories.

2.1 Vulnerability Management

Vulnerability Management (VM) is an inevitable part of securing IT systems. Due to the fast pace in which vulnerabilities are discovered in modern systems, as shown in figure 1.1, it is crucial to remediate them to keep systems secure. Defined in *Practical vulnerability management*, "Vulnerability management is the practice of staying aware of known vulnerabilities in an environment and then resolving or mitigating these vulnerabilities to improve the environment's overall security posture."[8, p. 4] The easiest model of VM involves gathering vulnerability information about a system and remediating them in a second step. The gathering of the information can be done with a vulnerability scanner and the remediation is mostly done manually. Due to challenges in this process, prioritization and scoring of vulnerabilities is required, since not all vulnerabilities can be remediated in practice[7, p. 20].

This section of the thesis will define important terms used in this thesis from VM in section 2.1.1. It will then go into detail on VM in organizations as well as a possible process for VM in section 2.1.2.

2.1.1 Definitions

The term **weakness**, required for CWE and the definition of vulnerabilities, is defined by MITRE [1] as "flaws, faults, bugs, or other errors in software or hardware implementation, code, design, or architecture that if left unaddressed **could** *[emph. added]* result in systems, networks, or hardware being vulnerable to attack"[10]. **Vulnerabilities** in contrast are defined as "A flaw in a software, firmware, hardware, or service component resulting from a weakness that **can be exploited, causing a negative impact** *[emph. added]* to the confidentiality, integrity, or availability of an impacted component or components"[11]. The important difference is that each vulnerability is based on a weakness, but not every weakness is a vulnerability since it needs to be exploitable and a negative impact must be the result of this exploitation. A **vulnerability instance** is the actual vulnerability on a given system. In this thesis, different vulnerabilities are two distinct vulnerabilities, each with their own characteristics. Different vulnerability instances on the other hand can be the same vulnerability, but they are present on different systems or different services on the same systems.

The act of **exploitation** is the creation of an impact to a vulnerable component by making use of a vulnerability. A requirement for exploitation can be an **exploit**. "An exploit is the specially crafted code adversaries use to take advantage of a certain vulnerability and compromise a resource."[12] Some vulnerabilities do need an exploit for exploitation, others only need special knowledge on the vulnerability and can be exploited without special software.

A **vulnerability scanner** is a tool used to identify vulnerabilities on devices, by interacting with them[8, p. 6] or observing its behavior. This can be either via **network**, **authenticated** or **passive** scans. "Network scanners reach out to every Internet Protocol (IP) address within a range, or a specific list of IPs, to determine which ports are open, which services are running on those ports, the operating system (OS) versions and relevant configurations, and software packages running on each device."[8, p. 6] Authenticated scans use credentials to login to systems and perform the identification of vulnerabilities locally within the context of the credentials supplied. Therefore, these scans results can be more accurate and contain vulnerabilities which cannot be identified via the network[13], [14]. Authenticated scans can target specific services like web-applications or the OS as a whole, via a management interface like Secure Shell (SSH) or Windows Management Instrumentation (WMI). Alternatively, authenticated scans can be performed with special soft-

[1] *MITRE* Corporation is a private, not-for-profit organization, which is dedicated to "bring innovative ideas into existence in areas as varied as artificial intelligence, intuitive data science, quantum information science, health informatics, space security, policy and economic expertise, trustworthy autonomy, cyber threat sharing, and cyber resilience."[9] MITRE is an important organization, which is creating standards for VM and sharing vulnerability information. They created and own the CWE (section 2.2.1) and CVE (section 2.2.4) trademarks and work together with National Institute of Standards and Technology of the US Department of Commerce (NIST) to address cybersecurity challenges.

ware installed on each device, called agents [8, p. 6]. Passive scans are conducted by performing network sniffing, which is analyzing the network traffic of particular parts of a network. By analyzing the traffic, used applications, their versions and OSs as well as devices can be identified[15, p. 3-4]. This technique is often employed in Industrial Control Systems (ICS), because it does not require interaction with potentially fragile systems[16, p. 3]. Scanning for vulnerabilities is similar to patch management systems, described in [17], since both systems need to identify software and their versions installed on various systems.

2.1.2 Vulnerability Management in Organizations

To fulfill organization's goals, governance principles are employed on an organizations IT. To create this governance system custom tailored to an organization, COBIT can be used. "COBIT is a framework for the governance and management of enterprise information and technology, aimed at the whole enterprise."[18, p. 13] Its core structure and components are described in [18]. COBIT defines 40 core governance and management objectives, that can be used to create a tailored enterprise governance System for Information and Technology from design factors like enterprise strategy, goals and size as well as compliance, ITs role and sourcing model of the IT. For each of the 40 core governance and management objectives, COBIT is referencing other standards, frameworks and regulations to assist implementation. The core governance and management objectives of COBIT are described in detail in [19]. Each objective is composed using the seven governance components. The components of the objectives of COBIT are[18, pp. 21]:

- **Processes**, as the most familiar component of governance systems, defines how to fulfill the objective using a set of practices, which in turn have a set of activities.
- **Organizational structures** define which roles in an organization are responsible or accountable for a certain practice, derived from the process.
- **Principles, policies and procedures** are linked to the processes to create guidance for using them and keeping them up to date.
- **Information** defines inputs and outputs of the process and therefore connects processes with each other.
- **Culture, ethics and behavior** are used to help achieve the objectives, by defining desired elements in the organization.
- **People, skills and competencies** are the requirements to fulfill the processes and practices successfully.
- **Services, infrastructure and applications** defines the technology required to fulfill the governance objectives.

VM is implemented in COBIT in the *Managed Security Services* management objective (DSS05)[19, pp. 257-263]. Its defined purpose is to "Minimize the business impact of operational information security vulnerabilities and incidents"[19, p. 257]. The management practice, *Manage vulnerabilities and monitor the infrastructure for security-related events* (DSS05.07) suggests using tools and technologies, managing vulnerabilities and monitoring the infrastructure for unauthorized access[19, p. 261]. It has to be furthermore ensured to integrate into the general event monitoring and incident management of the organization. The activities in this practice, that are important in a VM program are activity 1 "Continually use a portfolio of supported technologies, services and assets (e.g., vulnerability scanners, fuzzers and sniffers, protocol analyzers) to identify information security vulnerabilities."[19, p. 261] and activity 2 "Define and communicate risk scenarios, so they can be easily recognized, and the likelihood and impact understood."[19, p. 261]. The referenced National Institute of Standards and Technology of the US Department of Commerce (NIST) publication [20], defines VM in the *Vulnerability Monitoring and Scanning* control (RA-5). The publication is a catalog of security and privacy controls for information systems and organizations. For further reference, to implement VM in an organization, [17] is used. The current version of this document, version 3[17], goes into detail of modern patch management and software inventory and does not cover VM in detail, in fact it defines vulnerability scanning as out of scope of the document[17, p. 10]. Instead, the publication references to version 2[21] of the same document[17, p. 1].

The guidance to create a VM program in [21] contains a VM process with multiple steps, which should be employed by a group of people in the organization. It goes over creating a system inventory utilizing existing inventories of the organization. On the systems within this inventory, monitoring for vulnerabilities, remediations and threats should be employed. Sources for these vulnerabilities are vulnerability scanners as an automatic source and vulnerability databases, vendor information, and third party information sources as manual sources.

Since the publication was created in Nov. 2005, vulnerability scanners capabilities have improved, but the tasks in the process are still valid. A common practice is to identify vulnerabilities in an organizations IT environment with a vulnerability scanner, which contains definitions for well known vulnerabilities of standard software (most likely based on CVE[22]). After running a vulnerability scan on the environment of the organization, a number of vulnerability instances are identified. These get reviewed and prioritized by different metrics. The de-facto standard that is used today to prioritize vulnerabilities based on their impact and severity is CVSS, more specifically the CVSS Base Score[23, p. 20:4].

After identification, the remediation of vulnerabilities needs to be prioritized. [21] is suggesting not to overwhelm administrators with too many remediations and suggest setting priorities for vulnerability patching.

The publication suggests basing the prioritization on significance of the threat or vulnerability, the existence of exploits and the impact of the exploitation. Furthermore, the risk of the patch should be included into this prioritization too. To create this prioritization information is gathered, which is how the vulnerability can be remediated and if that remediation could impact required functionality. In this analysis the possible outcomes are, that a vulnerability can be fixed without any problems and that a vulnerability cannot be fixed due to missing patches or other requirements which prevent the patching.[21] A third option, mentioned in [8, p. 34], is to accept the risk and impact associated with the vulnerability and not apply the patch. If a risk cannot be accepted and patches cannot be applied, due to unavailability or requirements, risk-reducing measurements must be taken to reduce the risk to an acceptable level. These measurements can be additional security measurements that make exploitation of the vulnerability harder or workarounds to temporarily remediate vulnerabilities.

The information on which patches need to be applied, should be maintained in a database to perform the remediation. To remediate the vulnerabilities classical change management concepts are performed, which test the remediations and then deploys them to affected systems either directly, via the systems local administrators or via automated deployment tools. After applying the patch, the remediation should be verified, a rescan with the vulnerability scanner can be done to check if the vulnerability was actually closed and if the change did create new vulnerabilities. Alternatively, manual examinations can be done to verify the correct closure of the vulnerability or exploits could be attempted.[21]

The VM process composed of vulnerability identification, prioritization, remediation and verifying should be repeated regularly to find new vulnerabilities and to assess systems that could not be assessed in prior scans. As shown in figure 1.1 vulnerabilities are getting published more often and new vulnerabilities can be identified daily. Furthermore, changes in systems can always create vulnerabilities or recreate vulnerable situations remediated before, therefore the vulnerability management process needs to be conducted in a cycle.

More information on how to build VM programs and processes in organizations can be found in [2], [8], [19]–[21]. This section gave a shortened overview of the concepts told in these sources for reference of the concepts required in the scope of this thesis.

2.2 Security Content Automation Protocol - SCAP

The Security Content Automation Protocol (SCAP) is a framework to provide interoperability between security applications by combining specifications to exchange content for security, vulnerability and compliance

checks. It is derived from community ideas, maintained by NIST and was created in 2006. It contains specifications to exchange content describing, identifying and generalizing assets, vulnerabilities, configurations and components of IT. The components can be for example software, hardware, applications and OS. Assets, as defined per the Asset Identification (AID), can incorporate "anything that has value to an organization"[24, p. 1], which includes more than just IT components. The current specifications to SCAP is version 1.3[25], released in February 2018, with version 2[26] currently in development.

SCAP uses multiple components which have a widespread use in VM. In addition to the components relevant for this thesis described in the next sections (Common Vulnerabilities and Exposures (CVE), Common Weakness Enumeration (CWE), Common Platform Enumeration (CPE), Software Identification Tag (SWID)). SCAP and its accompanying specifications also provide definitions for checklists (Extensible Configuration Checklist Description Format (XCCDF), Open Checklist Interactive Language (OCIL)), common configurations (Common Configuration Enumeration (CCE), Common Configuration Scoring System (CCSS)), assets (Asset Identification (AID), Asset Reporting Format (ARF)) and the definition of Open Vulnerability and Assessment Language (OVAL), "a language for representing system configuration information, assessing machine state, and reporting assessment results"[25, p. 4]. In addition to VM, security checklists or benchmarks, that can be used to harden systems to lessen the impact of vulnerabilities, are another usecase of SCAP[27]. These checklists are exchanged using the National Checklist Program (NCP)[2]. The full list of contained specifications in SCAP can be derived from the official project website [28] and the specifications of the versions of SCAP [25], [26].

SCAP furthermore certifies products that incorporated their specifications. These certifications can be seen on the official project homepage[3]. At the time of writing only 20 products are certified, which is also a shortcoming described in [29]. Another problem with SCAP, described by MITRE in [30], is that the incorporated standards are not fully utilized by SCAP. [31] describes how SCAP can be used and explains a simple usecase on how to use SCAP and its accompanying specifications.

Version 2 of SCAP will try to overcome many of its shortcomings as described in [26]. Vulnerability Assessment is one of the proposed usecases of the new interoperability features of version 2. It should enable the continuous collection of software inventory data by using SWID and therefore eliminate the need to rescan endpoints to refresh collected data, by incorporating a push model. Furthermore, the data can be used for different usecases (e.g. license management) and does not need to be collected multiple times for use in multiple tools.

[2] https://ncp.nist.gov/repository

[3] https://csrc.nist.gov/projects/scap-validation-program/validated-products-and-modules

2.2.1 Common Weakness Enumeration - CWE

Common Weakness Enumeration (CWE), a trademark by MITRE Corporation, tries to catalog different generalized weaknesses that can be present in IT components. The definition of weaknesses used by MITRE can be found in section 2.1.1. The difference to CVEs is that a general weakness can cause a vulnerability but does not have to. CWEs are used to describe common weaknesses and help to prevent security vulnerabilities in software.

The target of CWE lies more in development as in securing exiting environments by fixing vulnerabilities like with CVE. The frameworks built around CWE are the Common Weakness Scoring System (CWSS) and the Common Weakness Risk Analysis Framework (CWRAF), both also trademarks by MITRE Corporation. CWSS, as defined per [32] is very similar to CVSS, it serves a similar purpose as CVSS serves, but it is based on weaknesses and CWE instead of vulnerabilities and CVE like CVSS. It tries to prioritize different weakness discovered in software, by assigning a quantitative measurement. Like CVSS it is split into different metric groups. They are "Base Finding", "Attack Surface" and "Environmental". The main difference between CVSS and CWSS is explained in Appendix A of the specification[32]. As per the analysis by MITRE of different usage scenarios of CVSS Version 2 (CVSSv2) and CWSS, CWSS can be used before vulnerabilities have been discovered and it has built-in support for incomplete information. The CWRAF includes mechanisms linked to weaknesses, which measure risk to an organization[33].

Improvements to the CWSS and CWRAF are not in the scope of this thesis. Since CWSS also incorporates environmental factors for the scoring it provides, a future work can be transferring this works result to CWSS.

2.2.2 Common Platform Enumeration - CPE

The Common Platform Enumeration (CPE) is defined by NIST in four publications, which build upon each other. The current specification is version 2.3 published in August 2011[34]–[37]. The CPE is used to identify (classes of) software (or as named by the specification applications), OS and hardware[34, p. 1]. The standardized identification of these so-called IT classes is important for IT management to be able to make decisions based on the inventory of them. The probably most popular usecase of CPE names and the possibilities of logically comparing them to each other is VM. The modular and set-based approach of the CPE standard enables easy comparison and therefore selection of one-to-one matches of CPEs, but also the decision if one CPE is a subset of the other or if names do not match and are disjoint. The CPE Naming Specification [34] describes the syntax and semantics of the different formats of CPE names, the different

attributes of the CPE name and how to "bind" and "unbind" them, which is the conversion between the old Uniform Resource Identifier (URI) format specified by version 2.2 of the standard and the new Well-Formed CPE Name (WFN) format introduced with version 2.3 of CPE. Building upon the Naming Specification is the Name Matching Specification [35] which defines how to compare CPE names in the different formats. The CPE Dictionary Specification [36] is the third part of the specification which defines the CPE dictionary. It is a repository of IT product names identifying unique classes of real-world IT products. The official CPE dictionary is hosted by NIST as part of the US National Vulnerability Database (NVD) [38]. By providing an official dictionary, where organizations can submit their own CPE names, naming conflicts between multiple dictionaries should be avoided. Literature shows in analysis of the NVD dictionary, that this central approach also has its downsides[39]. The fourth and last part of the specification is the CPE Applicability Language Specification [37] and enables grouping of CPE names and description of complex IT systems.
In combination the CPE enables the assignment of the different parts of an IT system to standardized, unique names, like the OS and applications running on it via the naming and dictionaries. A vulnerability can be assigned to a group of CPE names via the applicability language and the rules of name matching, to which it applies. With these two data sources it can be automatically decided if a system is vulnerable by comparing the CPE names of the system to the statements of vulnerabilities and deciding if the CPE names of the system are subsets of the CPE names of each vulnerability. If a vulnerability applies to all versions of a particular software, it can be easily modeled with CPE. More complicated statements, like if it only applies to software running on specific hardware or OSs, can be defined too.
By providing this open standard of defining the applicability of vulnerabilities to certain CPE statements, sharing of vulnerability information can be automated. The current standard of SCAP is using CPE for its various usecases like VM. SCAP version 2 will transition to use SWID instead of CPE to overcome shortcomings of CPE, like the inability to identify software patches [26].

2.2.3 Software Identification Tag - SWID

The Software Identification Tag (SWID), as defined by International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) in [40], enables the possibility of unique tagging of software by software developers and third parties. It was first published in November 2009 and revised in 2015. SWID defines a data structure to store information about software, including version and patches, for automated inventory and management. These tags are created by SWID producers, software creators which develop and distribute software or third parties. Third parties can be automated inventory tools and can assign temporary or permanent SWIDs to software, which do not have SWIDs assigned to them. By

including evidence information in automatically assigned tags, it is possible to distinguish versions, without knowing the version schema. Due to the complex data structure of SWID tags, they can include information about patches and requirements and can be digitally signed if required. Complex hierarchies of software and its updates and patches can be built, by using links between the SWIDs. [40]

By implementing an international standard to identify software, different shortcomings of CPE are mitigated. Due to the uniqueness of the identification of organizations, which is implemented via URI and the assignment of *regid*s to organizations based on authority parts, such as domain names, it is possible to create a decentralized system, which fulfills the requirement of having unique identifiers for software. Each SWID has a *tagid*, which is unique in its organization. By comparing *regid*s and *tagid*s it is possible to distinguish different software. Due to the modular datastructure of SWID, it is possible to complement information to SWID tags by creating supplemental SWID tags, even though modification of SWIDs is only allowed for their creators. SWIDs can reference each other to complement each other. [40]

The usage of SWID has been added to the SCAP specification in version 1.3[25]. In version 2 of SCAP[26], there will be a transition from the CPE, as an identification for software to SWID. Since SWID is only able to identify software products, CPE will still be used for hardware identification. NIST has published its own guidelines for the creation of SWIDs[41]. These guidelines describe how to use SWID tags in software asset management and cybersecurity. By using XML Path Language (XPath) Queries, it is possible to reference different versions of software via their SWID tags in Extensible Markup Language (XML) format. The document also suggests different usecases for SWID with accompanying processes, one of them is to identify vulnerable endpoints in section 6.1.3. SWID tags can also be converted to CPE names, as defined in [42].

2.2.4 Common Vulnerabilities and Exposures - CVE

Using the Common Vulnerabilities and Exposures (CVE), a trademark by MITRE Corporation, a list of publicly known cybersecurity vulnerabilities, the CVE List, is maintained. It started in 1999 and the underlying CVE program is sponsored by the U.S. Department of Homeland Security.

The CVE program assigns unique identifiers, the CVE Identifier (CVE ID), to each identified vulnerability in standard software via its currently 179 CVE Numbering Authoritys (CNAs)[43], organizations which are responsible for the assignment and publishing vulnerability information in the CVE Records. By doing that, it became the de facto international standard for identifying vulnerabilities. It enables an easy way to link and search specific vulnerabilities between different data sources by using the CVE ID[22].

Each CVE record consists of a unique CVE ID, a description of the security vulnerability and references.

The CVE ID has the format *CVE-YYYY-NNNN*, where *YYYY* is a placeholder for the year the CVE ID was assigned or the vulnerability was made public (it is not referencing the year the vulnerability has been discovered). The *NNNN* part of the CVE is a number starting from *0001* each year. In 2014 the syntax has been changed from 4 digits to any number of digits, since the 9 999 unique identifiers per year, as defined in 1999, would not be enough for today's number of vulnerabilities.

2.3 Common Vulnerability Scoring System - CVSS

The Common Vulnerability Scoring System (CVSS) is a standard defined by FIRST, a US-based non-profit organization. It tries to quantify the characteristics of software vulnerabilities. The current version of CVSS is CVSSv3.1[4], which was released in June 2019. CVSSv3.1 offers no change in scoring to CVSS Version 3.0 (CVSSv3.0), it only improved guidance on selecting metrics. Therefore, CVSSv3.0, CVSSv3.1 and CVSS Version 3 (CVSSv3) is often used interchangeable in literature.[4] CVSSv3.0 was released in June 2015[44]. The following section is mostly summarizing the specifications of CVSSv3.1[4] with some additional information from other sources and less details. The CVSS is created through the work of the CVSS Special Interest Group (CVSS SIG), which is currently working on the next version of the CVSS standard[45]. Information on the changes and updates to CVSS Version 4.0 (CVSSv4) are described in section 2.3.3.

The release of CVSSv3.1 also included guidance on how to integrate extensions into CVSS and the CVSS Vector. To incorporate this, the CVSS Extensions Framework has been added in Section 3.9 of [46]. The proposed framework can be used to incorporate additional metrics into CVSS, while retaining the existing metrics. Adding more scores is also permitted by the framework, as long the original scores are not modified.

The CVSS creates a score ranging from 0 to 10, with one decimal place, to quantify vulnerabilities characteristics. The three components of the score, Base Score, Temporal Score and Environmental Score, assemble an iterative model to rate vulnerabilities. The components at first rate the vulnerability in general with the Base Score, the optional Temporal Score adds information about the vulnerability which can change over time and the Environmental Score, which is also optional, adds information of the specific environment the vulnerability is present in. The components of the CVSS and the different metrics used in rating are explained in 2.3.1. The calculation of the CVSS is described in 2.3.2. The Temporal Score includes the Base Score Result already, they are not independent scores from each other. The Environmental Score in-

[4] In this thesis CVSSv3.0/CVSSv3.1 is used when the referenced version is explicitly 3.0/3.1 and CVSSv3 if the distinction is not important.

Severity Rating	CVSSv2 Score (NVD)[48]	CVSSv2 Score (Tenable)[49]	CVSSv3 Score[4]
None/Info		0.0	0.0
Low	$0.0 - 3.9$	$0.1 - 3.9$	$0.1 - 3.9$
Medium	$4.0 - 6.9$	$4.0 - 6.9$	$4.0 - 6.9$
High	$7.0 - 10.0$	$7.0 - 9.9$	$7.0 - 8.9$
Critical		10.0	$9.0 - 10.0$

Table 2.1: Association of CVSS scores to qualitative severity ratings

cludes the Base Score Result, by using its metrics values if they are not modified and the Temporal Metrics. Therefore, the calculation of the Environmental Score includes all the metrics and defines the final score for a given vulnerability on a specific system. The numerical score is mapped to a qualitative severity class, by the range as defined by the specification[4]. CVSSv2, which is defined in [47], did not include such a severity rating, but the NVD has a commonly used scheme. Table 2.1 shows the qualitative severity ratings and the CVSS scores associated with them by NVD[48] and Tenable Nessus[49] for CVSSv2 and the severity as per the CVSSv3.1 specification[4].

The motivation behind CVSS is described as to "help organizations appropriately prioritize security vulnerabilities across their constituency"[50]. Before the release of CVSS Version 1 (CVSSv1) in April 2005 many organizations created their own scoring system to score vulnerabilities in their products without coordination, which lead to incomparable ratings. After solving inherent problems of CVSSv1, CVSSv2 was released in June 2007[51]. It solved many of the problems and is still used today, due to the large number of vulnerabilities that have not been rated with CVSSv3. The NVD, operated by NIST, one of the biggest repositories of vulnerability information, converted all CVSSv1 scores in their database to CVSSv2. There is currently no plan to assign CVSSv3 scores to all CVEs prior to December 20, 2015[48]. This means that just 53% of all vulnerabilities in the NVD contain a CVSSv3 score[52][5]. The problem with the gap in CVSSv3 data is often circumvented in literature by just using the CVSSv2 scores[53]. The authors of multiple publications [45], [54], [55], show, that the changes from CVSSv2 to CVSSv3 created problems. Mentioned in [54] the transformation function for old scores does not exist which further complicates VM for organizations. Other sources [56] show, that a conversion should be possible, but requires manual input. The work in [55] tries to mitigate the problem by creating Machine Learning (ML) algorithms to convert the scores to the new version. Commercial products from Tenable[6] published updates to their vulnerability

[5] Retrieved data via statistic query from NVD on July 18, 2021 [52]

[6] `https://www.tenable.com/`

management tools between December 2020 and May 2021, to incorporate the possibility to use CVSSv3 vulnerability scores to measure vulnerability severity[57]. Before that, they used solely CVSSv2 for severity classification, since not every vulnerability has a version 3 score associated. For vulnerabilities that do not have a CVSSv2 score associated with them, they now use CVSSv2 scores instead of their CVSSv3 scores. This is the new default behavior for newly setup Tenable products. They did not implement any conversion or estimation for old vulnerabilities. It is possible for the user of the product to choose whether to use CVSSv2 for all vulnerabilities or the new behavior.
This thesis will focus on the CVSSv3.1 to assess scores. Conversion of CVSSv2 scores is outside the scope of this thesis. To keep as much compatibility with CVSSv4 as possible, the approved changes to CVSS for the upcoming release are examined closely.

2.3.1 Components of CVSS

The CVSS score consists of three parts, the Base Score, the Temporal Score and the Environmental Score. Only the base score is required for scoring vulnerabilities with CVSS, the temporal and environmental score are optional. This section explains the different metrics that are combined into the scores of CVSSv3.1. The scores, metrics and descriptions following are according to the current standard of CVSS, Version 3.1[4].
All metrics that create the score, can be represented as a textual representation, that contains all assignments for a rated vulnerability. This representation is called CVSS vector. Optional parts of the score, which are not assigned can be omitted. The CVSSv3.1 specification demands to always publish the CVSS Score of a vulnerability with its vector to better understand the creation of the score. [4]

CVSS Base Score

The Base Score defines the characteristics of the vulnerability, which are static. It consists of a set of two metric groups, the exploitability metric group and the impact metric group. The exploitability metric group defines how the vulnerability can be exploited and the impact metric group defines what direct consequence the exploit of the vulnerability is causing in a worst case. The consequences are reflected for the component that suffers the impact itself and not the vulnerable component. This was introduced with CVSSv3.0. The scope metric is measuring if the vulnerable component is affecting another component beyond its scope or if the impact metric group is referencing the vulnerable component itself.[4]
The metric abbreviations in the scope of this thesis are prepended with *CVSS-* and then their abbreviations as per the CVSSv3.1 standard[4]. This is done, to differentiate them from other abbreviations in this thesis. The abbreviations of the values of the metrics are not prepended.

The components creating the mandatory part of CVSSv3 and the **Base Score** are listed below.[4]

- Exploitability Metric Group
  - CVSS Attack Vector Metric (CVSS-AV)

    The CVSS Attack Vector Metric defines from where the vulnerability can be exploited. It has the following values.

    * Network (N)

      The vulnerability can be exploited from any network, including the internet (not taking any network segmentation or security measures into account).

    * Adjacent (A)

      The vulnerability can be exploited from a network adjacent to the vulnerable component. The remote exploitability is limited due to the nature of the vulnerable component or protocol or due to the vulnerability itself.

    * Local (L)

      The attack is not bound to the network, it requires access to the target system or it requires user interaction.

    * Physical (P)

      The attack requires physical access to the vulnerable component.
  - CVSS Attack Complexity Metric (CVSS-AC)

    The CVSS Attack Complexity Metric distinguishes between easily repeatable success in exploiting the vulnerability (value Low (L)) and attacks that depend on conditions which are beyond the control of the attacker (value High (H)).
  - CVSS Privileges Required Metric (CVSS-PR)

    The CVSS Privileges Required Metric denotes if no privileges are required to exploit the vulnerability (value None (N)), if user privileges are required (value Low (L)) or if administrative privileges or other significant privileges are required (value High (H)).
  - CVSS User Interaction Metric (CVSS-UI)

    The CVSS User Interaction Metric decides if interaction from a human user is required for a successful attack (value Required (R)) or if no interaction from a third party is needed (value None (N)).
- Impact Metric Group
  - CVSS Confidentiality Metric (CVSS-C)

    The CVSS Confidentiality Metric denotes if confidentiality of data is totally lost or if there is a

direct, serious impact (value High (H)), if only some data is impacted (value Low (L)) or if no loss of confidentiality is occurring (value None (N)).

  - CVSS Integrity Metric (CVSS-I)

    The CVSS Integrity Metric denotes if integrity of data is totally lost or if there is a direct, serious impact (value High (H)), if only some data is impacted (value Low (L)) or if no loss of integrity is occurring (value None (N)).

  - CVSS Availability Metric (CVSS-A)

    The CVSS Availability Metric denotes if availability of the impacted component is totally lost or if there is a direct, serious consequence (value High (H)), if there is only reduced performance or interruptions in the component's availability (value Low (L)) or if no loss of availability is occurring (value None (N)).

- CVSS Scope Metric (CVSS-S)

  The CVSS Scope Metric shows if the metrics from the impact metric group impact the vulnerable component itself and its scope (value Unchanged (U)) or if other components outside its security scope are impacted too (value Changed (C)).

CVSS Temporal Score

The Temporal Score of CVSS is measuring the characteristics of a vulnerability that change over time. According to the specification it is an optional part of CVSSv3.1. A vulnerability can be in different stages of exploitability, going from a theoretical vulnerability over a Proof of Concept (PoC) to a proven and guaranteed to work exploit. As well as the exploit maturity can change over time, the remediation possibilities of a vulnerability will, in most cases, improve over time. Starting with unavailable solutions, going over workarounds and temporary fixes to official fixes for a vulnerability. The default values for the Temporal Score Metrics is *Not Defined (X)*, which does not change the Base Score when applied.[4]
The complete list of metrics that form the **Temporal Score** are listed below.[4]

- CVSS Exploit Code Maturity Metric (CVSS-E)

  The CVSS Exploit Code Maturity Metric defines which exploits are available for a given vulnerability. It goes over the state of exploit techniques, availability of exploits and active exploitation of vulnerabilities.

  - Not Defined (X)

    This metric value changes the metric to have no impact on the Temporal Score. This is for situations when there is insufficient information available to choose a value for a metric.

  - High (H)

    The High value of exploit maturity denotes exploits which are widely available, working in every situation or no exploits are required to exploit the vulnerability. These generate the most severe score since these vulnerabilities can impose a great risk.
  - Functional (F)

    Functional exploits are exploits that do only work in most situations but not in every situation.
  - Proof-of-Concept (P)

    Proof of Concept (PoC) exploits work in demonstrations but are not practical for most systems. They require substantial modification for usage.
  - Unproven (U)

    No exploit code is known to be available or an exploit is only theoretically possible.

- CVSS Remediation Level Metric (CVSS-RL)

  The CVSS Remediation Level Metric denotes the different stages that remediation of the vulnerability can be in. The specification of CVSS says: "The less official and permanent a fix, the higher the vulnerability score."[4]. This is understandable for scoring the severity of a vulnerability itself but can be backwards for usage for planning remediation measurements.
  - Not Defined (X)

    This metric value changes the metric to have no impact on the Temporal Score. This is for situations when there is insufficient information available to choose a value for a metric.
  - Unavailable (U)

    There is no remediation available for the vulnerability that can be applied.
  - Workaround (W)

    Workarounds are defined per the specification as "unofficial, non-vendor solution"[4].
  - Temporary Fix (T)

    Temporary Fixes are fixes issued by the vendor which can be anything temporary like workarounds and temporary patches.
  - Official Fix (O)

    When an official solution (patch, update or upgrade) is provided by the vendor, which can be applied at will, the value Official Fix is to be chosen.

- CVSS Report Confidence Metric (CVSS-RC)

  The CVSS Report Confidence Metric gives a measurement of two factors that affect the quality of the scoring of a particular vulnerability. One factor is the degree of confidence to correctly identify

a vulnerability and its existence, for example with an automated vulnerability scanner. The second factor is the credibility of the technical details of a vulnerability. The more certain a vulnerability is, the higher its score.

- Not Defined (X)

  This metric value changes the metric to have no impact on the Temporal Score. This is for situations when there is insufficient information available to choose a value for a metric.

- Confirmed (C)

  The Confirmed value is chosen when certainty is reached that a vulnerability exists. This can be done via functional exploits, availability of the source code which can be verified independently or via acknowledge of the author of the vulnerable component.

- Reasonable (R)

  The value Reasonable is chosen, when significant details are available, but confidence is not high enough or requirements are missing for assigning the Confirmed value.

- Unknown (U)

  This value indicates that it is unknown what the cause of a certain vulnerability is or if there are discrepancies in reports. This can be assigned, if it is uncertain, if a weakness is actually a vulnerability that could be exploited if more resources are invested or more details get known.

CVSS Environmental Score

The third score completing CVSS is the Environmental Score. According to the specification it is optional. It consists of two parts, the Security Requirements and the Modified Base Metrics. The Environmental Score connects a vulnerability that is rated independent of the system it is found on and adds context of the actual instance that is present on a given system in a given environment. By defining the Security Requirements, which consist of scores rated on confidentiality-, integrity- and availability requirements on an instance basis, different occurrences of the same vulnerability can take the requirements of different systems into account and reduce or increase the total score. This is based on the principle that, if a requirement is higher and it could be affected by the vulnerability, the vulnerability gets a higher score. If the requirement is low and it could be affected by the same vulnerability, the score is reduced. The Modified Base Metrics, override all of the Base Metrics defined above with new values. This enables the inclusion of mitigations to the score, which are in place on a given system. It can also be used to increase the score of a given instance if the Base Metrics were chosen too low. The default value for the Environmental Score Metrics is *Not Defined (X)*, which does not change the Base Score when applied.[4]

The metrics contained in the CVSSv3.1 **Environmental Score** are listed below.[4]

- Security Requirement Metric Group

  The Security Requirements Metric Group consists of CVSS Confidentiality Requirement Metric (CVSS-CR), CVSS Integrity Requirement Metric (CVSS-IR) and CVSS Availability Requirement Metric (CVSS-AR). They all have the same possible values associated with them, which are representing the requirements of security in the three categories that a given system possesses. The requirements only impact the score, if there is an impact defined on the corresponding impact metrics. The weighting is neutral when the value Medium (M) is chosen.

  - Not Defined (X)

    This metric value changes the metric to have no impact on the Score. This is for situations when there is insufficient information available to choose a value for a metric.
  - High (H)

    The High value denotes a catastrophic effect if the security requirement is lost.
  - Medium (M)

    The Medium value denotes a serious effect if the security requirement is lost.
  - Low (L)

    The Low value denotes a limited effect if the security requirement is lost.
- Modified Base Metric Group

  The Modified Base Metrics override the corresponding Base Metrics with the value given here. They are applied prior to the Security Requirements Metrics and are used to correct the Base Score to take mitigations on the instance of the vulnerability into account and to increase the score if required. The values of all the metrics are the same as in the Base Score, therefore only the names of the Metrics are listed. In addition to the values of the Base Metrics there is also the value *Not Defined (X)*. If it is assigned the value of the Base Metric is used for the Modified Base Metric.

  - Modified Exploitability Metric Group
    - CVSS Modified Attack Vector Metric (CVSS-MAV)
    - CVSS Modified Attack Complexity Metric (CVSS-MAC)
    - CVSS Modified Privileges Required Metric (CVSS-MPR)
    - CVSS Modified User Interaction Metric (CVSS-MUI)
  - Modified Impact Metric Group
    - CVSS Modified Confidentiality Metric (CVSS-MC)
    - CVSS Modified Integrity Metric (CVSS-MI)

    - CVSS Modified Availability Metric (CVSS-MA)
  - CVSS Modified Scope Metric (CVSS-MS)

2.3.2 Calculation of CVSS

The calculation of the CVSS score is defined with multiple equations for the different scores. The equations use a lookup table to assign numerical values to the qualitative values of the different metrics, described in section 2.3.1. This lookup table can be found at the end this section in table 2.3.

Base Score

The CVSSv3.1 Base Score Formula in equation 2.4 is created from the sub-formulas the Impact Sub-Score in equation 2.1, the Impact in equation 2.2 and the Exploitability in equation 2.3.[4]

$$ISS = 1 - [(1 - Confidentiality) \times (1 - Integrity) \times (1 - Availability)]$$

Equation 2.1: CVSSv3.1 Impact Sub-Score (ISS) Formula [4]

$$Impact = \begin{cases} 6.42 \times ISS & \text{If } Scope \text{ is } Unchanged \\ 7.52 \times (ISS - 0.029) - 3.25 \times (ISS - 0.02)^{15} & \text{If } Scope \text{ is } Changed \end{cases}$$

Equation 2.2: CVSSv3.1 Impact Formula [4]

$$Exploitability = 8.22 \times AttackVector \times AttackComplexity \times PrivilegesRequired \times UserInteraction$$

Equation 2.3: CVSSv3.1 Exploitability Formula [4]

$$BaseScore = \begin{cases} 0 & \text{If } Impact \leq 0 \\ \text{Roundup(Minimum}[(Impact + Exploitability), 10]) & \text{If } Scope \text{ is } Unchanged \\ \text{Roundup(Minimum}[1.08 \times (Impact + Exploitability), 10]) & \text{If } Scope \text{ is } Changed \end{cases}$$

Equation 2.4: CVSSv3.1 Base Score Formula [4]

Temporal Score

The CVSSv3.1 Temporal Score Formula is given in equation 2.5. It uses the Base Score directly and modifies the result. Therefore, the Temporal Score combines the result of the Base Score with its own metrics. [4]

In this representation of the Temporal Score Formula $BaseScore$ is used like a variable, to simplify the Environmental Score Formula in equation 2.9. By doing that the formula can be applied to the Environmental Score as a function and enables a shorter representation. For calculation of the Temporal Score, the Base Score value can be substituted for the $BaseScore$ variable.

$$TemporalScore(BaseScore) = \text{Roundup}(BaseScore \times ExploitCodeMaturity \times RemediationLevel \times ReportConfidence)$$

Equation 2.5: CVSSv3.1 Temporal Score Formula [4]

Environmental Score

The CVSSv3.1 Environmental Score Formula in equation 2.9 is very similar to the Base Score Formula since it contains all metrics of the Base Score as Modified Metrics. The Environmental Score Formula uses three sub-formulas to calculate the Environmental Score, the Modified Impact Sub-Score ($MISS$) Formula in equation 2.6, the Modified Impact Formula in equation 2.7 and the Modified Exploitability Formula in equation 2.8. In contrast to the Impact Sub-Score (ISS) Formula the sub-formula Modified Impact Sub-Score ($MISS$) contains the CVSSv3.1 Security Requirement Metrics. The Temporal Score Formula is used in equation 2.9 as a function for a shorter representation. The Modified Base Score, created with the Modified Base Metrics is substituted into the function represented in equation 2.5. If Modified Base Metrics have the value *Not Defined (X)* assigned, the value of the Base Metric is used in place of the Modified Base

Metric. The Environmental Score includes all metrics from the Base Score and the Temporal Score and represents a total result for a vulnerability instance at a specific time in a particular environment. [4]

$$MISS = \text{Minimum}(1 - [\quad (1 - ConfidentialityRequirement \times ModifiedConfidentiality)\times (1 - IntegrityRequirement \times ModifiedIntegrity)\times (1 - AvailabilityRequirement \times ModifiedAvailability)], 0.915)$$

Equation 2.6: CVSSv3.1 Modified Impact Sub-Score ($MISS$) Formula [4]

$$ModifiedImpact = \begin{cases} 6.42 \times MISS & \text{If } ModifiedScope \text{ is } Unchanged \\ 7.52 \times (MISS - 0.029) - 3.25\times \\ (MISS \times 0.9731 - 0.02)^{13} & \text{If } ModifiedScope \text{ is } Changed \end{cases}$$

Equation 2.7: CVSSv3.1 Modified Impact Formula [4]

$$ModifiedExploitability = 8.22 \times ModifiedAttackVector \times ModifiedAttackComplexity\times ModifiedPrivilegesRequired \times ModifiedUserInteraction$$

Equation 2.8: CVSSv3.1 Modified Exploitability Formula [4]

$$EnvironmentalScore = \begin{cases} 0 & \text{If } ModifiedImpact \leq 0 \\ \left.\begin{array}{l} TemporalScore(\text{Roundup}(\text{Minimum}(\\ (ModifiedImpact+ \\ ModifiedExploitability) \\ ,10))) \end{array}\right\} & \begin{array}{l} \\ \text{If } ModifiedScope \\ \text{is } Unchanged \end{array} \\ \left.\begin{array}{l} TemporalScore(\text{Roundup}(\text{Minimum}(\\ 1.08 \times (ModifiedImpact+ \\ ModifiedExploitability) \\ ,10))) \end{array}\right\} & \begin{array}{l} \\ \text{If } ModifiedScope \\ \text{is } Changed \end{array} \end{cases}$$

Equation 2.9: CVSSv3.1 Environmental Score Formula [4]

Numerical Metric Values

Table 2.3 lists the corresponding numerical values for each metric and each metric value. The numerical values are substituted into the CVSS formulas for the corresponding qualitative metric value which has been chosen. The metrics are assigned the metric value as described in section 2.3. With this assignment the table can be used to get the numerical values for the equations listed the beginning of this section. [4]
In the representation, the value *Not Defined* is not listed for the CVSS Modified Base Metrics, since they do not need numeric values. When it is chosen for the Modified Base Metrics, the value of the same Base Metric is chosen instead, which needs to be defined per the specification. [4]

Metric	Metric Value	Numerical Value
CVSS-AV, CVSS-MAV	Network	0.85
	Adjacent	0.62
	Local	0.55
	Physical	0.2
CVSS-AC, CVSS-MAC	Low	0.77
	High	0.44
CVSS-PR, CVSS-MPR	None	0.85
	Low	0.62 (0.68 if CVSS-S/CVSS-MS is Changed)
	High	0.27 (0.5 if CVSS-S/CVSS-MS is Changed)
CVSS-UI, CVSS-MUI	None	0.85
	Required	0.62
CVSS-C, CVSS-I, CVSS-A, CVSS-MC, CVSS-MI, CVSS-MA	High	0.56
	Low	0.22
	None	0
CVSS-E	Not Defined	1
	High	1
	Functional	0.97
	Proof of Concept	0.94
	Unproven	0.91
CVSS-RL	Not Defined	1
	Unavailable	1
	Workaround	0.97
	Temporary Fix	0.96
	Official Fix	0.95
CVSS-RC	Not Defined	1
	Confirmed	1
	Reasonable	0.96
	Unknown	0.92
CVSS-CR, CVSS-IR, CVSS-AR	Not Defined	1
	High	1.5
	Medium	1
	Low	0.5

Table 2.3: CVSSv3.1 Metrics, with qualitative- and numerical-values for calculation [4]

2.3.3 CVSS Version 4.0 - CVSSv4

As described in section 2.3, the CVSS SIG is already working on improving CVSSv3.1. The current state of improvements is published in [45]. It contains an improvement tracker with approved and pending changes for CVSS Version 4.0 (CVSSv4). The currently approved proposals that will be added into Version 4 are listed below[45]. Clarifications to some of these points were added from the presentation by Dugal *et al.*[58] held at the FIRST SIG Updates: EPSS, CVSS, Ethics, Women of FIRST | Thursday. May 13[59].

- Changes to Temporal Metric Group
  - Temporal Metric Group replaced with Threat Metric Group [45], [58]
  - Remediation Level (CVSS-RL) and Report Confidence (CVSS-RC) Metrics removed [45], [58]
  - Exploit Code Maturity updated to Exploit Maturity (CVSS-E) [45], [58]

  These changes suggest that the Temporal Score, which was often not utilized[23], [59], is changed to only include CVSS-E together with some other metrics that are to be defined. Currently discussed proposals are Threat Intelligence, Collateral Damage and others[45]. Metrics like CVSS-RL and CVSS-RC were removed since they have not been understood and were problematic, as described in [60] (chapter 3 goes into more detail on this work).
- Attack Requirements (AR) added as Base Metric (New metric approved already in 2017 before release of CVSSv3.1[59])
- Scope Metric expanded to tri-state value: Unchanged/VulnerableComponent/ImpactedComponent
  This was done to clarify on which component the ratings of the Impact Metric Group (CVSS-C, CVSS-I, CVSS-A) are based upon, which is currently not distinguishable[59].
- Explicit Nomenclature added to specification: CVSS-BTE [45], [58]
  This is to explicitly tell from wording, if only the Base Score or all score components are assessed, which is not possible from the often used wording “CVSS Score”. When the full set of metrics is used, the new nomenclature should be “CVSS-BTE Score”. BTE in this case stands for “Base-Threat-Environmental”[59].
- New Supplemental Metric Group (including approved component Automatable) [45], [58]
  This new metric group should not influence the scoring but provide additional information about vulnerabilities. Automatable should tell, if a vulnerability can be traversed automatically in a network as if it is “wormable”. It is inspired by the Stakeholder-Specific Vulnerability Categorization (SSVC)[61]. More components that are proposed and not yet approved are Recovery, Mitigation Effort and Provider-Specific Urgency. Recovery denotes if the vulnerable component automatically restarts after being exploited or if it is nearly impossible to recover like corrupted firmware. Mitigation

Effort shows how hard it is to defend from the exploitation of a specific vulnerability[59].

- Enhanced User Interaction (CVSS-UI) Granularity (None/Active/Passive) [45], [58]
  This was introduced to differentiate if the user needs to actively do something (which can be avoided with security awareness) to make the exploit happen or if the interaction happens with normal user behavior[45].
- Repudiation Clarification for Integrity [45], [58]
  This should be clarification for the CVSS Integrity Metric (CVSS-I) metric to clarify exploits that try to remove traces to create repudiation[59].

Other goals for CVSSv4 are incorporating Operational Technology (OT) and safety impacts of exploitation to the real "physical" world. Another important goal for CVSSv4 is aiding the importance of using Threat and Environmental Metrics, which should be filled in by users, according to [58], [59]. Additionally, the support for undefined Base Metric Values is being discussed and the return of Target Distribution from CVSSv2. Furthermore, adding a metric to indicate likelihood of exploitation in the wild of a vulnerability is in the state of being discussed[45]. The proposal from CVSS SIG was approved to create a Best Current Practices (BCP) document to guide CVSS users how CVSS is incorporated and used into risk analysis and response priority[45], [59]. A complete list of all proposals can be found in [45] and the CVSS SIG is open to new participants to discuss CVSSv4[7].

Another proposal, which could be very important for CVSSv4 adoption, is, to provide an official backwards compatibility to convert CVSSv3 scores to CVSSv4. The current state of this proposal is "Awaiting completion of CVSSv4 modifications"[45, p. 15], which is not clear if that means it is approved or approval will be done when CVSSv4 is finished. This could be detrimental to the adoption speed of CVSSv4 and the discussed problems introduced with the switch from CVSSv2 to CVSSv3 should not be repeated.

[7] https://www.first.org/cvss/#Current-initiatives

2.4 Other Scoring Systems used by the Industry

Even though CVSS is a widely used scoring system for vulnerabilities and probably every vendor has adopted it into their solutions, there exist other scoring systems too, which can be used to score vulnerabilities to prioritize and compare them. This section gives an overview over popular alternatives to CVSS, which are mostly created by commercial software companies, they are popular partially due to the market strength of the companies. Furthermore, other scoring systems are included, that possess high expectations to be used for further improvement of this research.

2.4.1 Exploit Prediction Scoring System - EPSS

In FIRST's efforts to improve prioritization of remediation efforts, the community-driven effort, the Exploit Prediction Scoring System (EPSS) was created. The EPSS has been first presented at WEIS 2019[8] in Boston[62] and later at Blackhat 2019[9][23] which resulted in two papers describing the model. It builds upon different aspects of automatically retrieving publicly available data over vulnerabilities denoted by their CVE IDs and creates an estimate of the probability of a vulnerability being exploited in the first year after disclosure in the real world. It is based on machine learning algorithms, using data from MITRE CVE, NVD (CVSS Base Score, CPE, descriptions and references), text in references, exploit information (from Exploit Database (EDB), Rapid7's Metasploit framework[10], D2 Security's Elliot Framework[11] and the Canvas Exploitation Framework[12])[23]. How these data-sources are combined is described in detail in [62]. The model has been trained on data from multiple commercial security vendors (Proofpoint, Fortinet and AlienVault), which is not public, but provides insight into exploitation of vulnerabilities in real scenarios[23]. The model does not take into account several web application vulnerabilities like Cross-Site-Scripting (XSS) and other exploits that do not have Intrusion Detection System (IDS) signatures, due to missing training data. Furthermore, the project only focuses on vulnerabilities that have CVE IDs, other vulnerabilities were not considered. The created score is dependent on the vulnerability itself and does not take the environment of organizations into account. The goals for the initiative were to create a model that is simple to implement, uses only public data sources, is easily updatable and extendable, interpretable and transparent while still being accurate[23]. The model provides a score as soon a vulnerability is released and can be easily updated over time to reflect changes in threats. When compared with CVSSv3.1 Base Scores, the EPSS shows a

[8] https://weis2019.econinfosec.org/
[9] https://www.blackhat.com/us-19/
[10] https://www.rapid7.com/products/metasploit/
[11] https://www.d2sec.com/elliot.html
[12] https://www.immunityinc.com/products/canvas/

significant reduction in the number of vulnerabilities needed to close (78%) to achieve the same number of closed vulnerabilities that are actually exploited determined from the training data, when comparing policies that require to remediate all vulnerabilities above a certain score threshold of both systems[23], [59], [63].
In Section 7 of [23], considerations how to use EPSS were given. These conclude, that CVSS and EPSS should coexist and give different recommendations for that, since risk includes threat as well as severity. Furthermore, the paper argues, that combining CVSS and EPSS is not possible with simple arithmetics alone, since there could be a gap for vulnerabilities that are rated medium in both scoring systems. The recommended approach would be to prioritize vulnerabilities with high severity and high probability, de-prioritize vulnerabilities with both measures low and further evaluate vulnerabilities in-between. Figure 2.1 further illustrates the correlation between EPSS and CVSSv3.1 in a scatter plot. The darker the color of a point in the figure, the more vulnerabilities are corresponding to the position. As per the EPSS User Guide [64], the vulnerabilities in the upper left quadrant, where exploit probability is high and severity is low, should be investigated further, why they are exploited, even though their impact is rated low. These could indicate vulnerabilities used for chained attacks. Vulnerabilities in the bottom right quadrant are less likely to be exploited, but should be watched closely, due to possible changes in threat landscape and their high severity. The EPSS Special Interest Group (EPSS SIG) talk[59], suggested to set the threshold to create the quadrant in the two scoring systems as policies of the organization require and how much remediation effort is available. Section 6.4 will go into more detail on how the EPSS can be used combined with the CVSS.
EPSS scores and statistics are currently published daily on the EPSS project website[65] for all MITRE CVE IDs since 2017. Furthermore, EPSS Version 2 is currently being worked on by the EPSS SIG[59]. An important target for the work is to keep the data centralized, unlike CVSS Temporal Scores, to ensure that the data is always available and up to date. Another plan is to incorporate adjustable times into the model to realize a model that is more flexible in predicting in which time the exploitation of a CVE will take place. Further efforts also could be beneficial to the system, like having more standardized descriptions of CVEs or even machine tags in databases. Currently CWE information is not incorporated into the calculations due to the complex hierarchy of them. Another planned improvement to EPSS will be the repeated reevaluation of the factors that create the score to incorporate changes in threat landscape, which furthermore enables further analysis of the threat landscape. Limitations to the model are currently mostly created through biased training data, that is currently only collected from network-based IDS systems and these are eventually missing some attack types and niche usecases like ICS exploitation could be under-represented in the training data. [23] There is a call to action to help and provide input or data to contribute to this effort[13].

[13] https://www.first.org/epss/

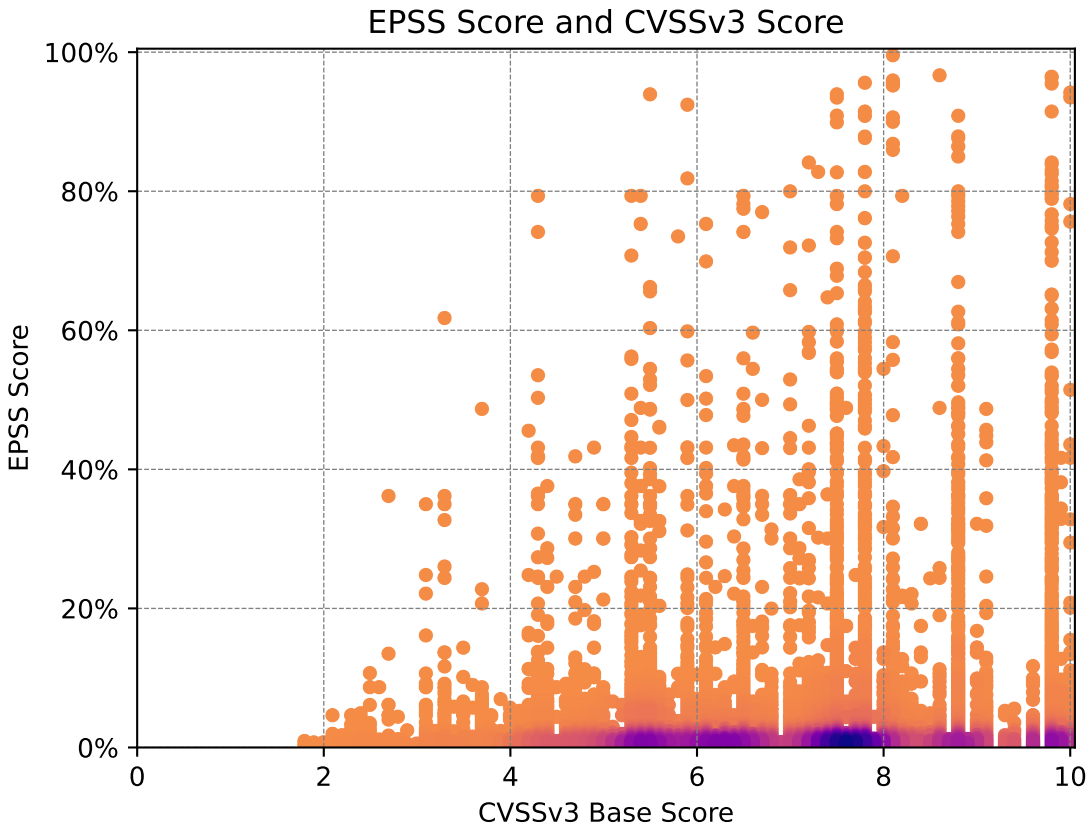


Figure 2.1: Correlation between EPSS and CVSSv3.1 scores retrieved on August 18, 2021 with CVSS data from NVD[52] and EPSS data from [65]

The calculation of EPSS is using linear regression and is incorporating the 16 most significant variables as presented in the publications[23], [59], [63]. The variables are named with the type of variable and then its measurement, which is separated by a dot. $vend$ stands for the vendor of the vulnerable component, exp denotes the state of exploit code existing for the vulnerability in PoC or more sophisticated ways $weaponized$. tag is used for properties of the vulnerability and its impact. $ref.count$ is the count of the references of the published CVE which includes a transformation to account for the possible value 0. The $LogOdds$, which is created from the coefficients of the regression in the EPSS model, in equation 2.11, is calculated using the terms in equation 2.10.[23]

$$
\begin{aligned}
LogOdds = -6.18 + 2.44 \times vend.microsoft \quad & +2.07 \times vend.ibm \\
+2.00 \times exp.weaponized \quad & +1.91 \times vend.adobe \\
+1.62 \times vend.hp \quad & +1.50 \times exp.poc\_code \\
+1.10 \times vend.apache \quad & +1.01 \times log(ref.count + 1) \\
+0.57 \times tag.code\_execution \quad & +0.23 \times tag.remote \\
+0.22 \times tag.denial\_of\_service \quad & +0.06 \times tag.web \\
-0.20 \times tag.memory\_corruption \quad & -0.63 \times tag.local \\
-0.89 \times vend.google \quad & -1.92 \times vend.apple
\end{aligned}
$$

Equation 2.10: EPSS Regression Coefficients [23], [63]

$$
Probability[exploitation] = \frac{1}{1 + e^{-LogOdds}}
$$

Equation 2.11: EPSS Logistic Regression Formula [23], [63]

2.4.2 Microsoft's Vulnerability Scoring System

Microsoft is using a combination of an impact factor, represented by its *Security Update Severity Rating System*[66] and a probability of exploitation, the *Microsoft Exploitability Index*[67] to rate and rank vulnerabilities in their software in addition to CVSS. It is available for all vulnerabilities disclosed in the Security Update Guide[14] and is assigned by the Microsoft Security Response Center (MSRC) directly without a public scoring system. Therefore, it is only available for Microsoft products and software.

Microsoft Security Update Severity Rating System

The *Security Update Severity Rating System* is a categorization into four different severity classes, which represent a risk "according to the worst theoretical outcome were that vulnerability to be exploited"[66]. It does not measure likelihood of exploitation and exploit availability. This is done with the *Microsoft Exploitability Index*. "The Security Update Severity Rating System assumes that exploitation will be successful."[67] The highest risk is represented by the rating *Critical*. It classifies a vulnerability which is able

[14] `https://msrc.microsoft.com/update-guide`

to give the attacker remote code execution without user interaction required. It is followed by the rating *Important*, which requires user interaction or displays warnings to the user. The next rating is *Moderate*, which is mitigated by its nature due to authentication requirements or is only applicable with non-default configurations. The lowest rating is *Low*, where the impact is mitigated by the characteristics of the affected component itself. The rating itself is very similar to the qualitative severity rating of CVSS shown in table 2.1.

Microsoft Exploitability Index

The *Microsoft Exploitability Index* is a number ranging from 0 to 3[67]. It was introduced in 2009[68] and refined in 2011[23]. The value 0 means exploitation detected. It is assigned, when Microsoft is aware of exploits that are happening on the vulnerability and represents the highest priority recommendation. The value 1 is associated with a vulnerability when Microsoft analyzed that a consistent exploitation of the vulnerability would be possible. The assigned textual meaning is exploitation more likely. 2 means exploitation less likely and is assigned when additional expertise, sophisticated timing and/or varied results are to be expected for creating exploits. The lowest value is 3, which means exploitation unlikely. It is assigned when it is unlikely that a vulnerability is exploited in real attacks. If multiple products are affected the lowest number, which is the highest Exploitability score, is chosen for the index. Furthermore, Microsoft is observing the threat landscape and incorporates the type of vulnerability and past exploits for similar types of vulnerabilities. This data is also added to the score. Exploitability is updated by Microsoft when new details about vulnerabilities are being discovered in the first month reliably. After that Microsoft sees little need for updating the score since “customers have made their prioritization decisions, the Exploitability Index will not be updated as it is no longer useful to the customer”[67]. According to [60] Microsoft started in November 2020 to also include CVSS Temporal Scores with all vulnerabilities affecting its products.

The Microsoft Scoring System for vulnerabilities differs vastly from CVSS. CVSS is based on different metrics and calculates a numeric value with much higher possibility of rating values. The Microsoft Scoring System, consisting of the Exploitability Index and the Security Update Severity Rating, is not as transparent and the creation of the score is not easily understood. The differences in CVSSv2 and the Microsoft rating are discussed in [68]. Furthermore, the CVSSv2 Base Score is compared with Microsoft’s Severity Rating and Exploitability Index. The authors conclude that both ratings have a high rate of false positives when it comes to measuring if exploits will be created.

2.4.3 Tenable Vulnerability Priority Rating - VPR

The commercial vulnerability management company Tenable, which created the Nessus vulnerability scanner and has been awarded the biggest vulnerability management company by revenue in 2019 [69], created their own rating for vulnerability prioritization. It is called Vulnerability Priority Rating (VPR) and applied to all vulnerabilities that have a CVE in the NVD[49]. The VPR is like the CVSS a numerical rating from 0.1 to 10.0 and the severity categories, including the assignment of the number-ranges, correspond to the qualitative severity ranking for CVSSv3.1, which are shown in table 2.1. In a blog post[70], Tenable explains how VPR works. They created the new rating, due to the criticism on CVSS and its inability to measure risk, since it measures severity instead[71]. The VPR is built from two components, the technical impact, equivalent to the CVSSv3 Impact Metric Group, the CVSSv3.1 Impact Sub-Score Formula and the threat component. The threat component is determined by threat intelligence by Tenable, with proprietary machine learning algorithms. It incorporates threat sources such as "public proof-of-concept (PoC) research, reports of exploitation on social media, emergence of exploit code in exploit kits and frameworks, references to exploitation on the dark web and hacker forums and detection of malware hashes in the wild"[70]. With that, the VPR provides a similar score as the EPSS combined with some CVSS metrics. The VPR score is updated daily, therefore it should always represent real threats. A key problem of CVSS, which VPR should overcome is the high percentage of critical vulnerabilities. According to Tenable, VPR should only have 1% to 4% on an average day, whereas CVSS has had 16%[70]. Tenable's own research showed, that VPR is approximately 22 times more efficient than CVSS in predicting if vulnerability IoCs are spotted after four weeks, depending on the severity calculated[70]. The measurement of IoCs is taken to identify exploitation of vulnerabilities. Furthermore, Tenable's research shows that, "VPR High and Critical are more strongly correlated to high/functional exploit code maturity than CVSS High and Critical"[70]. The VPR does not take the environment of the vulnerability into account and cannot be customized by users[49]. To achieve that, Tenable has created another product called Tenable *Lumin*[15], which builds upon the VPR score and their VM products to enable risk management. Tenable VPR is available on all their VM products, *Tenable.io*, *Tenable.sc* and its *Nessus* Vulnerability Scanner.

2.4.4 Red Hat Classification of Security Issues

Red Hat, the creator of Red Hat Enterprise Linux, is employing a similar rating to the Microsoft Security Update Severity Rating System. The Red Hat Security Ratings are a four point scale, representing the severity

[15] `https://de.tenable.com/products/tenable-lumin`

of a security issue in a Red Hat product. [72] The highest value of the scale is called *critical impact*. These denote easily exploitable vulnerabilities that do not require authentication and result in complete compromise of a system without user interaction. The next score is *important impact*. It is used for vulnerabilities that require authentication or that do not require authentication but do not affect a complete compromise. The third value, *moderate impact*, is used when the exploitation is difficult and therefore less likely. The lowest value is *low impact* and denotes all other security issues that have a security impact and do not fit into the other categories. The assignment of the scores is done by Red Hat experts and not via a formula like CVSS. The security bulletins contain detailed information on the (multiple) vulnerabilities, together with the Security Rating, as well as references to the contained CVEs, with their associated CVSSv3 Base Scores. CVEs from 2009 to 2016 contain CVSSv2 Base Scores and from 2016 on only CVSSv3 Base Scores are used. Red Hat explicitly states that the "scale takes into account the potential risk based on a technical analysis of the exact flaw and its type, but not the current threat level; a given rating will not change if an exploit or worm is later released for a flaw, or if one is available before the release of a fix"[72]. This means, that the Red Hat rating is only a measure of static severity like the CVSS Base Metric.

2.4.5 Qualys Severity Score

Qualys Inc., a major vendor of VM tools as per [69], comprises the severity levels of the detection of its products differently by not directly utilizing CVSS like Tenable Nessus. There are three categories of vulnerability detection, *Confirmed Vulnerabilities*, *Potential Vulnerabilities* and *Information Gathered*. The two vulnerability categories have each a severity rating with five values ranging from *Minimal* over *Medium* to *Serious* then *Critical* and lastly *Urgent*. These two categories are used to classify confirmed vulnerabilities and vulnerabilities that are potentially present but could not be confirmed. The third category *Information Gathered* represents information that could be gathered from a system. The information is also assigned a severity ranging from *Minimal* to *Medium* and *Serious*.[73]

The severities with the same name have the same underlying meaning, but they are tailored to the category in which they exist. As the lowest severity score *Minimal* denotes vulnerabilities that collect information about the system and could be used to find other vulnerabilities. *Medium* is already leaking sensitive information like precise software versions. Vulnerabilities rated *Serious* can be used to obtain specific information from the system, including security relevant information. In addition, denial of service attacks and unauthorized use of services is also covered by this severity. *Critical* vulnerabilities can be used to eventually obtain control over a system or a leak of highly sensitive information is possible. In contrast *Urgent* vulnerabilities can be easily used to gain control over a system, leading to a potential compromise of the entire network.[73]

The Qualys Severity Score is associated by their analysts based on risk associated to an exploited vulnerability. Factors such as complexity of the exploit, likelihood of successful exploitation, network location, privileges required and existence of attacks are also taken into consideration when assigning the severity score. In addition to its own score, Qualys is also publishing CVSS Base and Temporal Scores for its vulnerability findings. The same problems as with CVEs in the NVD apply for availability of CVSSv3 scores for vulnerabilities before 2016. The relationship between the Qualys Severity and the CVSS score is described as congruent by the documentation, even though the added information can cause significant deviations from the CVSS Score. A mentioned example for this is the Heart Bleed vulnerability of OpenSSL.[74]
It is also mentioned briefly in the documentation, that CVSS Environmental Metrics can be assigned to groups of systems, called *asset groups*, in the Qualys Vulnerability Management, Detection, and Response[16] product. This could be used with the approach presented in this thesis, to assign CVSS Environmental Scores to the results shown in the VM. There was no guidance to be found in the documentation of the product on how to assign the Environmental Metrics other than the CVSS Specification. It also seemed, that the CVSS Environmental Metrics are only based on CVSSv2, due to the use CVSSv2 Environmental Metrics represented in the current documentation.[75]

2.4.6 Rapid7 Risk Scores

The vulnerability scanner Nexpose[17] by Rapid7 is another popular commercial tool to identify vulnerabilities on systems. [69] identified Rapid7 as the third biggest vulnerability management company. The product InsightVM[18] extends the capabilities of the Nexpose vulnerability scanning with VM capabilities[76]. Both products include the risk quantification based on the *Real Risk Score*. This score is numerical and ranges from 1 to 1000. The score is created to be a replacement for CVSS to ease vulnerability prioritization. It includes CVSSv2 impact and likelihood metrics, which are not further explained. They are combined with exposure, malware kits, the rank of the exploit and the length of exposure. The risk strategy used by the product can be chosen by the user, the available scoring strategies created by Rapid7 in addition to *Real Risk* are *Temporal*, *TemporalPlus* and *Weighted*. [77]
The *Temporal* strategy does not have an upper bound and is increasing with the age of vulnerabilities, factoring in the CVSSv2 Impact Metrics, as well as the CVSSv2 Access Vector, Access Complexity and Authentication Required Metric. In contrast, the *TemporalPlus* strategy is expanding the risk contribution of the CVSSv2 Impact Metrics and includes a differentiation between the values *Partial* and *Complete* as

[16] https://www.qualys.com/apps/vulnerability-management-detection-response/
[17] https://www.rapid7.com/products/nexpose/
[18] https://www.rapid7.com/products/insightvm/

per the CVSSv2 specification[47]. The *Weighted* strategy uses importance associated to systems and adds the vulnerability's severity, the number of vulnerability instances and services running on the asset. The score is represented as a number from 1 to 100 and scales with the vulnerability count per asset. [78], [79] Nexpose and its different rating systems were analyzed in depth in [80]. A comparison between Tenable's VPR and Rapid7's approach was published by Tenable in [81]. It concludes that Rapid7's multiple scoring systems are not assessing the probability of exploitation in the future and instead mainly scores historic data.

2.4.7 Summary of Vulnerability Scoring Systems

The different commercial vulnerability scoring systems, even though they should all be used to prioritize vulnerabilities, differ in their characteristics. Table 2.4 lists some of these different characteristics to enable a comparison of the scoring systems. It is apparent that only the CVSSv2 Base Score is publicly available for all vulnerabilities, followed by the CVSSv3 Base Score for all vulnerabilities since 2016. The scores made by VM vendors are available too for most of their detected vulnerabilities, according to the documentations of the vendors, but require licensing und using their products and are therefore not publicly available. Many scores are not available for a broad number of vulnerabilities, like the vendor-specific scoring systems, which are only available for the products of the vendors themselves. What is furthermore a problem, is the bad availability of CVSS Temporal Scores. [60, p. 45] lists the IBM X-Force Exchange as a possible source for CVSS Temporal Scores for a considerable part of CVEs, which seems to be correct on a brief examination of the data available[82]. The examined vendor scores, create their scoring in categories, which are numbered or named directly. The CVSS scores have a qualitative category measurement, that can be inferred from the score, which was added to the specification of CVSSv3, since it was missing prior and caused multiple qualitative scales as shown in table 2.1. The VPR from Tenable, tries to mimic the CVSS score, by having a similar numeric and qualitative range as described in section 2.4.3. Most of the scores, created by vendors are either expert ratings, where assignment of the scores is solely based on expert opinion and the few calculated scores are not open and cannot be verified therefore. This is because of either missing the algorithms, the metrics or both. Only the Rapid7 Temporal Ratings have the formula on an old informational page listed [79] and their scores were further examined in [80]. Other examinations of different vulnerability scoring systems are performed in [83], [84].

The only real open scoring systems are CVSS and EPSS, which will be used in this thesis to create an open model for vulnerability prioritization. CVSS is a measure of the impact and severity of a vulnerability and EPSS can predict the probability of the vulnerability being actually targeted by attackers and being exploited as concluded in section 2.4.1 a combination of these two can be beneficial.

	Measurement of the Scoring System	Scale	Availability	Assignment Open and Verifiable
CVSSv2 Base Score	Severity of a Vulnerability	Numerical 0-10 with 1 decimal	For all Vulnerabilities in NVD	Yes [47]
CVSSv2 Temporal Score	Severity of a Vulnerability over Time	Numerical 0-10 with 1 decimal	In some Vendor Databases [60] and VM/Threat Intelligence products [82], [85]	Yes [47]
CVSSv2 Environmental Score	Severity of a Vulnerability over Time in a specific Environment	Numerical 0-10 with 1 decimal	Created Manually by Organizations	Yes [47]
CVSSv2 Qualitative Severity Rating by NVD	Qualitative Severity of a Vulnerability based on a CVSSv2 Score	3 Classes	With the CVSSv2 Score	Yes [48]
CVSSv2 Qualitative Severity Rating by Tenable	Qualitative Severity of a Vulnerability based on a CVSSv2 Score	5 Classes (including *None*/*Info*)	With the CVSSv2 Score	Yes [49]
CVSSv3 Base Score	Severity of a Vulnerability	Numerical 0-10 with 1 decimal	For all Vulnerabilities in NVD since 2016 [48]	Yes [4]
CVSSv3 Temporal Score	Severity of a Vulnerability over Time	Numerical 0-10 with 1 decimal	In some Vendor Databases [60] and VM/Threat Intelligence products [82], [85]	Yes [4]
CVSSv3 Environmental Score	Severity of a Vulnerability over Time in a specific Environment	Numerical 0-10 with 1 decimal	Created Manually by Organizations	Yes [4]
CVSSv3 Qualitative Severity Rating	Qualitative Severity of a Vulnerability based on a CVSSv3 Score	5 Classes (including *None*)	With the CVSSv3 Score	Yes [4]
EPSS	Probability of Exploitation of a Vulnerability	Numerical 0-100%	For all Vulnerabilities in NVD since 2017 [65]	Yes [23], [62]

Continued on the next page

	Measurement of the Scoring System	Scale	Availability	Assignment Open and Verifiable
Microsoft Security Update Severity Rating System	Severity of the Outcome of Exploitation of a Vulnerability	4 Classes	For Microsoft Vulnerabilities [66]	No, Expert Rating [66]
Microsoft Exploitability Index	Exploitation likelihood of a Vulnerability	Numerical Classes 0 to 3	For Microsoft Vulnerabilities updated in the first Month [67]	No, Expert Rating [67]
Tenable VPR	Vulnerability Prioritization with Impact and Threat	Numerical 0.1-10 with 1 decimal	For all Vulnerabilities in NVD with a CVE accessible through Tenable Products, updated daily	No, Metrics and Algorithm not public [70]
Red Hat Classification of Security Issues	Impact of exploitation of a Vulnerability	4 Classes	In Red Hat Security Bulletins created for Red Hat Vulnerabilities	No, Expert Rating [72]
Qualys Severity Score	Severity of a Vulnerability with the Class and Detection Certainty (or Information Gathered)	5 Classes times 3 Categories	In Qualys Products for all Detections	No, Expert Rating [74]
Rapid7 Real Risk Rating	Vulnerability Prioritization based on Impact, Exposure and Exploitation	Numerical 1-1000	In Rapid7 Products for all Vulnerabilities	No, full Algorithm with Metrics not public [77], [78]
Rapid7 Temporal and TemporalPlus Risk Rating	Vulnerability Prioritization incorporating Age of the Vulnerability	Numerical no upper bound	In Rapid7 Products for all Vulnerabilities	Yes, Formula based on CVSSv2 Metrics Public [79], [80]
Rapid7 Weighted Rating	Vulnerability Prioritization incorporating Importance of Systems	Numerical 1-100	In Rapid7 Products for all Assets	No, Algorithm with Metrics not public [78], [80]

Table 2.4: Comparison of Vulnerability Scoring Systems

3 Related Work

In this chapter an overview of related work to this thesis is provided. Related work, that is a prerequisite for this thesis has already been discussed in chapter 2. The chapter is structured into three parts, in which the related work could be categorized into. The section 3.1 is discussing related work, which is dedicated to analyzing the CVSS in various versions. Followed by section 3.2 improvements to CVSS in all versions are discussed, which were partially already added to CVSSv4. The last section, section 3.3 is dedicated to related work, which created other rating and prioritization methods for VM that are not improving CVSS and rather recreating different methodologies. In the end of this chapter, in section 3.4, the related work is summarized and a discussion is performed. Furthermore, the findings of some of the related analysis and improvements to CVSS are being shown which are later used as requirements to create the model.

3.1 CVSS Analysis

The credibility of CVSSv3 Base Metrics is analyzed in [86]. The analysis took CVSS vectors from different sources and showed that NVD is the database with the best assessed result, when comparing the probability of a result being correct. Bayesian analysis is used to statistically assess the values of the different CVSS metrics. The matching of the Impact Metric Group shows good results on all assessed databases, but other metrics like CVSS Privileges Required Metric (CVSS-PR) show discrepancies on some tested vulnerabilities. A possible usecase of the analysis performed is according to Johnson *et al.*, the use of the results to calibrate decisions when prioritizing risks and threats depending on sources and different metrics.
Allodi *et al.* compared the CVSSv2 scores published in the NVD with real world attacks and exploits traded in black markets in [87]. It also goes into detail on the differences of open and closed sources of exploits and how they represent the exploits that are actually happening. The authors concluded that NVD and EDB do not represent a reliable source of exploits that are used in the real world. It was found that CVSS (Base Score) reduced on severity classes by NVD is not a good indicator of exploits happening.[87]
Further research [88], [89] improves upon the results of [87], by evaluating the performance of CVSSv2 for risk reduction. The paper concluded that the CVSS Base Score is “equivalent to randomly picking

vulnerabilities" [89, p. 1] in predicting the likelihood of exploitation. The study suggests including the existence of PoC exploits and existence of black market exploits to improve the scoring.[89] The usage of black market data of exploit existence is backed by [88]. With CVSSv3.1 this information is included into the score via the optional Temporal Score as described in section 2.3.1. Tenable VPR includes this information as presented in section 2.4.3 and EPSS is also incorporating the existence of exploits to predict the probability of future exploitation as shown in section 2.4.1.

The paper [54] created a collection of problems observed with the CVSSv2 and how the newly released CVSSv3 specification at the time of creation of the work was changing these problems. The authors have also noted that a conversion from the old to the new standard would be required, which is still not officially possible today, as discussed in section 2.3. Furthermore, the problematic of organizations only using the CVSS Base Scores, instead of the Temporal or the Environmental Scores, due to availability and problems arising from that are described.

In [71] and [90] Spring *et al.* examine the misuse, against the advice of the CVSS SIG, of CVSSv3.0 as a risk measurement, even though it is just a measure of technical severity according to them. Furthermore, the problem that justification of the CVSSv3.0 formulas is not existing is examined. Problems identified with CVSSv3.0 listed in the papers are the complexity of chaining vulnerabilities which is not handled by CVSS, context missing from the score, not accounting for threat and consequences of a vulnerability and the possible inconsistent assignment of scores by security analysts. The papers summarize these problems and suggests fixing the CVSS or replace it, but there are no direct solutions for the problems given.

The article [91] investigated the disagreement and variance of the CVSSv2 Base Score by conducting surveys with experts. The experts were also consulted on improvements to CVSS. A notable finding was an information requirement concerning the environment in which the vulnerability is present to improve the rating. This further aids the research of this thesis. The survey has been conducted on a large scale of vulnerabilities (more than 3000) with 384 experts. The conclusion of the article is, "that there is a significant difference between the scores provided by experts and the Base Score"[91, p. 28]. When specific vulnerability types are observed, the direction in which these differences go (more or less severe), is dependent on the vulnerability type.

The differences between famous vulnerabilities (vulnerabilities with huge media attention) and vulnerabilities that are critical, but do not get reported in news are investigated in [53]. Discrepancies in the CVSS scores and the analyzed so-called "rock star vulnerabilities" are seen, since only one of them has the maximum CVSS score. Furthermore, the analysis conducted suggests, that analysts could be biased, when assigning CVSS Metrics to the vulnerabilities that have already caused damaged in real world. This bias

is seen since these vulnerabilities tend to get a higher score. The conclusion suggests, that the analyzed CVSSv2 and CVSSv3 does not cover all facets of vulnerabilities and therefore it cannot be derived from the CVSS metrics, if a vulnerability gets huge attention.

Boechat *et al.* analyzes the CVSSv3 Temporal Score in their recent work [60]. The conclusion, which was confirmed by the CVSS SIG and added to the CVSSv4 approved proposals[45], is that CVSS-RC is redundant, since its information can be retrieved by the two other CVSS Temporal Metrics (CVSS-E and CVSS-RL). Furthermore, their work analyzes different public sources for the CVSS Temporal Score Vectors, that show discrepancies when compared, due to not updated data. Different estimations are proposed to estimate the CVSS Temporal Metrics with publicly available data, like from EDB or Metasploit.

Ruohonen analyzed the time delays of availability of CVSS in NVD over years[92]. The research found, that in 2017, when the analysis was done, the time delays were negligible and therefore the usage of NVD could not cause delays to VM. The analysis however revealed, that between 2000 and 2007, the delays were indeed long which could affect the results of analysis that were based on this data. The different CVSS metrics and the resulting severity were also analyzed in context to delays, which did not show any correlation between them.

The work of Allodi *et al.* is empirically analyzing the difficulty in using the CVSS Environmental Metrics to model changes in security requirements of a system[6]. Their experiment shows that the assignment according to the CVSSv3.0 standard is difficult to perform correctly. The paper suggests integrating network dependencies, configurations and mitigation controls and system interaction in addition to attack graphs for automating this assignment. Another recommendation is to use Business Impact Analysis of organizations to create the Security Requirement Metrics.[6] In another article Allodi *et al.* analyzed the difference in security education when performing CVSSv3 scoring of vulnerabilities. A clear difference in accuracy could be seen between security unskilled and skilled personnel. The difference between security students and security professionals seemed to be less pronounced.[93]

In [5] a software tool was created to study the effect of different CVSSv2 metrics, by calculating all possible values for each metric. The applications should assist at selecting Environmental Metrics to achieve a certain score. The motivation of this work is not easily understood, since VM should not be an exercise on reducing the scores, to be compliant, instead the scores should reflect the reality as much as possible. The work gives no guidance on which remediation actions need to be performed, to reduce the risk to the new metric values chose to reduce the CVSSv2 Environmental Score.

The paper [94] created a model to estimate the cost of adding more information to CVSS based vulnerability prioritization and compared it to the cost of critical responses due to inaccurately scored vulnerabilities.

Their finding is that in general the employment of CVSS Temporal and Environmental scores can reduce the average CVSS score by about 0.5, but the number of *critical* vulnerabilities decreased drastically. A notable finding in the analysis conducted is that the CVSS Security Requirement Metrics do not factor in the overall importance of a system to an organization.

The paper [95] crated an analysis using a variation of the CVSSv2 Security Requirement Environmental Metrics. All values that are *Low* and *High* were assigned to Android Vulnerabilities and the effect on the number of vulnerabilities per severity class, as defined in the NVD[48] were observed. The CVSSv2 Environmental Metrics, which were removed in CVSSv3 have not been analyzed in this paper. The conclusion of the paper was that depending on the values selected, a significant change could be made to the number of vulnerabilities per class.

3.2 CVSS Improvements

Walkowski *et al.* created a new tool, called Vulnerability Management Centre (VMC), to automatically calculate the CVSS Environmental Score [96], [97]. The system is collecting the data required for the calculation with different modules and presents calculated Environmental Scores for CVSSv2 and CVSSv3. The VMC is open source and is published on the projects GitHub page[1]. At the time of writing active development on the project was taking place, which could be seen, by recent commits. The created tool did not employ any means of estimating the metrics used in the calculation of CVSS scores. The metrics required for calculating the Environmental Score for CVSSv2 and CVSSv3.1, are being added in the required Configuration Management Database (CMDB) component on an asset basis, which can be seen in the documentation of the project on GitHub[98]. The main contribution of the project is a scalable application to manage and process Environmental Scores for enterprise networks with thousands of IT assets[96]. The required data is automatically collected from source systems, which are CMDB and vulnerability scanners. Future work of both publications[96], [97], references the creation of ML algorithms to predict CVSSv3.1 components, without going into more detail about them.

The article [99] possesses a similar approach to [100], also using text mining on the descriptions of vulnerabilities from NVD. The approach differs in the fact, that features are engineered from the text to predict the CVSSv2 and the Weighted Impact Vulnerability Scoring System (WIVSS), which is introduced in section 3.3. The results of [99] show, that the three different ML approaches tested, can predict the CVSS score with nearly 80% accuracy and the WIVSS with a marginally higher accuracy. The paper concludes that the

[1] `https://github.com/DSecureMe`

models could be used to assist vulnerability scoring by cross-validation and even correction of scores[99]. In addition to the mentioned research possibilities in the paper, a prediction of missing CVSSv3.1 scores, based on the description of vulnerabilities and their CVSSv2 scores could be implemented, to aid the migration from CVSSv2 to CVSSv3 of historical vulnerabilities.

Using a different ML algorithm as used in [99], this migration capability, a ML algorithm that is capable of converting CVSSv2 to CVSSv3 scores, is proposed in [55]. It is based on VMC and the paper was written including parts of the research team behind the tool. To the authors of the article's knowledge this is the first publication attempting this. To aid the information that is present in the metric vector components of the CVSS, the top 50 keywords from the description field were included in the ML model. The research concluded after multiple tests, that different algorithms are best at predicting certain vectors. The classification for three metrics did not possess results with high precision, which will be addressed in future research. By creating an automated way of converting CVSSv2 scores to version 3 scores, current problems in VM could be solved, since a large number of vulnerabilities have not been converted to CVSSv3 as described in section 2.3.

The work in [101] considers the full CVSSv2 vector for creating attack graphs to model attacks in network environments. It proposes possibilities to assess probabilities of attack chains and their summarized impact with CVSSv2 metrics as input. This enables the authors to model complicated attack scenarios and assessing the risk that is imposed by the combination of vulnerabilities. The model does not propose a good means of automating these calculations, since the conditions between vulnerabilities need to be known. The paper [102] further improves on attack graphs and created a model to include business processes to them. This is done to derive risk to organizational processes from the attack graphs and to prioritize vulnerabilities on them. The model is created manually and is only a small prototype, since automation of the creation of the model was not in the scope of the work.

Opposed to the attack graphs, the paper [103] provides a formal risk calculation based on CVSSv2 and a stochastic model for the vulnerability lifecycle. It does not consider vulnerability chains but summarizes the risk of multiple vulnerabilities to get a complete risk score for one piece of software. It is based on calculations for risk presented in [104] and improves upon them. [105] also analyzes how CVSSv2 Base Scores can be aggregated for assessing severity of systems with multiple vulnerabilities.

Based on CVSSv2, the proposed improvements in [106] create a different Base Score using the Server Type and the OS Type of systems. This is achieved by changing the weights of the components of the CVSSv2 Base Score and adding a new measurement to it, the *Host Environment Score*. This additional scoring metric is dependent on the Server Type, which denotes if the vulnerable system is a server, business client

or a common client and the OS Type of the system. The ideas of the paper will be used in this thesis to create the model to infer the CVSSv3.1 Security Requirements, based on the information used in the paper and other information from the vulnerable system. This part of the model is described in section 5.1.

3.3 Other Rankings

The Vulnerability Rating and Scoring System (VRSS) introduced in [84] and refined in [83] proposes an improved vulnerability scoring system, that combines qualitative and quantitative methods. A qualitative rating is used for rating the impact, which results in High, Medium or Low. The quantitative measurement consists of the Exploitability Metrics (Access Vector, Access Complexity and Authentication). Version 1 of VRSS [84] suggests that a normal distribution represents better measurement for vulnerabilities and used IBM X-Force Severity Rating as an example for that. The authors of [83] find, that results of vulnerability scoring systems should be as discrete as possible, to distinguish the different nature of vulnerabilities. To aid this, the vulnerability type has been added as a measurement based on CWEs and a prioritization process.

The Weighted Impact Vulnerability Scoring System (WIVSS) is an alternative rating to CVSS, based on the metrics used in the CVSSv2 Base Score [107]. The aim of the work was to achieve higher diversity of values and better accuracy, which has both been achieved in the evaluation conducted. The main difference between the two scoring systems, is that the WIVSS is using weighted impact metrics, where confidentiality is considered more severe than integrity and integrity more than availability. CVSSv2 is using the same weight for the Impact Metrics Group. The work was further enhanced in [108], where an algorithm was developed to further enhance the weighting of the metrics, that are used. A new set of weights has been found, by defining simple rules and testing all possible values that fulfill these rules. It has been optimized to provide a better representation of severity classes and a distribution that is more tailored to the normal distribution. The weighting of the impact metrics is applied to the CVSS standard[4] in a different way, as the authors of the publications of WIVSS[107], [108] intended. The weights of the impact factors are incorporated into the Environmental Score, using CVSS-CR, CVSS-IR and CVSS-AR and are changed qualitatively with the possible categories as represented in section 2.3.1. To the knowledge of this thesis author, the WIVSS has not been adapted to CVSSv3 until today.

The paper [109] created a ML model consisting of combined recommenders to aid prioritization of vulnerabilities in a software development context. It uses metrics from CVSSv2 and publicly available data to create recommendations based on user profiles and general domain knowledge for prioritization of remediation of vulnerabilities. The authors evaluated the model on a small user group and found better error rates

to the preferred prioritization of the specific user than the CVSSv2 Environmental Score. Even though the system is tailored towards software developers, it could be used for a general model. To create better models for the algorithms used, a bigger dataset and more users would be required to remove bias from the model, according to the authors. Furthermore, the algorithm could benefit from the more mature CVSSv3 metrics. It is unclear why the CVSSv2 was chosen for as a data source for the algorithms, since the paper has been created years after the release of CVSSv3. The results of the work could also benefit from an automatically generated environmental score, on which the recommendations are further tailored towards user preference with the recommender.

The work of Lee *et al.* created a vulnerability prioritization based on the text of their descriptions in NVD[100]. To create this prioritization a text mining approach was chosen with the hypothesis that vulnerabilities that have a description more in common with other vulnerabilities possess a higher severity. This hypothesis is not proven in their work and it was concluded that expert-based performance evaluation is required to further improve their work. A common pattern in vulnerability management is that similar vulnerabilities are found which can be closed together on specific systems. The similarity algorithm could be used to group these vulnerabilities together to reduce the time spent on researching vulnerabilities, just to conclude they are the same or very similar.

The vulnerability optimization algorithm *VULCON* is proposed in [110]. It involves prioritization for vulnerabilities based on its CVSS severity (from the underlying vulnerability scanner), the age of the vulnerability, the number of months a specific vulnerability instance was present on a given system, the estimated time to mitigate a vulnerability and system specific measurements. The underlying algorithm optimizes the time-to-vulnerability remediation (TVR) and the total vulnerability exposure (TVE) of the underlying data. The calculation assigns each system a weight which indicates the criticality of a system. This work has to be done manually to use this functionality of the system. By incorporating an Environmental Score instead of the CVSS Base Score used by the vulnerability scanner, this could have been improved. The proposed methodology is strongly tailored towards a Cyber-Security Operations Center (CSOC) since it was tested on data from a CSOC. The information required for estimating the time to remediate a vulnerability is probably not known to organizations with low security maturity, that are trying to remediate their own vulnerabilities for the first time.

In the papers [80], [111] an analysis of the risk strategies of the commercial product Nexpose by Rapid7 has been performed. The detailed risk strategies employed by Rapid7 VM products were already described in section 2.4.6. Nexpose has been compared to other commercial vulnerability scanners and their risk strategies in the paper. Furthermore, other measurements like license models and CPE compatibility have

been collected over ten vulnerability scanners[111]. Unfortunately, at least one of these scanners has been already discontinued in 2020 by the owner of the product, Retina CS and Retina Network Security Scanner, which was renamed into BeyondTrust Enterprise Vulnerability Management[112]. After comparing the vulnerability scanners and their rating methods available, as well as their integrations, the paper [80] analyzed the different risk strategies employed by Nexpose on a real environment. The authors concluded that Nexpose can be used to develop a "cybersecurity risk analysis and management system"[80, p. 574] which was started in [111] using a context aware system to aid decision making in cybersecurity for organizations.

Another analysis of the capabilities and the detection quality of vulnerability scanners has been conducted in [14]. The differences in features of seven vulnerability scanners were analyzed, including the three market leaders Tenable, Qualys and Rapid7 according to [69]. Furthermore, a scan with all products was performed on a test environment. The conclusion of the work is that authenticated scans discover a more accurate result of the vulnerabilities present on a system, which was to be expected. Furthermore, a significant higher detection rate was observed on the tested Microsoft Windows systems over Linux, even when combining all scanners results.

Mokotoff *et al.* created a visualization to aid vulnerability prioritization in organizations[113]. After conducting research into challenges in VM, the paper uncovered four distinct challenges, which are "A Sisyphean Task" involving the number of threats and the available resources, "Finding the Right Net to Cast" when configuring and choosing tools, "Lack of Context" and "The Need for Multiple Sources" for insight into the vulnerabilities[113, pp. 32]. The "Lack of Context" challenge was being chosen to be worked on in the project. It is described that context to the vulnerability is required to make decisions about approaching vulnerabilities and three context information sources are needed the environment of the vulnerability, the value of the affected asset and the timeframe[113, p. 33]. To help with these context problems, a visualization dashboard was created from a source project visualizing Nessus scan results. The CVSS Base and Temporal Scores, the presence of Exploits according to Tenable Nessus and Metasploit and the presence of the vulnerability in the Qualys Top 10 Vulnerability List for each vulnerability was added to aid the described challenge[113, p. 34]. Unfortunately, there were no sources or more requirements on the environment of vulnerabilities examined in the thesis.

The commercial[114] Cyber Risk Scoring and Mitigation (CRISM) tool provides a possibility for scoring networks with vulnerabilities, based on network topologies and vulnerabilities[115]. With these the tool creates attack graphs to score cyber risk for an insurance usecase. The paper gradually shows the process of creating this risk rating and furthermore lists vulnerability prioritization as another possible usecase. The importance of assets is also incorporated in the generated score and the tool also tries to conduct real exploits

on the systems. The paper does not go into detail on how these graphs are generated and how the importance of assets is incorporated into the scoring.

The thesis of Allodi proposes a unique way of measuring the risk of vulnerabilities, exploiting the attackers' economic focus[116]. The author argues, that CVSS is not measuring risk, but in fact it is measuring severity, since the likelihood of an exploit is considered equally. By incorporating information of exploits in black markets and open source PoC repositories like EDB, a significant reduction in workload is possible, with very low effort as shown in section 7.3.2 of [116]. With this conclusion the work shows that the measurements employed by Tenable's VPR, as described in section 2.4.3, could be valid and significantly better than CVSS. Due to the intransparent collection and calculation of VPR and no scientific validation, this is not proven. The work[116] is limited in the fact, since it only uses the CVSS Base Score of vulnerabilities for its validations and calculations, justified with the argument, that the others are optional and not used by standards and common practices. The Temporal Score incorporates measurements for the existence of exploit code, CVSS-E, which could have altered the results of the thesis, by incorporating the added measurements into the CVSS prioritization.

Martinsson created in [117] an approach to use the CVSS Environmental Score, determined for single applications, and mapping them to International Organization for Standardization (ISO) 27002 controls using CWEs. The paper created a questionnaire, to easily assess the CVSSv3 Security Requirement Metric Group for an application. Guidance for estimating the CVSSv3 Modified Base Metric Group could not be given. To create a risk assessment for organizations, the top 50 CWEs determined by MITRE were associated to relevant security controls in the ISO 27002 in the chapters "8: Asset Management, 9: Access Control, 10: Cryptography, 12: Operations security, 13: Communication Security, and 14: System Acquisition, development and maintenance"[117, p. 38]. The association of vulnerabilities found in an application to the CWE were made using the association of CVEs to CWEs, which is available for most vulnerabilities as concluded in the paper. Since CWEs are less specific than CVEs they were preferred to create an easier mapping[117, p. 37]. The demonstrated example in the work shows that by using the method presented, a risk assessment for ISO security controls can be made from vulnerabilities and risk treatment can be inferred from the analysis conducted. Since the analysis in the paper is still performed manually and prioritized by the manually created CVSSv3 Environmental Score of the vulnerabilities, it cannot be easily performed for whole organizational networks. But the results of could benefit from an automatically inferred CVSS Environmental Metrics.

The Stakeholder-Specific Vulnerability Categorization (SSVC)[61] creates prioritization of vulnerabilities via decision trees. It builds upon different CVSS shortcomings, described by the authors in [71], [90] and

proposes a completely new model to rank and prioritize vulnerabilities incorporating risk management procedures. It is currently in version 2, published Apr. 2021[61] and was first introduced in Nov. 2019[118]. Their concept overcomes problems the authors have seen in CVSS. The main shift that is changed from known models is that their decision trees do not use quantitative risk measurements generated from qualitative inputs, as CVSS is doing it and completely work qualitative. The decision trees that are created for different audiences, that do vulnerability prioritization seem to be logically and it is possible to validate them. They incorporate mandatory information that is optional with the CVSSs Temporal and Environmental Metrics. Due to the completely new model and no possibility to automatically convert existing vulnerability information, their work is not easily automatable and therefore not usable for any security maturity. This is induced since some decisions require risk management to be implemented in the organizations using the model, which is not present in all organizations on low security maturities. Furthermore, since the decisions are partially system specific, they need to be answered for each vulnerability and each system, which can create an overhead of manual work, that cannot be automated completely. Therefore, a pre-prioritization could be beneficial to create a list of vulnerability instances that need to be prioritized further with the framework. The results of this thesis could be used to do exactly that and further improve the prioritization with SSVC. [61]

Other risk management principles, that can be applied as a framework to aid prioritization of vulnerability remediation actions is the Open FAIR, a trademark by The Open Group, Body of Knowledge[119] which is based on the Open FAIR Risk Analysis[120] and the Open FAIR Risk Taxonomy Standards[121]. The FAIR acronym stands for “Factor Analysis of Information Risk”. It provides a methodology to analyze risks, that is tailored towards IT security risk. It provides a model to quantify risks that result in a loss, by combining the frequency of loss events and the loss magnitude[122].

Another risk management approach using CVSSv3 is described in [123]. A semi-automatic model is described to create risk assessments for software development incorporating threats. It uses CVSS together with likelihood and stakeholder information to create a prioritization of threats to remediate. It adds a Threat Type measurement to distinguish between accidental or deliberate threats and adds likelihood of threats together with information on stakeholder to calculate risk.

Using pure risk management principles can aid vulnerability prioritization, but it is hard to automate and requires substantial effort to incorporate in an organization with a low security maturity. Therefore, it cannot be used to aid the vulnerability management principles and effectively improve security posture, by prioritizing the most severe vulnerabilities automatically.

3.4 Discussion

In the CVSS analysis papers many suggested, that exploit information is not covered reliably by NVD and EDB [87], even though exploit information is critical to prioritization of vulnerability as shown by [88], [89]. The article [89] even goes as far as telling, that the most commonly used part of the CVSS, the Base Score, cannot tell at all if a vulnerability can or will be exploited. The problems with the missing information with exploitability also created incentives to create the EPSS, described in section 2.4.1 and Tenable VPR, introduced in section 2.4.3.

A collection of other problems observed with CVSS s reported in [54]. The usage and misusage to the guidance given by the CVSS SIG as well as problems with the assignment of CVSS metrics is analyzed thoroughly by a lot of papers [6], [71], [86], [90], [91], [93], [116]. Furthermore, inherent problems with the CVSSv3 Temporal Score were observed by [60], which already caused changes for CVSSv4.

The requirement of the context information to vulnerabilities as well as the CVSS Environmental Metrics itself, are reported by [5], [91], [94], [113]. Difficulties in the usage and assignment of the Environmental Metrics are furthermore shown in [5], [6], [117]. To reduce these difficulties and ease the adoption by organizations [117] created a questionnaire to assign the CVSSv3 Security Requirement Metrics. The work surrounding the VMC [96], [97] also seems to try to ease the acceptance rate of CVSS Environmental Scores, by creating easy tools to handle vulnerability instances with added Environmental Metrics. To create a better context awareness of CVSSv2, [106] adds the type of systems and the OS to the Base Score. The effects of the usage of the CVSS Environmental Score is shown in [95]. By assigning the CVSSv2 Security Requirement Metrics, which are equivalent to their CVSSv3 counterparts, it was concluded that a significant change in the qualitative severity scoring of the analyzed vulnerabilities can be caused.

To create assistance in assigning CVSS scores and predict missing scores, [99] created a ML approach to predict parts of it from descriptions in the NVD. Since problems with the missing conversion of CVSSv2 to CVSSv3 are observed in multiple publications [45], [54], [55], the work in [55] created ML algorithms to create this migration, with additional input data from NVD.

Attack graphs to model the chaining of vulnerabilities and their impacts are examined in [101], [102]. Automation of these attack graphs was not performed in the scope of the papers. The CRISM tool[115], should incorporate attack graphs in its ratings, but no information about how they are created could be found. The aggregation of risk scores in various environments, is not only performed in commercial products, as described in section 2.4.6, but also [103]–[105] show different possibilities to create these aggregations.

To compare the scoring models presented in other papers, that create a scoring or a prioritization, with

scoring systems from section 2.4, a table similar to table 2.4 has been created as table 3.2. System without names, are represented with the source as a reference to the bibliography. To improve upon the CVSS, multiple models are created, which incorporate metrics for different prioritization scores [83], [84], [107]–[109]. These are the only systems that share the numeric scale with the CVSS. The other systems in the table, only create a prioritization and not a score for vulnerabilities that can be compared. The prioritization is mostly created by classifying the vulnerability instances into groups, that are either required or not required to be remediated.

	Measures	**Scale**	**Availability**	**Assignment Open and Verifiable**
VRSSv1	Severity of Vulnerabilities	Numeric 0-10 and Qualitative in 3 Classes	With CVSSv2 Base Score Metrics	Yes [84]
VRSSv2	Severity of Vulnerabilities	Numeric 0-10 and Qualitative in 3 Classes	With CVSSv2 Base Score Metrics and CWEs	Yes [83]
WIVSS	Severity of Vulnerabilities	Numeric 0-10	With CVSSv2 Base Score Metrics	Yes [107], [108]
[109]	Recommendations for Vulnerability Prioritization	No Scoring or Classification	With CVSSv2 Base Score Metrics, public Data and User Profile	Yes [109]
VULCON	Vulnerability Prioritization	No Scoring or Classification	With different Vulnerability and Vulnerability Instance Metrics	Yes [110]
CRISM	Vulnerability Prioritization	No Scoring or Classification	Unknown	No [115]
[116]	Vulnerability Prioritization based on Exploitation	No Scoring or Classification	Unknown, many proprietary Data Sources	Yes [116]
SSVC	Vulnerability Prioritization based on Risk	Decision Tree with Prioritization Decision as Outcome	Metrics need to be manually assigned to Vulnerability Instances	Yes [61], [118]

Table 3.2: Comparison of Vulnerability Scoring and Prioritization Systems of Related Work

Requirements for the created prioritization model of this thesis that can be obtained by the work analyzed is that the assignment of CVSS Security Requirements can be done per IT asset as performed in VMC [98] based on the research in [96], [97]. This enables to reduce the number of metrics needed to be inferred for assigning CVSS Environmental Metrics. Features of these assets to base this model on can be device type and OS as shown in [106]. In contrast to the modification of the CVSSv2 Base Score, the data will be incorporated into the environmental part of CVSS instead. The usage of CWEs is demonstrated in [83], [117]. By using CWEs it is possible to generalize vulnerabilities on their underlying weakness and simplify the model as shown in [117]. Problems with the usage of CWE are mentioned in [59], which should be examined further, when using CWEs extensively. The need to incorporate the importance of systems, has been identified in [94], [110], even though CVSS cannot do this properly[94]. This is another requirement that should be handled by the prioritization of the model created. Incorporating security measurements into the prioritization was concluded in [6]. The described requirements will be examined further in section 4.1.

4 Approach

To systematically approach the described challenges with prioritization of vulnerabilities, multiple steps are used to deduct a model to infer the CVSS Environmental Metrics and create a vulnerability prioritization. The model will consist of rules to deduct the metrics from different datasources. The starting point to create these rules is defining the requirements to the model from sources of the industry and literature. These requirements can then be processed into the features, that should be extracted from the input data to the model, by taking the input data into consideration to engineer features that can be inferred from the data sources. With this a data analysis can be performed and data sources available to organizations can then be assigned to the input data and a mapping to automated data retrieval and manual data retrieval can be inferred. If input data is already available to an organization in a format that can be processed and this data has not been assigned to the automatic data retrieval, due to various reasons, like availability of new tools, it can be easily incorporated into the model, even if the data sources were not used in the resulting model. Finally, rules for the resulting model are created and depending on the required data for these rules, they are assigned to steps of the model.

Figure 4.1 visualizes this process, starting with the requirements on the left and incorporating them with analyzed input data into the features. These features are mapped to the data, retrieved from the datasources and rules are defined to infer the CVSS Environmental Metrics. These rules are then assigned to steps of the model depending on the degree of automation possible, depending on the datasources and the security maturity required by the organization to have the data on hand.

Since vulnerability scanners are required in a VM program to identify vulnerabilities on the numerous different systems and software in an organization, they will be used as a primary datasource for the automatic part of the model. Different tools will be analyzed for information, that they provide in addition to vulnerabilities, to create the input data defined in the requirements. A short analysis of different vulnerability scanners and the data they can provide will be conducted, to be able to transfer the model easily to other vulnerability scanners, if the model proves to be beneficial to VM and vulnerability prioritization. After defining the requirements for data to use in the model, a set of real-world data will be used, which is available to the

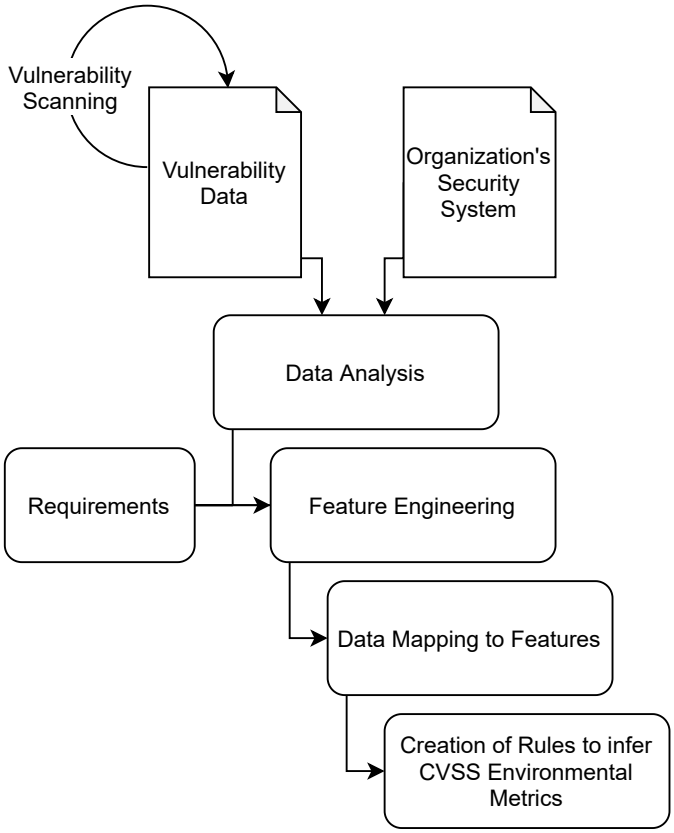


Figure 4.1: Visualization of the general Approach to generate Rules for the resulting Model

author in vulnerability data from Tenable Nessus scanners to derive specific input data for the model, which is available to real organizations performing vulnerability scanning. This directly enables easy implementation of the model to organizations, since vulnerability scanning data can be automatically gathered. Tenable Nessus was chosen due to the flexible Application Programming Interface (API), which was used to export the vulnerability data collected over multiple months into a database to make it searchable via automatic queries and to extend the data with a prototype of the resulting model. The vulnerability scanner was also chosen for the evaluation since the data in this scanner was available for analysis. Furthermore, Tenable was awarded to be largest vulnerability management company measured on market revenue[69] and their products build upon the popular vulnerability scanner as a data source. This makes the model's rules directly usable for the large number of organizations utilizing Tenable products.

With defined datasources that are automatically fulfilled with vulnerability scanning tools, the automated creation of an initial prioritization score for vulnerabilities can be defined. Using the available data from vulnerability scanners, rules and general scoring tables will be created to infer the environmental metrics from the collected data. These rules can be based on generalization and classification generated from the

features of the different systems found in an organizations IT network. The automatically collected data can be processed by rules to create this initial classification and further rules can infer the CVSS Environmental Score. In a next step of the model's maturity, data that is inaccurate or incorrectly inferred from the vulnerability scanner, can be modified based on other information sources that are available to the organization. These information sources will be defined to provide more granularity and accuracy for improving the categorization of IT systems. Furthermore, adapting the scoring table to organization's needs can be defined, to enable organizations to automatically infer Environmental Metrics tailored to organization requirements, based on automatically collected data, to reduce manual work required in creating security requirements for each vulnerability instance manually. The possibility of incorporating more rules, based on properties of vulnerabilities and the environment the vulnerabilities are found in, can be investigated too. By creating these more sophisticated rules, the environmental factors that are modifying the vulnerabilities impact on the organization are better represented.

To create a model that is future-proof and can be adapted easily to CVSSv4 when it releases, the announced changes to CVSS, introduced in section 2.3.3, are taken into account. A notable change introduced for CVSSv4 is the redesign of the Temporal Score. Since the research question of this thesis involves the Environmental Score, which includes information from the Temporal Score, this change could impact the resulting model. The CVSS Temporal Score Metrics involve information that is based on vulnerabilities itself and their metrics that change over time, not the environment the vulnerability is found in. Exploitability will stay a part of CVSSv4 as of now. The EPSS model creates a scoring to measure the likelihood of exploitability. It seems to deliver promising improvements to vulnerability prioritization based on exploit probability, as concluded in section 2.4.1. Therefore, an additional prioritization step of the model will be incorporating the EPSS score with the inferred CVSS score into a combined metric for prioritization and define how to use the CVSSv3 Environmental Score together with the EPSS score. Removing some metrics from the CVSSv3 Temporal Score, if they are available, will also be investigated. By doing that, problems with the changes to the CVSS Temporal Score can be avoided while still incorporating a good measurement for exploitability into the final prioritization result with EPSS.

From the different automatic and manual data sources the different stages of the model will be defined, from easily automated data retrieval to more complicated data retrieval to specific data, which needs to be incorporated manually to the model. By applying the model to the real dataset, the inferred CVSSv3 Environmental Scores and their quality improvement can be shown. A comparison on the quality improvement generated by the automatic steps of the model will be done, which should quantify the resulting increase of granularity of the prioritization score. Organizational specific improvements to the model, will further

enhance these improvements shown.

The evaluation of the model will compare the CVSS scores and the distribution of them on the dataset. By applying the model to all systems in the dataset and calculating the inferred CVSSv3 Environmental Scores for all vulnerability instances, the distribution of scores can be compared. This evaluation will be conducted on the automatic part of the model. The manual part of the model cannot be evaluated in the scope of this thesis, due to missing information that can be added to improve the model with organizational insights. Evaluation of the model needs to be conducted for the modifications done to customize the model to organizational requirements.

4.1 Requirements

To reason about different metrics of a system and infer them from a dynamic dataset, generalization is required. By generalizing and classifying similar systems into groups, metrics of the CVSSv3 Environmental Score can be defined based on these groups and be applied to the systems that fit into these groups. The definition of such groups and scoring based on them is performed in [106] and EPSS as introduced in section 2.4.1. EPSS, which is described in detail in [23], [62]–[64], is classifying vulnerabilities by using different properties of the systems the vulnerabilities apply to, to infer the exploitability scoring.

The groups and properties required for creating the model, can incorporate the type of device, the OS and security measurements taken on the systems. The device types should classify systems granular enough to infer security requirements from them, as well as not be too many, to enable a small footprint of the model. Having too many device types to classify systems into, it could happen that many of these classes compose the same requirements and modification of the default requirements to tailor the model to organization's needs gets too complicated, which can reduce the likelihood of adoption of the model as it is the case with the CVSS Environmental Score right now. To get a granular overview on the different roles a general purpose device, like a server in a network, is serving, the classification of devices into different roles can be performed. Purpose-built devices identified by their device type, for example printers, can easily be associated to their functions and roles in an organization. To assess general purpose devices too, a grouping by roles should define which functions a general purpose system is serving to the organization. By mapping one or more roles to a specific system, the roles can be used to infer more details on the CVSS Environmental Metrics. The OS of a device can also be utilized to infer device types and roles of a system. Furthermore, the OS is also used by other scoring systems like EPSS for prediction of exploitation. By knowing the OS of a specific IT asset, it is possible to infer some roles as well as create insight for vulnerability management

processes, built upon a vulnerability prioritization.

After assessing the risk of vulnerabilities on different system groups, which have specific security requirements in an organization, it is also important to factor in risk-reducing measurements, that reduce the possibility of exploiting vulnerabilities on a particular system, as concluded in [6]. These could be isolation of systems, which reduce the reachability via the network and therefore make exploitation of vulnerabilities more complicated. Other security measurements could be installing special software like endpoint protection to protect systems from exploitation of vulnerabilities. By factoring in these risk-reducing measurements in the ranking of vulnerabilities, vulnerabilities which are severe, but very hard to actually exploit, due to security measurements or hardening, do not get a very high ranking and are therefore not on top of the prioritization. Using this system instead of accepting the risk until a specific date, it is easily possible to re-evaluate the vulnerability again, when the vulnerability is on top of the prioritized list, when other vulnerabilities have been closed.

Security Measurements can also include restricting access to resources to qualified personnel only. Network segments, that are only used by administrators are unlikely to be impacted by vulnerabilities that require extensive interaction and exact preparation of exploitation. A good example for this are XSS vulnerabilities, which can be found on administration interfaces of IT infrastructure, where it is very unlikely for them to be ever exploited, since an attacker would need to have access to this interface to exploit the vulnerability and trick an administrator into interacting with a special crafted link. When segregated correctly, vulnerabilities which require active user interaction, can be scored even lower, when found in a management network.

Identification of network segments can also be used to improve scoring of vulnerabilities. Classification of IT assets and their roles can be achieved or improved based on network segments. A requirement for this classification is for the organization to have implemented network segmentation and planned the requirements of these segments carefully. By incorporating the segmentation plan into the prioritization, attack vectors can be reassessed. Furthermore, the device type and role identification can be aided when segments for specific device types are defined.

Another requirement for the model can be assessing different vulnerabilities, caused by specific weaknesses, identified by CWEs, differently depending on the organizations security system and re-evaluating the risk of them. A CWE describes different categories of weaknesses, which are in turn the problem behind a vulnerability. This re-evaluation should help to enrich the scoring with potential impacts resulting from the weaknesses on different system-categories and can be combined with other information about the systems to infer scoring rules. [117] shows that by addressing CWEs instead of CVEs a generalization of rules can be created, which was performed in the paper for mapping vulnerabilities with ISO controls.

The data required to fulfill the requirements stated in this section is listed below for reference. These data-requirements are assessed in the creation of the model and sources for them are identified. These sources can be either automatic, which enables complete automation of the model, or manual if for example special rules need to be defined which are derived from network topology and an organizations security system.

- Roles of IT systems to an organization
- Device Types to infer roles in the organization
- System Roles to infer roles in the organization
- Security Measurements which reduce the likelihood of exploitation and reduce risk of vulnerabilities
- Classification and properties of vulnerabilities

5 Data Collection and Analysis

To create a model, which infers the CVSSv3 Environmental Metrics to better prioritize vulnerabilities, data sources for the required input data to the model need to be identified and assessed for the kind and the quality of information that can be obtained. Since vulnerability management requires to use tools to identify vulnerabilities, these tools are assessed for data that goes beyond vulnerability information and can be used to fulfill the requirements of the model, as defined in section 4.1. This information that a vulnerability scanner collects, in addition to vulnerability instances on particular systems, can then be used to automatically infer information about the systems to improve prioritization. The improvement of prioritization will be done by automatically inferring a recommendation for the CVSSv3 Environmental Score.

To collect a knowledgebase of information and to infer the existence of vulnerabilities from it, Tenable Nessus reports informational vulnerabilities, which have a CVSS score of 0 and the severity *Info* associated, as presented in table 2.1[85]. Vulnerability instances created from these vulnerabilities do mostly not impose a real vulnerability with a risk to the organization, but instead give information about the scanned system. In VM programs these vulnerabilities are often discarded[21, p. 3-8]. In addition to informational vulnerabilities, Nessus also creates the *Nessus Knowledgebase* for each scanned IP address to share information between the plugins, that it uses to scan for vulnerabilities[124].

The vulnerability scanners by Greenbone Networks GmbH[1] and Tenable Nessus share the same initial architecture and codebase, the open-source variant of Nessus, which was made into OpenVAS[125]. Therefore, the products also report vulnerability instances that have the severity *Log* associated with them, like Nessus reporting vulnerabilities with the severity *Info* [126, p. 326]. Furthermore, the detection family of the vulnerabilities, which groups together similar detection plugins, also contains a *Product Detection* and a *Service Detection* family[127]. This shows that the informational data retrieved by Greenbone products, should be very similar to the data obtained by Nessus.

Other vulnerability scanners use different datastructures to store information about the systems scanned, that are not vulnerabilities itself. As already discussed in section 2.4.5, the vendor Qualys is comprising

[1] `https://www.greenbone.net/en/technology/`

its severity scoring for actual vulnerabilities into the detection validity with confirmed and potential vulnerabilities and the severity in five levels. In addition, there are also vulnerabilities that are assigned to the *Information Gathered* category, with three severity levels.[73] There is also a vulnerability group dedicated to informational vulnerabilities, that is named *Information gathering*[128]. Furthermore, the asset management capabilities of Qualys Global AssetView could be used to aid the creation of the datasource or provide another datasource to the detections itself in form of a dynamic asset inventory[129].

The VM products by Rapid7, presented briefly in section 2.4.6, incorporate no informational vulnerabilities as stated by [130] to help security personnel to focus on so-called *real* vulnerabilities. Information about systems scanned is stored, instead of informational vulnerabilities, in the asset database and can be used with dynamic asset groups. The available information per the documentation is among others OS names, service names, software name and host type, which contains four values *Bare metal*, *Hypervisor*, *Virtual machine* and *Unknown*.[131] Using the InsightVM API the full asset inventory, with information like CPEs for OS and software and services identified on the system, can be accessed.[132]

This short analysis shows that the most popular vulnerably scanners, as identified by vendors from [69], can all be used to retrieve information about systems scanned with them, which are not directly vulnerabilities. The datastructures used vary. Some vulnerability scanners create informational vulnerability instances in the same data structures as the real vulnerabilities. Others create asset inventory databases, which need to be accessed separately and at least Qualys seems to do both. The following chapter will focus on the information available in the dataset, collected with Tenable Nessus. An adaption of the model created from this dataset, should be possible since similar data is available in other VM products in different form.

As with the vulnerabilities on a particular system itself, more information is available when conducting authenticated, local scans as opposed to only network scans as concluded in [14]. These authenticated or local scans use management interfaces to log into machines and perform checks with them or are employed by agents installed on the system. Since not all organizations are conducting authenticated scans and it is not possible to scan every device authenticated, this extra information is not always available. The datasources for the model will benefit from performing these authenticated scans, but it will not be required. A baseline of information for the model can be obtained via network scanning, which is improved by enabling these more sophisticated scanning techniques that require access to the systems.

The following sections will go into detail on which information can be obtained by analyzing the vulnerability data retrieved by the scanner. The information covered is OS and device type in section 5.1 and different services and applications detected in section 5.2. Section 5.3 lists all groups of systems created, that will be used to infer CVSS Environmental Metrics in the next chapter.

5.1 Operating System and Device Type

Capturing the roles of an IT system in an organization can be started by examining the type of device and its OS. The OS is required by the vulnerability scanner to determine the existence of a vulnerability and its possible effects on the system. The type of devices is reported in Tenable Nessus with the plugin "Device Type" (54615) [2]. Some plugins have a textual output where more information on the detection is presented to the user. For this plugin this information is the type of device identified. In the dataset analyzed in this thesis, the device-types listed in table 5.1 could be inferred from the plugin's output. The percentage in the table shows how many devices with the device type detection were identified in each category. Not all systems found had the detection associated to them. Systems where the detection is missing, can be added later-on to the *unknown* category, but this is not done in table 5.1.

Device Type	Percentage
embedded	3.2%
firewall	1.6%
general-purpose	66.9%
hypervisor	0.8%
printer	0.2%
router	0.3%
switch	2.1%
webcam	0.8%
wireless-access-point	0.0%
unknown	24%

Table 5.1: Device Types retrieved from Dataset with Percentage of all Identifications

The device type is created from similar sources that also identify the OS, which is reported in the plugin "OS Identification" (11936)[133]. The data has been analyzed for each OS found with the corresponding device types by manually checking each association of OS and device type. Table 5.3 shows the device type assigned automatically by Nessus in the rows and the manually corrected device types in the columns. There were some entries, which did not have a device type associated, they can be seen in the second last

[2] Each plugin has a unique Tenable Nessus Plugin Identification Number (plugin ID), which is presented in brackets in this thesis. Information on plugins can be looked up, by adding it to the end of the following link, as presented with this particular plugin: `https://www.tenable.com/plugins/nessus/54615`

row. The systems with the OS identified and the device type missing are reported in the row *No device type assigned*. As seen in the table certain device types were always rated correct. These are *firewall*, *hypervisor*, *router*, *switch*, *webcam* and *wireless-access-point*. Since they are easier to decide from the reported OS than others like *embedded* and *general-purpose*, which both possessed many *unknown* types. The device types in the dataset were correct on 74.51% of all OS entries. When weighting the correct and incorrect guesses by the count of devices found, the percentage of correct device types decreases to 73.87%. The device types reported cannot be used to distinguish clients and server, since they are mostly identified as *general-purpose*, therefore another possibility to distinguish clients and servers is needed. When removing server and client operating systems manually, to simulate the usage of the device type only for devices with non-standard OS, the counted correct guesses are stable at 73.12% and the weighted correct guesses fall to 66.90%. To improve the correctness of the device type classification, the value *unknown* and missing values can be removed. This increases the counted correct percentage to 94.44% for both all OS and for the OS which are not servers or clients. The correct percentage, weighted by the number of systems is increased to 98.64% for all OS and to 91.71% for OSs which are not servers or clients.

For the data analyzed, it seems to be possible to use the device type to reliably identify devices of the types listed below. Device types which are not reliable or do not possess enough information to derive Environmental Metrics from them, like *embedded* or *general-purpose* need to be analyzed further for their roles.

- *firewall*
- *hypervisor*
- *printer*
- *router*
- *switch*
- *webcam*
- *wireless-access-point*

To distinguish the *general-purpose* group better and differentiate servers and clients, other measurements are required, since the "Device Type" plugin does not deliver this differentiation. Identification of servers and clients via network scans can be achieved by analyzing the OS to a specific degree. For Microsoft Windows OSs it seems obvious to split the OS versions into the Windows Server and the client Windows variants, like Windows XP, Vista, 7, 8, 8.1 and 10. But in terminal server applications Windows Servers can act as clients for multiple users.

Manual Device Type:	*embedded*	*general-purpose*	*hypervisor*	*firewall*	*router*	*switch*	*wireless-access-point*	*printer*	*webcam*
embedded	23	1				1			
general-purpose	2	82							
hypervisor			10						
firewall				3					
router					4				
switch						8			
wireless-access-point							2		
printer	1							4	
webcam									11
unknown	9	25		3		2			
No device type assigned		1				3	2	2	
Total number of OSs:	**35**	**109**	**10**	**6**	**4**	**14**	**4**	**6**	**11**

Table 5.3: Mapping Nessus Device Types to Manually assigned Device Types and count of OS

Tenable Nessus is reporting OSs as mentioned before with the "OS Identification" (11936) plugin[133], furthermore it is also noting the CPEs of a system in the plugin "Common Platform Enumeration (CPE)" (45590). The CPE is explained briefly in section 2.2.2. The plugin output of the CPE plugin distinguishes OS, hardware and application CPEs. When analyzing the dataset, it can be seen that 31.34% of the systems found have no OS assigned from the "OS Identification" plugin and 38.52% of the systems found have no CPEs assigned. Combined 30.59% of the systems have no OS and no CPE assigned. That shows that not every OS detection creates a CPE entry. About 0.75% of CPE entries are created without the OS detection plugin. With the analysis it can be concluded that the OS detection can be used for more systems than the CPE reporting. The model constructed will not go into detail on how to differentiate the OSs used for classification, since this can be implemented differently depending on the vulnerability scanner used and the information source of the vulnerability scanner used. The use of SWID, described in section 2.2.3, which should replace CPEs for software identification in SCAP version 2[26] is currently not to be found in Tenable Nessus.

Considering that the differentiation between servers and clients is not obvious from network scanning information alone, this information needs to be inferred by a combination of the OS and services detected on the system. For Microsoft Windows Systems, a client variant of the Windows OS can be used to identify client systems. Apple Mac OS can also be directly classified a client device type. Mobile OSs, like Android and Apple iOS, can also be used to identify clients. For Linux systems and Windows Server Systems this cannot be decided on the OS alone. To distinguish these two critical classes a rule is implemented, which checks if typical server applications are found, then the system can be classified a server, when they are not detected they can be classified as clients. Furthermore, it could be, that a Windows client OS is installed for some server tasks, such edge-case scenarios can be detected with searching for specific server roles on all OSs. Problematic systems need to be assessed manually if the information available in the vulnerability scanner is not enough.

A differentiation as performed in the paper [106] between "Common Clients" and "Business Hosts" is not done in the proposed model since all devices found on the network of an organization, which is assessed with a vulnerability scanner, directly create the attack surface of the organization. When non-business systems are connected to the network, they create a similar risk as if they were business systems, if not even more, when they are not properly secured and maintained. Therefore, the proposed model will add to the device types listed above the two types *server* and *client*. The *embedded* type seen in table 5.1 will be directly used to distinguish other systems from servers and clients. It will represent an embedded device that is not classified any further which is no server and not an embedded device captured by the other types.

5.2 Services, Applications and Roles

Identification of services is one of the main functions required for network vulnerability scanners. As mentioned briefly in section 2.1.1, the vulnerability scanner, when running in network scanning mode, first probes different ports and protocols to recognize running services. Then it tries to identify the software, including the version, to infer vulnerabilities in the software. Therefore, service identification is implemented very flexible in vulnerability scanners. Tenable Nessus in particular has one plugin, which reports a lot of different service types in the plugin output. It is called "Service Detection" (22964).[133]

The important services in the dataset, that could be identified with the "Service Detection" plugin are shown in table 5.4. The Service Identifier, listed in the first column is derived from the textual plugin output of the plugin. It denotes which service was found on the particular port and protocol. The column Type shows which information can be inferred from the presence of the detection. With some services a device type can be directly assigned to an IT asset identified by its IP address. These services are mostly found on specific device types only. Another value found in the Type column of the table is Server Role. These services are specific roles that a server can possesses, some of them can be critical for organizations. The table contains a recommendation which services can be critical to an organization. This recommendation can be adjusted to the organization's needs in later steps of the model. A server can have multiple roles, even though today's best practice is to only install few roles on a single server and utilize virtualization to host multiple servers on a single server hardware, for easier management and more security. Furthermore, Tenable has a whole plugin family dedicated to identify various services and products. It is called "Service detection", like the plugin analyzed before. It currently includes 522 different plugins for identifying various products and services[134]. After manually assessing the dataset for important service and product detections, table 5.5 was composed with the plugin names, identifiers and their corresponding services or products which they are able to detect. The table only contains services and products, that are not already identified in table 5.4. The Type column in table 5.5, provides insight into how the information of the detection can be used in the model, like in table 5.4. The table contains some more services, that generate more information for the model, as well as some product detections that can be critical for organizations. The table is by no means complete or exhaustive. The services in the table contain a small set of the services identified in the dataset, which are common. This table should be a starting point to implement and test the proposed model and can be improved by implementations of the proposed model.

A type which is new in table 5.5, is *Device Type and Role*, which denotes a service that not only identifies a device type but also a role. An example for that are the three listed server management interfaces. They

Service Identifier	Type	Remarks
FTP server	Server Role	File Server
HTTP proxy	Server Role	Web Server
IMAP server	Critical Server Role	E-Mail Server
LPD (Line Printer Daemon) server	Device Type	Printer
MariaDB server	Server Role	Database Server
MongoDB HTTP	Server Role	Database Server
MySQL server	Server Role	Database Server
POP3 server	Critical Server Role	E-Mail Server
rsync server	Server Role	File Server
SMTP server	Critical Server Role	E-Mail Server
SWAT server	Server Role	File Server
web server	Server Role	Web Server

Table 5.4: Nessus Service Identification with Plugin "Service Detection" (22964)

identify the IP address as a management interface for the hardware server and also serve as a management server role on the identified address. It can be critical if a vulnerability is found on these interfaces since all workloads on the hardware could be affected by the vulnerability. The interface detection plugins listed in the table are from HP and Dell, with their HP Integrated Lights-Out (iLO) and Dell Integrated Remote Access Controller (iDRAC). Furthermore, Intelligent Platform Management Interface (IPMI) an open specification for a server management interface by Intel is also listed in the table.

After detecting various services on different systems, their server roles can be derived from them. From the detection of a Lightweight Directory Access Protocol (LDAP) service running combined with a Microsoft Windows Server, it could be inferred that the server is running a Windows Server Active Directory (AD) as an AD Domain Controller (AD DC)[135, p. 226]. Unfortunately, Tenable Nessus does not provide a way to identify AD DCs via the network directly with a dedicated plugin. The existing plugins all require authenticated scans on the AD DC[136]. The AD DC is one of the most critical systems in an organization that relies on Microsoft Windows. Other means of detecting AD DCs can be Domain Name System (DNS) servers running on a Windows Server and information retrieved from the NetBIOS protocol[135, p. 227]. The "Windows NetBIOS / SMB Remote Host Information Disclosure" plugin (10150) contains the key *Domain Controllers* in the textual plugin output, when found on an AD DC. A fourth service that is required

Plugin ID	Plugin Name	Type	Remarks
20870	LDAP Server Detection	Critical Server Role	Authentication Server
43829	Kerberos Information Disclosure	Critical Server Role	Authentication Server
10150	Windows NetBIOS / SMB Remote Host Information Disclosure	Critical Server Role	Can contain *Domain Controllers* on AD DCs
108804	Microsoft Exchange Server Detection (Uncredentialed)	Critical Server Role	E-Mail Server
11002	DNS Server Detection	Critical Server Role	Service Critical for Network
10144	Microsoft SQL Server TCP/IP Listener Detection	Server Role	Database Server
10884	Network Time Protocol (NTP) Server Detection	Server Role	Network Service
26024	PostgreSQL Server Detection	Server Role	Database Server
25240	Samba Server Detection	Server Role	File Server
11424	WebDAV Detection	Server Role	File Server
38157	Microsoft SharePoint Server Detection	Server Role	File Server
21642	Session Initiation Protocol Detection	Server Role	Telephony Server
10663	DHCP Server Detection	Server Role	Network Service
22073	Oracle Database Detection	Server Role	Database Server
20285	HP Integrated Lights-Out (iLO) Detection	Device Type and Role	Server Management Interface
51185	Dell Integrated Remote Access Controller (iDRAC) Detection	Device Type and Role	Server Management Interface
72063	IPMI Versions Supported	Device Type and Role	Server Management Interface
63061	VMware vCenter Detect	Device Type and Role	Hypervisor Management Server
57396	VMware vSphere Detect	Device Type and Role	Hypervisor Server
110326	NetApp OnTAP Web Detection	Device Type and Role	Storage Server
10942	Citrix Server Detection	Critical Server Role	Desktop Virtualization Server
99168	Siemens S7 Protocol Support Detection	Device Type and Role	OT Device Type

Table 5.5: Nessus Service Identification Plugins

on AD DCs for operation is the Kerberos protocol, which can also be identified with a plugin. With these four indicators and the OS as a fifth check, AD DCs can be reliably identified. When testing this detection on the dataset, all AD DCs manually identified have LDAP, DNS and Kerberos Services detected. Furthermore, all systems that are not AD DC, do not have the NetBIOS entry mentioned, but some AD DCs do not have this entry either. That shows that the entry identifies an AD DC, but the absence of the entry does not tell, that the server is not an AD DC. In addition to that no server was detected that was a Windows system and had the Kerberos service running which was not an AD DC and all AD DCs had the service enabled. Some systems, which were no AD DCs, provided DNS or LDAP services, that means that these services cannot be used solely for AD DC identification. The dataset shows that the presence of the Kerberos service on a Windows OS can be used to reliably identify an AD DC. Only one anomaly was found in the dataset, one AD DC was identified as Debian OS, since it was scanned through a network tunnel. This system would not be identified as an AD DC, since the OS does not fit. This problem cannot be easily circumvented by rules since the OS is required to be identified correctly to infer this information reliably. This is another detection that needs to be improved manually to improve the accuracy of the environmental information derived from the automatic datasource.

With email being an important asset in business communication, the services and protocols used in it, can also be critical for organizations. The services for email are Internet Message Access Protocol (IMAP) and Post Office Protocol Version 3 (POP3) for retrieving emails in mailboxes from email servers and Simple Mail Transfer Protocol (SMTP) for receiving emails from external email servers and sending emails. These three services are easily identified as seen in table 5.4. In Microsoft environments the email server is often implemented as Microsoft Exchange Server. With the vulnerabilities that surfaced in 2021 around the Exchange Server[1], it is obvious that exploiting it can be a priority for attackers, since it is tightly integrated into the AD. Table 5.5 lists the plugin that is able to identify Microsoft Exchange Servers via network scans. This identification can directly be used to identify a server to possess the Microsoft Exchange Server Role. Another important communication method in organizations is internet telephony. Telephony via the computer network can be implemented using Session Initiation Protocol (SIP), which can also be used to transfer video and messages. The plugin to identify SIP servers is listed in table 5.5.

Web servers can be identified easily with the “Service Detection” plugin, as listed in table 5.4. Other identified server roles are file servers, which enable file transfers through the offered network service, like File Transfer Protocol (FTP). The *rsync* service is commonly used as an application via SSH, but it can also be used in service mode to remotely edit files[137]. *Samba* is a service for file transfer using the Server Message Block (SMB) protocol from Windows on other OSs. The Samba Web Administration Tool (SWAT)

tool can be used to easily configure *Samba* via a web interface and therefore also identifies a File Server. Webserver based file servers commonly use the service *WebDAV*, which can also be identified via network scans. Microsoft also has a file collaboration server called Microsoft SharePoint Server[138]. It can be used for creating websites for information sharing in a company. The *Server* variant is the on-premises installation of the product and is identified by a Nessus plugin which is listed in table 5.5. The identified services for servers that act as File Servers do not require to be used for file storage and file sharing only, they can also be used to alter the files of a website on a webserver and therefore the service is used for management purposes. But by identifying them and classifying them accordingly, security requirements can be inferred from this server role.

Different database server roles are listed in tables 5.4 and 5.5. Databases are used by different applications, which can range from web applications to critical business applications. The database services identified with the "Service Detection" Nessus Plugin, listed in table 5.4, are *MySQL*, *MariaDB* (a fork of *MySQL*) and *MongoDB*. Other databases like *PostgreSQL*, Oracle Database and Microsoft SQL Server are identified by other plugins, which are listed in table 5.5.

Network services required for operating networks are also identified by different plugins. The DNS Server was already mentioned in the detection of AD DCs, but it is also common to be on other servers. This role can be critical, since altering DNS results can critically affect the whole network, considering the traffic can be maliciously redirected to impact confidentiality, integrity and availability. Other network services that are listed in table 5.5 are Network Time Protocol (NTP) and Dynamic Host Configuration Protocol (DHCP). DHCP is important for availability of the network, but it can also impact the routing of network traffic by changing the default gateway. NTP can be used to cause all kinds of problems with applications that rely on accurate times.

Hypervisors are OSs used to install virtual machines on them. The commercial and commonly used hypervisor of VMware is vSphere[3] and its centralized management is vCenter[4]. Both are identified with Nessus plugins shown in table 5.5. Vulnerabilities in the hypervisor can impact all virtual machines on the hypervisor and therefore impact the whole server infrastructure of an organization quickly. Especially when the centralized management of the hypervisors is impacted. Therefore, it is crucial to classify hypervisors to rank vulnerabilities on them accordingly.

NetApp, a company which develops storage solutions, has a product called NetApp OnTAP[5]. Its web management interface is identified with a plugin listed in table 5.5. Vulnerabilities on the storage system for

[3] `https://www.vmware.com/products/vsphere.html`

[4] `https://www.vmware.com/products/vcenter-server.html`

[5] `https://www.netapp.com/data-management/ontap-data-management-software/`

databases or virtual machines, can also have a serious impact on the functionality of server infrastructure. This makes storage systems important assets for an organization which relies on its server infrastructure.

Some organizations are using desktop virtualization and terminal server solutions in their IT infrastructure. Citrix Systems, Inc. creates the most popular desktop virtualization platforms. Their remote desktop servers hosted as virtual desktop infrastructure are obtained with the plugin "Citrix Server Detection" (10942), as shown in table 5.5. These systems will be classified as clients since they are used to virtualize clients to run workloads centrally for users.

Devices in industrial applications are getting more and more interconnected in the industry. This interconnection is called Industry 4.0 and results in systems like ICS and in general OT devices that can be found in organization's networks by vulnerability scanners. Even though it is not advised to scan these devices[16], they can be misplaced in the wrong network segments. Alternatively, an effective network segmentation is completely missing in an organization. When these devices are identified in a network scan, they do not get scanned automatically by Tenable Nessus after identification with different plugins[139]. These devices are also added as a device type to distinguish them in vulnerability management and to assess the vulnerability prioritization on these devices.

A further refinement can be done by distinguishing different client roles. Not all clients are equal in their security requirements. An example for that is that not every client handles confidential data. Since client programs can be very different depending on the organization and can mostly only be identified with authenticated scans, only two examples for special clients are shown in table 5.6. The table shows two applications found as an example identified with local plugins on Microsoft Windows clients. These plugins can be used to identify the type of the client. An IT administration client is used to administrate IT components. It is likely that, when compromised, administrative access can be gained from these workstations. The other client role illustrated is an OT engineering client. This client role is used to administrate, configure and program ICS and OT devices. These workstations are likely to have unrestricted access to OT devices in the network and when compromised control of OT devices could be established. These client roles are listed as example rules to differentiate the types of clients found in an organization. The rules to reliably infer client roles need to be tailored towards the organization, which is discussed in section 6.2.

In addition to the service detection, CPE names of applications from the plugin "Common Platform Enumeration (CPE)" (45590) can be used to infer server roles. Since Nessus internally relies on the plugins instead of CPE names, all server roles were done with service detections without referring to the application CPE names reported with the CPE plugin. It is important to note, that the CPE identifies a specific software or hardware, while the service detection plugins detect a service, regardless of the product used to create

Plugin ID	Plugin Name	Client Role
81298	Siemens SIMATIC TIA Portal Detection	OT engineering client
64558	VMware vSphere Client Installed	IT administration client

Table 5.6: Nessus Plugins for Client Role identification

this service. This means, that for example the web server service detection cannot be mapped easily to all CPEs that identify webservers, since this list would be outdated as soon as a new webserver is released. To provide interoperability between products, the CPEs of the identifications are shown in table 5.7, which are available in the dataset. It is apparent, that not all service detections used can be associated CPE names. Only Service detections with CPE names associated directly are shown in the table. Using CPE can also cause new challenges due to various reasons. "CVE entries without CPE references, software products without assigned CPEs, typographical errors, and a lack of synchronization between both datasets"[39, pp. 26] are the ones analyzed in the work of Sanguino *et al.* The article gives insight into methods to overcome these challenges to work with CPEs in vulnerability analysis, which can be modified to be used to detect services and device types. This thesis will base the implementation of the model on the Nessus plugins and will therefore not go into detail on how these challenges can be solved.

Another requirement identified in section 4.1 is the identification of security measurements which can be used to alter the vulnerability prioritization. Security measurements that can be identified by vulnerability scanners are installed on systems. Endpoint protection programs can be installed directly on systems and reduce the likelihood of an exploit being successful. The various detections range from signatures of known bad files, to the detection of malicious network traffic to identification of the exploitation of certain weaknesses which cause vulnerabilities. When conducting authenticated scans, installed software is identified which is not supplying services to the network. With this identification the presence of an endpoint protection software can be performed. When identifying all systems in a network with endpoint protection installed, vulnerabilities on these systems can be ranked down in contrast to systems without endpoint protection installed. Plugins that identify installed endpoint protection software are "Check Point Endpoint Security SandBlast Agent Installed (Windows)" (139913) as an example for Windows based systems and "McAfee Endpoint Security Installed (macOS)" (141105) for Apples macOS. It is to be noted that endpoint protection alone is not able to prevent all exploitations of any vulnerability, but it can reduce the risk.

Plugin ID	Plugin Name	CPE
22964	Service Detection - Plugin Output *MariaDB server*	`cpe:/a:mariadb:mariadb`
22964	Service Detection - Plugin Output *MongoDB HTTP*	`cpe:/a:mongodb:mongodb`
22964	Service Detection - Plugin Output *MySQL server*	`cpe:/a:mysql:mysql`, `cpe:/a:mysql:community_server`
108804	Microsoft Exchange Server Detection (Uncredentialed)	`cpe:/a:microsoft:exchange_server`
10144	Microsoft SQL Server TCP/IP Listener Detection	`cpe:/a:microsoft:sql_server`
26024	PostgreSQL Server Detection	`cpe:/a:postgresql:postgresql`
25240	Samba Server Detection	`cpe:/a:samba:samba`
22073	Oracle Database Detection	`cpe:/a:oracle:database_server`
20285	HP Integrated Lights-Out (iLO) Detection	`cpe:/o:hp:integrated_lights-out`
63061	VMware vCenter Detect	`cpe:/a:vmware:vcenter_server`
57396	VMware vSphere Detect	`cpe:/a:vmware:esxi`, `cpe:/a:vmware:vmware_server`, `cpe:/o:vmware:esx_server`
110326	NetApp OnTAP Web Detection	`cpe:/o:netapp:data_ontap`
10942	Citrix Server Detection	`cpe:/a:citrix:xendesktop`, `cpe:/o:citrix:netscaler`
99168	Siemens S7 Protocol Support Detection	`cpe:/o:siemens:simatic_s7`

Table 5.7: Nessus Service Identification and CPE identifiers

5.3 Final Classification from Data Collection Phase

To create an improved classification of device types using the vulnerability scanner data in the dataset retrieved with Tenable Nessus, the following device types are inferred from the listed detections with specific plugins. As concluded in section 5.1 some classifications from the "Device Type" plugin (54615) can be directly used. With services described in section 5.2, these classifications can be improved and detection-gaps can be filled in. In total five device types solely rely on the "Device Type" plugin. Two more also have services associated that can also infer the device type in addition to the "Device Type" plugin. Two more device types were added from the services directly in addition to the *client* and *server* device types. The *embedded* device type from the "Device Type" plugin is only used if no other device type (except clients with non-client-only OS) fits the host.

In the following listing of the device types identified as a draft implementation for the proposed model, the Nessus plugin names in the listing are quoted, with their respective plugin ID in brackets. *Plugin Output* means, that the information is retrieved by parsing the textual output of the plugin. If *Plugin Output* is not stated, the existence of the detection of the listed plugin is assessed.

Device Types:

- *firewall* - Plugin Output "Device Type" (54615)
- *router* - Plugin Output "Device Type" (54615)
- *switch* - Plugin Output "Device Type" (54615)
- *webcam* - Plugin Output "Device Type" (54615)
- *wireless-access-point* - Plugin Output "Device Type" (54615)
- *printer*
  - "LPD (Line Printer Daemon) server" (22964)
  - Plugin Output "Device Type" (54615)
- *hypervisor*
  - "VMware vSphere Detect" (57396)
  - "VMware vCenter Detect" (63061)
  - Plugin Output "Device Type" (54615)
- *server-management-interface*
  - "HP Integrated Lights-Out (iLO) Detection" (20285)
  - "Dell Integrated Remote Access Controller (iDRAC) Detection" (51185)
  - "IPMI Versions Supported" (72063)

- *server-resources*
  - "NetApp OnTAP Web Detection" (110326)
- *OT-device*
  - "Siemens S7 Protocol Support Detection" (99168)
- *server*
  - Has one or multiple Server Roles from tables 5.4 and 5.5
- *client*
  - Microsoft Windows client OS
  - Mobile OS (Android, Apple iOS)
  - Apple Mac OS
  - "Citrix Server Detection" (10942)
  - Has no Server Roles, is not *embedded* and is not a special client with a client role
- *embedded* - Plugin Output "Device Type" (54615) and only if nothing else fits

As seen in the listing of the device types and the associated data sources, server roles are required to distinguish servers and clients. These server roles were defined in section 5.2. These roles can be used in addition to the device types to create a more granular estimation of the CVSS Environmental Score, by distinguishing servers with different server roles.

Server Roles:

- Database Server
  - *MariaDB server* - Plugin Output "Service Detection" (22964)
  - *MongoDB HTTP* - Plugin Output "Service Detection" (22964)
  - *MySQL server* - Plugin Output "Service Detection" (22964)
  - "Microsoft SQL Server TCP/IPListener Detection" (10144)
  - "PostgreSQL Server Detection" (26024)
  - "Oracle Database Detection" (22073)
- File Server
  - *FTP server* - Plugin Output "Service Detection" (22964)
  - *rsync server* - Plugin Output "Service Detection" (22964)
  - *SWAT server* - Plugin Output "Service Detection" (22964)
  - "Samba Server Detection" (25240)
  - "WebDAV Detection" (11424)
  - "Microsoft SharePoint Server Detection" (38157)

- Web Server
  - *HTTP proxy* - Plugin Output “Service Detection” (22964)
  - *HTTP server* - Plugin Output “Service Detection” (22964)
- E-Mail Server
  - *IMAP server* - Plugin Output “Service Detection” (22964)
  - *POP3 server* - Plugin Output “Service Detection” (22964)
  - *SMTP server* - Plugin Output “Service Detection” (22964)
  - “Microsoft Exchange Server Detection (Uncredentialed)” (108804)
- AD Domain Controller (AD DC) - Combination of Services as described in section 5.2
- Authentication Server
  - “LDAP Server Detection” (20870)
  - “Kerberos Information Disclosure” (43829)
- Network Service
  - “DNS Server Detection” (11002)
  - “Network Time Protocol (NTP) Server Detection” (10884)
  - “DHCP Server Detection” (10663)
- Telephony Service
  - “Session Initiation Protocol Detection” (21642)
- Desktop Virtualization Server
  - “Citrix Server Detection” (10942)

Similar to the server roles, the client roles identified in table 5.6 are listed below.

Client Roles:

- *OT-engineering-client*
  - “Siemens SIMATIC TIA Portal Detection” (81298)
- *IT-administration-client*
  - “VMware vSphere Client Installed” (64558)

To set the proposed classification into context it is compared to the SCAP Asset Identification standard. It is used to identify organization assets, including IT assets. It defines types for some IT asset types in addition to non IT asset types (*Person*, *Organization* and *Circuit*)[24, pp. 7]. The non IT asset types can be omitted for this model since they are not targeted by VM in IT systems. Some of the asset types, which are relevant for IT VM, can be mapped to the identified groups. *Database* can be mapped directly to the server role *Database Server*. *Computing Device*, as per the NIST specification[24, p. 3], can be mapped to hardware

devices and network devices. The network device types are *firewall*, *router*, *switch* and *wireless-access-point*. *Hypervisor*, *server-resources*, *server-management-interface* builds the hardware devices associated with servers. The *Network* asset type cannot be mapped directly to components that can have vulnerabilities, since it is an abstract construct and network components are not in the definition. *Software* is not directly addressed by the proposed groups, since the presence of software itself cannot be attacked only the services systems created and required by the software can be. The same can be said for *data* since it resides in a service or on a specific system. The *website* type and the more general *service* asset type can be mapped to the server roles. [24] This shows that the SCAP Asset Identification standard's categories are not granular enough for creating the rules for this model. The device types that can be identified in the dataset are more granular, fit better to the purpose and are therefore used instead for the model.

As defined in section 4.1, the requirements for the data which the model is based on can be partially fulfilled with the vulnerability scanner as an automatic data source. The roles of different IT systems to organizations are inferred by the device types identified in section 5.1 as well as the server roles described in section 5.2. In addition to that the possibility to create classes of clients is also briefly examined and an example for inferring them is drafted for two commonly found classes. The end of the section also gives insight how some security measurements like endpoint protection can be detected, depending on the scanning methods implemented in the vulnerability identification. The collected data in the dataset shows that most of the data requirements can be at least partially fulfilled with the detections found in a modern vulnerability scanner. In addition to the mentioned data, classification and properties of vulnerabilities itself also get reported by the vulnerability scanner, since this is the purpose of the scanner.

By adding more insight into the organizations security system, the presented classifications can be improved and missing classes, which can be unique to certain organizations, can be added easily. Furthermore, it is possible to aid the classification's accuracy by incorporating organizational information. Risk reducing measurements that do not rely on easily detectable technology, that are not detected by the vulnerability scanner, need to be manually added as inputs to the model via custom rules.

6 Inferring Environmental Metrics and Vulnerability Prioritization

This chapter will describe how the previously defined information sources can be used to infer the metrics creating the CVSSv3 Environmental Score. The full prioritization model for vulnerabilities created, based upon the CVSSv3 Environmental Scores of vulnerabilities is illustrated in figure 6.1 including the steps to infer the Environmental Metrics.

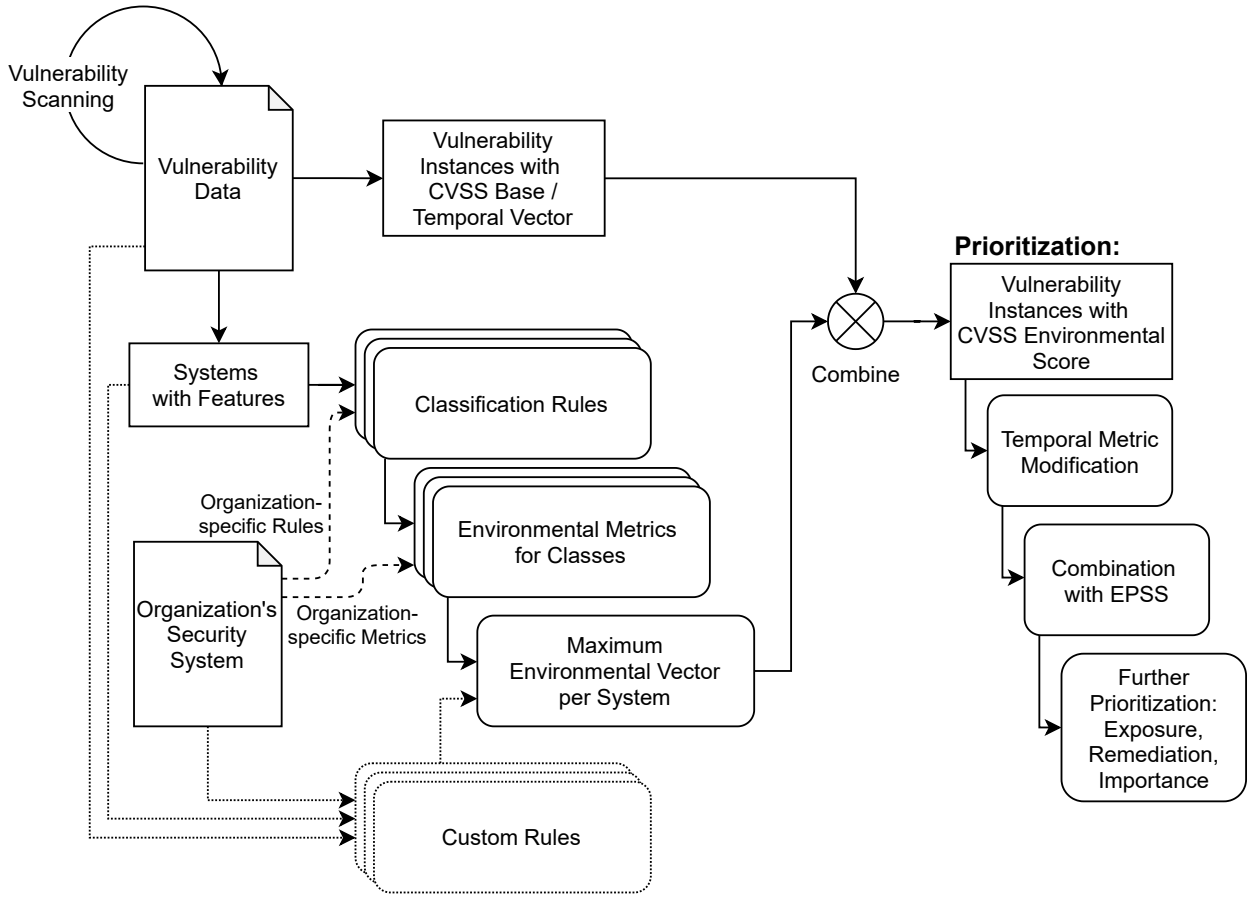


Figure 6.1: Vulnerability Prioritization Model

The first step of the model will define how the automatically collected data from the vulnerability scanner can be used to create a first estimation of the score based on generalized values. The figure illustrates this step in the left part. Starting with the path from vulnerability data over the systems, with the features extracted and classification rules, which were defined in section 5.3. An association between the classes and the CVSSv3 Environmental Metrics is performed in section 6.1 as well as the calculation of the maximum of the possible multiple Environmental Metrics per system according to the class of the system. The output of this part of the model can already be used to create a CVSSv3 Environmental Score for each vulnerability with CVSSv3 Base or Temporal Metrics, when combining the maximum Environmental Metrics with the metrics of the scores. An example of this association is that systems that were identified as a device type *switch* have a *low* CVSS Confidentiality Requirement Metric (CVSS-CR), a *medium* CVSS Integrity Requirement Metric (CVSS-IR) and a *high* CVSS Availability Requirement Metric (CVSS-AR). This association is done generalized for any organization and alters the CVSSv3 Environmental Score by reducing the impact ratings as described by the requirements specified in section 2.3.1.

This initial scoring can then be improved by organizations that require more detailed results by customizing the mapping, creating custom classes and improving the detection of classes based on the organization's environment. This first customization of the model is described in section 6.2. Figure 6.1 contains this second step as an improvement with the dashed lines, adding organization-specific rules to the general classification ruleset and an organization-specific class to metric mapping refining the general association presented in the first step of the model. A further customization of the model is discussed in section 6.3, which enables incorporating different security measurements to further differentiate scores. These customizations are represented as custom rules, using the data of vulnerabilities, the features of the systems identified and the organization's security system and can be incorporated into the model in the creation of the maximum CVSSv3 Environmental Vector created for the systems and vulnerabilities, as seen in figure 6.1 with the dotted lines. Since the customizations will still be performed using rules, the required work to create the modifications to the CVSS score is minimal in comparison to creating a custom score for each vulnerability instance or each system in an organization.

To be able to apply the model for prioritization and incorporate advanced other rating models into the prioritization of vulnerabilities, section 6.4 defines how to incorporate the generated CVSSv3 Environmental Score with the EPSS Score, presented in section 2.4.1. The section furthermore discusses different prioritization approaches, considering the challenges and changes in the CVSS, which were discussed in sections 2.3.3 and 3.2. The methods used to further enhance prioritization of vulnerabilities is shown in figure 6.1 on the right side, starting with the inferred CVSSv3 Environmental Score for each vulnerability

instance. Closing this chapter is a discussion in section 6.5 on how to incorporate other information into the prioritization, that is not assessed by the CVSSv3 Environmental Score, inferred with the model, and the EPSS. These other data sources are the exposure of systems, the importance of systems and the CVSS-RL as shown in the figure.

6.1 First Model Step - Inferring Environmental Score automatically

To create a first estimation of the CVSS Environmental Score, the device type and server roles, described in section 5.3, inferred directly from the vulnerability scanner can be used. By creating an estimation of the CVSS Security Requirement Metric Group based on this data, a first estimation of the Environmental Score can be created automatically. A similar approach to define Security Requirements is presented in [106]. The paper, based on CVSSv2 incorporates a new metric into a new calculation of the CVSSv2 Base Score, which represents the *Server Type* and *OS Type* of a system. The paper does not incorporate scores into the Environmental Score of CVSSv2, even though the *Server Type* and *OS Type* are not metrics of a vulnerability itself. To adhere to the spirit in the specification of CVSS, as presented in section 2.3.1, these metrics would better be implemented into the CVSSv3 Environmental Score.

The *OS Type* Metric introduced in [106] to an improved CVSSv2 Base Score, is based upon the argument: "For example, the security of windows operation systems is worse comparing with other operation systems in consequence of vulnerabilities. Other non-windows operation systems, like Linux, have the higher relative security than windows."[106, p. 353] Argumentation, why the Windows OS is worse than others is not given. The impact of the OS influencing the exploitation of vulnerabilities is examined in [23] and incorporated into the EPSS. Therefore, the vendor of the OS is not incorporated directly into the model for inferring the CVSS Environmental Score. The OS is only used to distinguish different device types. With the *Server Type* only having three values in [106], which are *Common Client*, *Business Host* and *Server Host*, the security requirements cannot completely be based upon them.

The device types identified in section 5.3 by using the vulnerability scanner as a datasource, will be mapped to the CVSSv3 Environmental Metrics in this section. This mapping should give a first rating of the environmental factors of a particular system, which can be used as a starting point for vulnerability prioritization for any organization. After assigning device types to specific metrics to modify vulnerability scoring, the different server roles are also assigned CVSSv3 Environmental Metrics to improve the scoring. In this first draft of the model, a whole system will be rated with one CVSS Environmental Metric Vector, which sets the CVSSv3 Environmental Metrics of vulnerability instances for this specific IT asset. In future improve-

Device Type	CVSS-CR	CVSS-IR	CVSS-AR
firewall	High (H)	High (H)	High (H)
router	Medium (M)	Medium (M)	High (H)
switch	Low (L)	Medium (M)	High (H)
webcam	Medium (M)	Low (L)	Medium (M)
wireless-access-point	Low (L)	Medium (M)	Medium (M)
printer	Medium (M)	Low (L)	Low (L)
hypervisor	Medium (M)	High (H)	High (H)
server-resources	High (H)	High (H)	High (H)
server-management-interface	Medium (M)	High (H)	High (H)
ot-device	Low (L)	Medium (M)	High (H)
ot-engineering-client	High (H)	High (H)	Medium (M)
it-administration-client	High (H)	High (H)	Medium (M)
client	Medium (M)	Medium (M)	Medium (M)
server	Not Defined (X)	Not Defined (X)	Not Defined (X)
embedded	Low (L)	Low (L)	Low (L)

Table 6.1: Default Values for Device Types to CVSSv3.1 Security Requirement Metric Group

ments of the model, multiple server roles present on a single server could be mapped differently to different vulnerability instances. Since it is not easy to decide, if a vulnerability affects certain server roles only or the whole system, this first scoring is not trying to fulfill this.

Table 6.1 lists the different device types and special client roles that can be identified and default mappings to the CVSSv3.1 Security Requirement Metric Group of the CVSSv3 Environmental Score, which is described in section 2.3.1. With the information contained in the device type only the Security Requirement could be estimated, since modifications of the Base Metric Group cannot be inferred from it directly. It can be seen that different device types can have different security requirements when compared to each other. Only one device type was not assigned a value directly since the security requirements can be assigned more granular using the server roles. As defined by the CVSSv3.1 specification[4], the value *High* is increasing the impact of the corresponding Impact Metric, while *Low* lowers it. This in turn increases or decreases the resulting score. The *Medium* value is neutral to the score.[4] Only common clients are assigned a *Medium* value for all Security Requirement, which results in not changing the CVSSv3 score.

Server Roles	CVSS-CR	CVSS-IR	CVSS-AR
Database Server	High (H)	High (H)	Medium (M)
File Server	High (H)	High (H)	Medium (M)
Web Server	High (H)	Medium (M)	Medium (M)
E-Mail Server	Medium (M)	Medium (M)	High (H)
AD DC	High (H)	High (H)	High (H)
Authentication Server	High (H)	High (H)	High (H)
Network Service	Low (L)	Medium (M)	High (H)
Telephony Service	Medium (M)	Low (L)	High (H)

Table 6.2: Default Values for Server Roles to CVSSv3.1 Security Requirement Metric Group

The different server roles are shown in table 6.2. Every server role is assigned a mapping for the CVSSv3.1 Security Requirement Metric Group. If a server possesses multiple server roles, the maximum of each value is used. When a server role with *Low-Medium-High* is combined with *Medium-High-Medium*, the result is *Medium-High-High* for the system. If the server device type gets a value associated, when organizations customize the default values, they present the lowest possible value for all server roles. If any Security Requirement Metric is set to *Medium* or *High*, server roles with *Low* or *Medium* are overridden to the higher value of the device type. No server role is assigned *Medium* for all values, which means that all servers identified will have changed CVSS Environmental Scores based on the Security Requirement Metrics. Server roles are always assigned to the whole system and not on a per port basis, since deciding if the vulnerability can be escalated to the whole system is not obvious from vulnerability information alone. It could be possible to create a model based on the CVSS Scope Metric (CVSS-S) metric, but vulnerabilities that can be chained would not be modeled correctly when the scope change occurs using another vulnerability.

By classifying the IT assets of an organization using automatic detections, based on network and authenticated scans performed with vulnerability scanners, it is possible to implement estimated CVSS Environmental Scores quickly in any organization. The only requirement to implement this model, is to have a vulnerability scanner, implement the classification of systems based on IP addresses into device types and perform the scans. The classification rules presented in this chapter are created as an example ruleset and are therefore not complete. They need to be supplemented with more detections to apply to a bigger number of organizations out of the box. The device types listed in table 6.1, as well as the server roles in table 6.2 can also be supplemented, if devices or roles are identified that are not covered.

6.2 Second Model Step - Improve Categorization with Organizational Information

The Security Requirement Metrics were assigned for universal use for any organization in section 6.1. Since CVSSv3 Environmental Metrics, should be tailored to the organization's environment, the scoring can be improved by customizing the values in tables 6.1 and 6.2. By assigning the Environmental Metrics based on device types and roles, the amount of manual work required to implement a VM prioritization based on CVSSv3 Environmental Scores is reduced. Organizations can have hundreds to thousands different devices and if the organization's asset and software inventory is not up to date or does not exist, the manual assignment can be impossible for all devices based on similarities, which would be captured in the asset and software inventory. Assigning the security scores only to different assets, can also be time consuming since many different assets can exist in an organization's network. By reducing the amount of labor required by grouping together similar assets, the assignment of custom metrics can be reduced to 13 device types, 8 server roles and 2 client roles. As mentioned in the previous step of the model it is not said that the listed device types and roles will fit any organization's needs, but it is unlikely that the number increases dramatically. Since the classes are defined on characteristics, that can be captured by vulnerability scanners automatically, the model can scale to all sizes of networks together with the VM program of the organization.

The model depends on inferring the classification into device types and roles correctly. To improve the automatic detection of these classes, rules can be created, based on security concepts of the organization network scanned. Network segmentation is used to segment parts of the network, limit communication paths between these segments and enforce security rules. Segments are often created for specific system types. This information can be used to improve the detection rules presented in the chapter 5. When rules are created that limit specific classes to certain network segments, identified by IP network addresses, the detections can be improved. Other rules based on network segments can incorporate assigning device types and roles based on OS in these segments. Network segmentation is also used to create security zones with different security requirements. These security requirements, inferred from the network segments, could be used to dynamically increase and decrease the metrics for systems in them. The impact of different attack paths in a network is modeled in multiple publications[101], [102], but due to their complexity they are mostly created manually. The CRISM tool presented in [115] seems to create these graphs automatically, but the algorithm for that is not public. By incorporating less complicated rules based on network segments only, the complexity in creating attack graphs for each vulnerability is avoided.

In an organization, different client roles are present, depending on the departments that the organization

has. These client roles can have differing security requirements. For example, a client used for accounting could have a higher confidentiality requirement than clients used for controlling ICS. The ICS control station in turn could have higher availability requirements than the accounting clients. By defining client roles, assessing their Security Requirement Metrics and defining rules, based on properties like OS, network segments and installed applications, these requirements can be added to the model to improve the CVSSv3 Environmental Score. This in turn can help to prioritize vulnerabilities based on their impact to different metrics on different classes of clients.

By starting with a general model that applies to most organizations and refining that with customizations, the model can be improved in small steps requiring less resources to prioritize vulnerabilities. Dynamically creating rules to infer the properties of IT assets of the organization enables the model to scale with new systems integrated easily. When starting with general rules and adding more specific rules later on, the model is improved quickly at the beginning and the overall accuracy is improved later on. Adding customized classes to the model benefits the accuracy of the inferred CVSS Security Requirement Metrics directly.

6.3 Third Model Step - Improve Scoring by Incorporating Security Measurements

Getting more insight into the properties of different parts of an organizational IT network can further benefit the prioritization of vulnerabilities. Rules to create CVSS Modified Base Metrics for further enhancing the CVSS Environmental Score can be created on this additional information. These rules are mapping different properties of specific classes identified to changes in the base metrics. To create such rules, information about the security system of an organization can be combined with properties of a vulnerability into conditions for the rules.

An example analysis of this concept will be conducted on the vulnerability `CVE-2020-11023`, which is a XSS vulnerability of the jQuery framework. The vulnerability can be found in different applications, ranging from general web applications to management interfaces found on server storage equipment. The vulnerability's CVSSv3 Base Score in the NVD is 6.1, which corresponds to the severity *Medium*. The CVSSv3 Vector of the vulnerability is `CVSS:3.1/AV:N/AC:L/PR:N/UI:R/S:C/C:L/I:L/A:N`.[140] When the vulnerability is identified on the management interface on server storage equipment in a management network, which has a security system in place that only permits access to administrators to the network segment, the vulnerability can become nearly impossible to exploit for an attacker. The attacker needs to convince an administrator that currently has access to the network segment to visit a malicious link to the

management interface web application and then the administrator needs to interact with the system, so the attacker can try to modify the webapplication. This modification could be used to exfiltrate data, cookies or credentials, depending on the structure of the management interface. According to the CVSSv3.1 specification [4] the CVSS Modified Attack Complexity Metric (CVSS-MAC) can be modified to *High* and depending on the segmentation the CVSS Modified Attack Vector Metric (CVSS-MAV) could be set from *Network* to *Adjacent Network*. This results in a reduction of the CVSSv3 score to 4.3. This behavior can be created as a rule, which identifies all vulnerabilities that have CVSS User Interaction Metric (CVSS-UI) in the management network, which in turn sets the CVSS-MAV from *Network* to *Adjacent Network* and the CVSS-MAC from *Low* to *High*. To create more granular rules, the rules can also identify the CWE for XSS (`CWE-79`) and only change the CVSS metrics based on this property. Other approaches utilizing the CWE are presented in [83], [117].

Other rules that can be built using the CVSS metrics and the security system, is increasing the CVSS-PR on systems that are only used by administrators for vulnerabilities that are exploited locally (CVSS Attack Vector Metric (CVSS-AV) is *Local*). If the system does not have any low privileged users and accounts, this fact can be used to further decrease the priority of the vulnerability instance, by setting the CVSS Modified Privileges Required Metric (CVSS-MPR) to *High*. Since a high privileged account is already required to exploit this vulnerability in this scenario, it could be, that if this account is overtaken by an attacker, no more exploitation is required and this vulnerability does not increase the attack surface of the system.

Organizations also employ security measures that proactively prevent threats, due to exploitation of vulnerabilities. Exploits can be prevented using techniques in endpoint protection focusing on signatures of commonly used exploit components and detection of exploitation of weaknesses. By doing that, the systems protected by these products get improved resiliency against exploitation and therefore an attack is harder to execute. If all systems would have similar endpoint protection capabilities this information cannot be used to prioritize vulnerabilities. Since it is not always possible to employ these programs on any system, due to requirements or compatibility, not all systems in a real organization will have endpoint protection installed. As mentioned in section 5.3, vulnerability scanners, can detect the presence of different endpoint protection suites, when performing authenticated scans. Using this information, the environmental metric for CVSS-MAC could be increased from *Low* to *High* to better differentiate vulnerabilities.

With the presented possibilities custom rules can be created, utilizing the security system, vulnerability properties and system information to further change the CVSSv3 Environmental Metrics, flexible but still in a scalable way. Other rules required by organizations that do not fit into the ideas shown can also be implemented at this stage of the model.

6.4 Prioritization with the Temporal Score and EPSS

As mentioned in section 2.3.1, the CVSS Remediation Level Metric (CVSS-RL) metric is impacting the CVSSv3 Temporal Score in decreasing the score, the more mature the remediation possibilities are for a vulnerability[4]. To measure the severity of a vulnerability this is understandable, but for remediation prioritization the metric should be the other way around. This reasoning is easily explained, since an official solution for a vulnerability should be easier to apply than a temporary fix or a workaround, which both need to be replaced with an official fix someday. For capturing the severity, this behavior can be correct, since vulnerabilities without solutions to remediate them could be more severe than vulnerabilities that could be fixed quickly. For CVSSv4 the CVSS SIG already approved to remove the metric from the score[45]. The paper [60] details how the CVSS Report Confidence Metric (CVSS-RC) is redundant and can be inferred from the other two CVSSv3 Temporal Metrics. This and some other concerns on the metric caused the CVSS SIG to decide to remove this other metric from the Temporal Score[45]. With the complete redesign of the CVSS Temporal Score with version 4, as the Threat Score and the more accurate scoring system EPSS in predicting vulnerabilities that are exploited, the CVSS Temporal Score seems not to be enough to solely base the prioritization of vulnerability remediation on. To create the full CVSSv3 score, including Base, Temporal and Environmental Scores, the metric should still be used to maintain compatibility with the official specification. For vulnerability remediation prioritization, it could be better to use the Base Score only with the Temporal Score element CVSS Exploit Code Maturity Metric (CVSS-E) and the Environmental Metrics and omit the metrics which will be removed in CVSSv4 by assigning the *Not Defined (X)* value to them. In addition to slightly modifying the Temporal Score Metrics, the CVSS-RL metric can be used to prioritize vulnerabilities with more mature remediations on vulnerabilities with less official fixes, when the prioritization of the vulnerabilities would be the same. By doing that remediation resources are used more efficiently when assuming that official fixes are easier to apply than less mature or official remediations.

The EPSS, as discussed in section 2.4.1, offers a better scoring system, when comparing the coverage of the exploit data used to train the EPSS model on, over CVSS. With the limitations mentioned in [23] considered, the EPSS can be used in addition to CVSS as illustrated in figure 6.2. By inferring the CVSS Environmental Score without the problematic Temporal Metrics and using it in combination to the EPSS an even better prioritization could be achieved compared to using the CVSS Base Score instead.

To combine EPSS and CVSS, the vulnerability prioritization model will use a threshold value for both, the CVSS score as well as the EPSS score, to classify vulnerabilities into four categories, with the following prioritization, as suggested by [64]. If the EPSS score is not available for a vulnerability a score needs to be

either assumed or inferred using data available in the vulnerability scanner, to prioritize these vulnerabilities. When assessing the Tenable Nessus dataset, of all distinct vulnerabilities discovered by distinct Nessus plugins, about 15.6% of all plugins with a severity that is not *Informational* have no CVE IDs associated. Since EPSS is only available for CVEs since 2017[63], the percentage of distinct vulnerabilities without EPSS information, due to old CVEs or missing CVEs is increased to 44.9% of all vulnerabilities found in the dataset. Therefore, the EPSS information is missing for nearly 45% of all the distinct vulnerabilities in the dataset. These vulnerabilities need to be prioritized, as if they impose an exploitable vulnerability, even though this is probably not true for most of them, as concluded in [62].

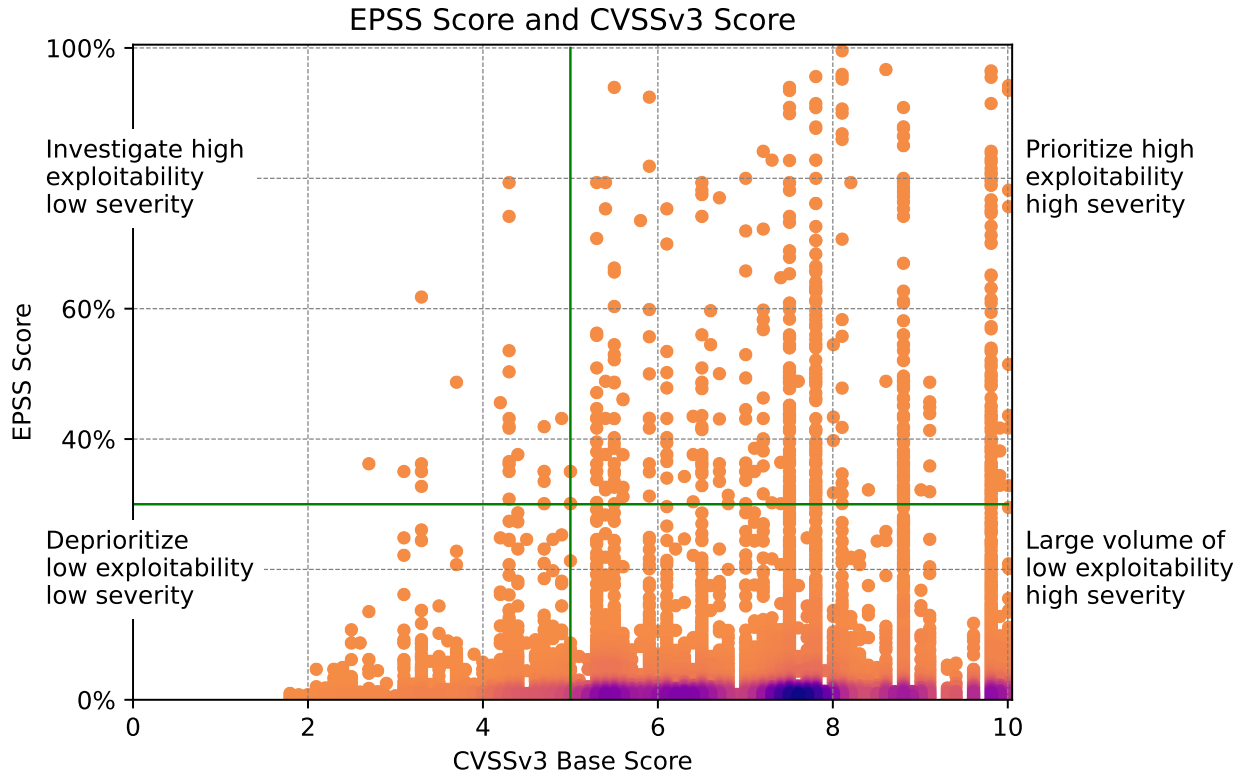


Figure 6.2: Correlation between EPSS and CVSSv3.1 scores retrieved on August 18, 2021 with CVSS data from NVD[52] and EPSS data from [65] including prioritization groups[64]

The guidance in implementing EPSS together with CVSS for VM given in Section 7 of [23] and in [64], can be used to create a categorization of all vulnerabilities into four groups. These groups can be seen in figure 6.2. The darker the color of the points in the figure, the more vulnerabilities are corresponding to the position. It is illustrated clearly, that the vast majority of vulnerabilities is rated low in exploitation probability and most vulnerabilities are located at CVSSv3 Scores between seven and eight, followed by

two other groups of vulnerabilities near 10 and between five and seven. To form the groups, cutoff values are used for both, the CVSS score and the EPSS score. The default cutoff values that are used for the model are CVSS score 5.0 (severity low or medium and higher as per table 2.1) and EPSS score 30%. The group with the highest prioritization is the group, where CVSS and EPSS is above their cutoff. The second prioritization group is, where the EPSS but not the CVSS is above the cutoff. The third group is the lower right quadrant, where the CVSS is above the cutoff but not the EPSS. And the last prioritization group is where both the CVSS and EPSS are below their cutoffs. When adding the vulnerabilities, where the EPSS score is unavailable, a fifth group is created. The prioritization group of vulnerabilities which is missing EPSS information, could be added at the third position, before vulnerabilities with EPSS below the cutoff. Furthermore, the group can be split at the CVSS cutoff and the vulnerabilities with the CVSS value below the cutoff can be moved further down. Since not all the vulnerability detections in Tenable Nessus contain CVSSv3 scores[57] and it is likely, that vulnerabilities from before 2017 do not all contain CVSSv3 scores, the CVSSv2 can be used in place of the CVSSv3 value, when the latter is missing. Alternatively, the work in [55] could be used to infer the missing CVSSv3 scores but testing this is out of the scope of this thesis.

The resulting final prioritization using EPSS and CVSS Scores (regardless of the type of score) used in the model will be:

1. Prioritization Group: CVSS $>$ *cvss-cutoff* and EPSS $>$ *epss-cutoff*
2. Prioritization Group: CVSS $\leq$ *cvss-cutoff* and EPSS $>$ *epss-cutoff*
3. Prioritization Group: CVSS $>$ *cvss-cutoff* and EPSS missing
4. Prioritization Group: CVSS $>$ *cvss-cutoff* and EPSS $\leq$ *epss-cutoff*
5. Prioritization Group: CVSS $\leq$ *cvss-cutoff* and EPSS missing
6. Prioritization Group: CVSS $>$ *cvss-cutoff* and EPSS $\leq$ *epss-cutoff*
7. Prioritization Group: CVSS $\leq$ *cvss-cutoff* and EPSS $\leq$ *epss-cutoff*

The prioritization groups listed above do not deliver a distinct order of vulnerabilities. Instead, it is only a grouping of vulnerabilities with similar properties and a prioritization of these groups. To get a distinct order of vulnerabilities inside the prioritization groups listed above, the EPSS score could be split further into groups by creating slices of, for example 5%. In these sub-groups the CVSS score can be used to prioritize each and every vulnerability from biggest to smallest. When there are equal scores of different vulnerabilities the absolute EPSS score can be used as a tiebreaker. Creating the slices in a logarithmic scale for vulnerabilities below the *epss-cutoff*, could further improve the number of vulnerabilities in each slice, since most of the vulnerabilities have EPSS scores below 2%, as can be seen in figure 6.3 and the vulnerabilities in the upper percentages are not as dense.

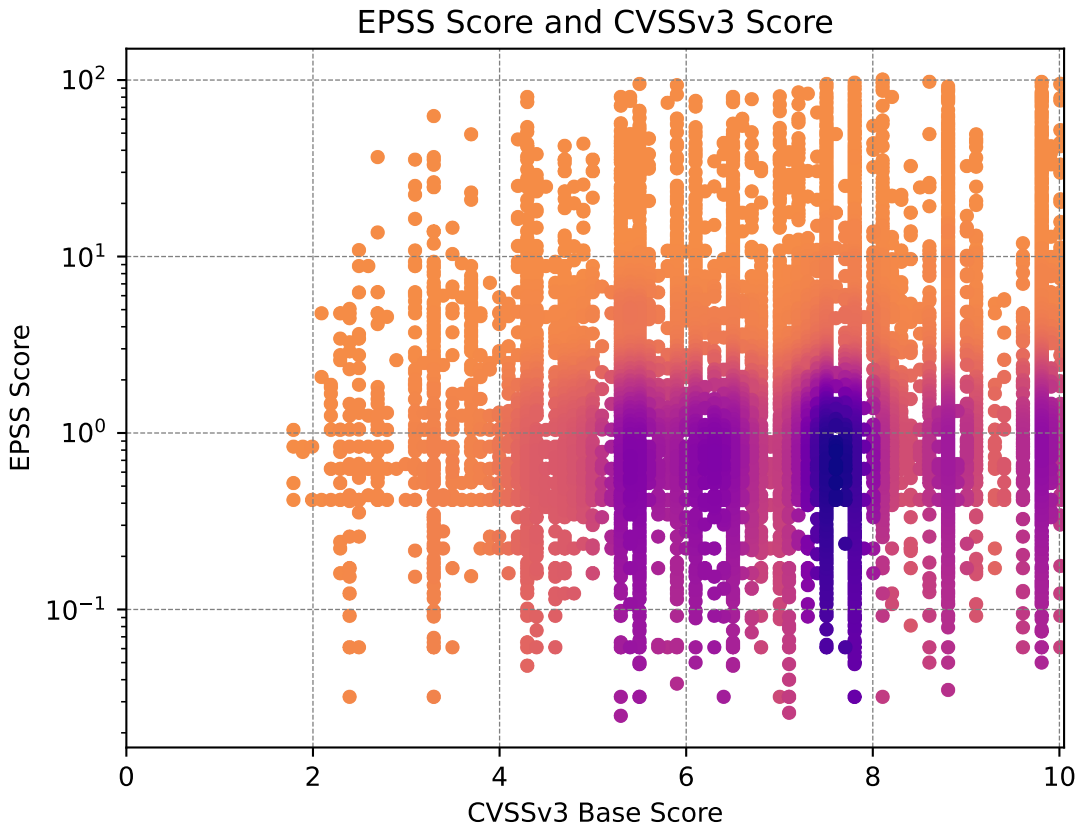


Figure 6.3: Correlation between EPSS and CVSSv3.1 scores retrieved on August 18, 2021 with CVSS data from NVD[52] and EPSS data from [65] in logarithmic scale

Since combination of the CVSS and EPSS scores is still an open topic of research, the presented methodology is a possibility to create exact rankings from the two scores. Testing of further combination algorithms for the two models needs to be touched upon in future work and can replace this step in the model.

6.5 Further prioritization

The prioritization of vulnerabilities on specific systems can further be improved, by incorporating information like the exposure of a vulnerable system, the CVSS-RL metric removed from the CVSS Temporal Score in section 6.4 and the overall importance of a system to an organization.

The exposure of a particular systems vulnerability can be easily determined by examining the positioning of the used vulnerability scanner relative to the system scanned. If the vulnerability scanner was able to identify a vulnerability via the internet, it is more exposed as if it was only identified internally. External

vulnerabilities could be harder to exploit due to firewall filtering, but if not, the attacks are likely to happen, as it was recently seen in [1]. It is to be mentioned that vulnerability scanners oftentimes have exceptions in firewalls to be able to conduct vulnerability scans and therefore should find vulnerabilities more easily than an attacker.

The CVSS-RL metric will be removed from the CVSSv4 specification[45] and will not be used in the CVSS score calculated for the prioritization as mentioned in section 6.4. It is added in this last step to creating the prioritization and vulnerabilities with better solutions to them are preferred to those with less official or less permanent fixes. This prioritization leads to more time spent implementing solutions supplied by the vendor, while solutions for other vulnerabilities are created in the meantime. Vulnerabilities without the metric assigned will be treated as *Unavailable*.

As a last step in this prioritization, internal information on system importance from either disaster recovery plans or business impact analysis can be mapped to specific IT assets and then be added to this model. If such information is not present in the organization this step can be omitted. Oftentimes not all systems are found in such documents, if that is the case, the classes of systems defined are appended with a class for all other systems. By having the approach described in this chapter, the systems in the same importance group get ordered by the model. That the system importance is not covered by the CVSS Environmental Score's Security Requirement Metrics is concluded in [94, p. 541], therefore it needs to be added additionally to the prioritization model.

These metrics could be ordered as shown in the listing below. If the data is present, it can be used to order the vulnerabilities, after applying the slice method for the six groups of vulnerabilities described in section 6.4, using the EPSS and CVSS scoring, for vulnerabilities with similar or the same CVSS score. The similarity of the CVSS Score to prefer these prioritization needs to be defined, depending on the accuracy desired of the CVSSv3 Environmental Score, to improve the prioritization as mentioned in [61, p. 5].

- Exposure of the system (first external, then internal)
- CVSS Remediation Level Metric (CVSS-RL) from more complete and official solution to workarounds to no solutions
- System importance based on disaster recovery plans or business impact analysis or similar documents

7 Evaluation

After implementing the model with the identification of device types and roles described in section 5.3 and applying the environmental scores from section 6.1 all vulnerabilities in the dataset were scored with the proposed model, without any organization specific customizations described in sections 6.2 and 6.3. The scoring methods applied to the dataset using the model, were derived from the CVSS vectors in version 2 and 3 reported directly by Tenable Nessus. Each detection from the vulnerability scanner with a severity that is not informational, as defined per table 2.1 was used. Therefore, all vulnerabilities have a CVSS score higher than 0. Using the CVE IDs listed with each Nessus plugin, the EPSS score was added to the data. If more than one CVE ID was listed with each plugin the maximum EPSS score found was used.

Table 7.1 shows the different scoring systems with the percentage of all vulnerability instances retrieved from the dataset that had the particular score associated. It can be seen that about 2.5% of the vulnerability instances had no CVSS score associated with them, since they have no CVSSv2 score. Every vulnerability since 2016 is scored using both, the CVSSv3 and CVSSv2[57]. Therefore, the availability of CVSSv3 Scores is lower than the availability of CVSSv2 Scores, because older vulnerabilities are still present. The rather low percentage of 9.53% of the vulnerability instances found in the dataset with EPSS scores is caused by the official repository of EPSS data only being available for CVE IDs starting in 2017[65]. The source of the CVSS feeds used by Tenable products, like Nessus, is the NVD and third-party intelligence feeds[85]. Availability of CVSS Temporal Scores is also rather low, with 25.98% for CVSSv2 and 21.07% for CVSSv3, which is expected due to general low availability of the metrics[65].

After applying the rules defined in section 5.3, the distribution of device types shows a rather high number of 78.19% of clients followed by 12.85% of servers, which is to be expected from the dataset. The table 7.2 lists the percentages for all device types identified in the dataset that have vulnerabilities. It is to be noted, that not all device types are present in the table, since not all devices have vulnerabilities identified in the dataset. The two special client roles identified in table 5.6, were implemented with the same logic as the server roles and are therefore not listed in this table. The score applied through the model's implementation was not altered by this minor modification.

Scoring System	Percentage of Rows
EPSS	9.53%
CVSSv2 Base Score	98.49%
CVSSv2 Temporal Score	25.98%
CVSSv3 Base Score	71.40%
CVSSv3 Temporal Score	21.07%

Table 7.1: Percentage of Vulnerability Instances retrieved from Dataset with Score from Scoring Systems

Device Type	Percent of Systems
client	78.19%
embedded	1.60%
firewall	0.67%
hypervisor	0.90%
printer	0.23%
router	0.28%
server	12.85%
server-management-interface	1.55%
server-resources	0.52%
switch	2.32%
webcam	0.85%
wireless-access-point	0.05%

Table 7.2: Percentage of Systems with Device Types associated retrieved from Dataset

Role	Percent of Roles	Percent of Systems
AD DC	1.86%	0.46%
Authentication Server	4.64%	1.16%
Database Server	21.44%	5.36%
Desktop Virtualization Server	44.23%	11.05%
E-Mail Server	7.22%	1.80%
File Server	14.02%	3.50%
IT administration client	0.52%	0.13%
Network Service	21.65%	5.41%
Telephony Service	3.20%	0.80%
Web Server	1.96%	0.49%

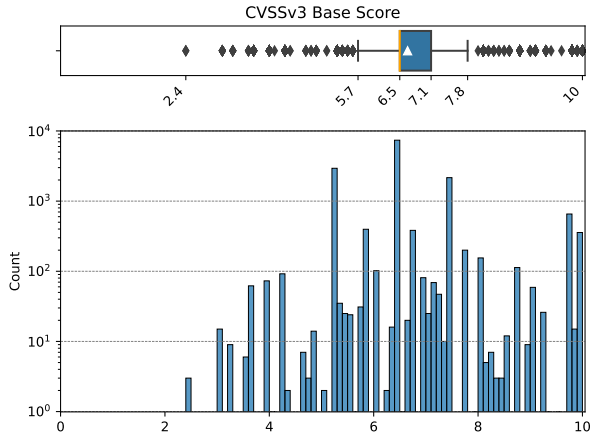
Table 7.3: Percentage of Roles associated to Devices retrieved from Dataset

The systems in the dataset were assigned with client and server roles listed in table 6.2, derived from the rules defined in section 5.3. 24.98% of all systems in the dataset were assigned a role. Table 7.3 lists all roles identified with vulnerabilities and the corresponding percentages of systems in total and the percentage normalized on the percentage of systems that have roles. It is to be noted, that the sum of the Percent of Roles column is more than 100%, since a system can possess multiple roles as discussed in section 5.2.

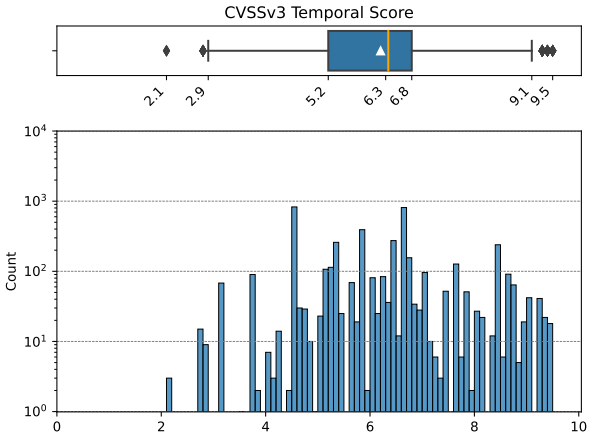
To evaluate the model, the scoring has been applied, with the rules described in tables 6.1 and 6.2, to each vulnerability instance in the dataset. The CVSSv3 Environmental Score was calculated with the Security Requirements added from the rules. The CVSSv3 Base Vectors were supplemented with the CVSSv3 Temporal Vectors, where it was available and the CVSSv3 Security Requirement Metrics were added to the vectors for each vulnerability instance. Vulnerabilities without CVSSv3 Scores, were not considered for this analysis. Figure 7.1 visualizes the count of vulnerability instances in the dataset with each particular score. The median in the box plots is drawn orange, to distinguish it from the quartiles. (In figure 7.1a it is at the lower quartile at 6.5.) Each bar in the histogram represents a 0.1 increment of the score, with that all possible scores are represented. A logarithmic scale was chosen for the histograms count axis, due to the high number of vulnerabilities in the range of 6.5, particularly in the CVSSv3 Base Score. Figure 7.1b displays the CVSSv3 Temporal Score and is filtered to include only vulnerability instances, which have CVSSv3 Temporal Metrics associated with them. It is illustrated that the Temporal Score reduces the Base Scores, when comparing the figures since most vulnerabilities have official fixes and no exploits associated

with them. It can be seen in the box plot of figure 7.1a, that most vulnerability instances of the CVSSv3 Base Score are in the range from 5.7 and 7.8. The outliers were calculated using the 1.5 times the interquartile ranges rule for box plot creation[141, pp. 236]. The CVSSv3 Environmental Score inferred with the model is shown in figure 7.1c. Figure 7.1d included the modification for the CVSSv3 Temporal Score as described in section 6.4 and displays the CVSSv3 Environmental Score used for ordering, calculated with the proposed model. The histograms show an improvement in the diversity of the values generated with the CVSSv3 which makes ordering of the values easier. It can also be seen from the box plots, that the values of the Environmental Scores are slightly increased in their values compared to the Base Score. When comparing the mean values of the four scores, this increase in values is illustrated clearly. The CVSSv3 Base Score has a mean of 6.65, the Temporal Score has 6.20 without missing values and 6.41 with missing values replaced with the Base Score. The models inferred Environmental Score has got a mean of 6.67 and the modified Environmental Score without CVSS-RC and CVSS-RL has 6.77. The mean is annotated in the boxplots in figure 7.1 with a triangle. When comparing the standard deviation, to quantify the bigger spread of the values[141, pp. 96], the CVSSv3 Base Score gets a value of 1.19, the Temporal Score reaches 1.41 without missing values and 1.12 with missing values. The inferred CVSSv3 Environmental Scores are able to increase the standard deviation of the values to 1.26 with the original Temporal Score and 1.28 with the modified Temporal Score. The two visualized Environmental Score variants in figures 7.1c and 7.1d show only small differences, even though the CVSSv3 Temporal Score has been modified between them. This can be explained by the low percentage of vulnerabilities having a CVSSv3 Temporal Score associated with them as seen in table 7.1.

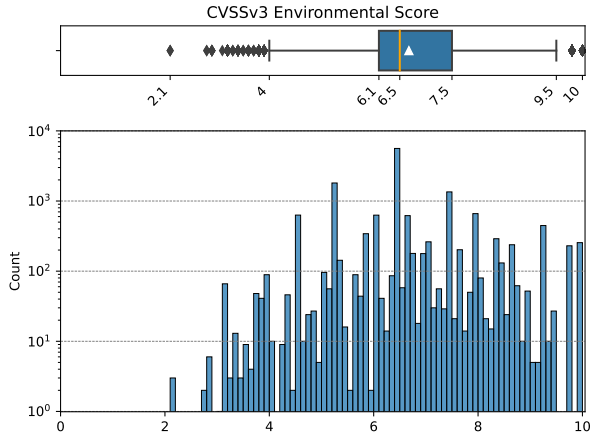
Due to the low number of EPSS Scores available in the dataset, as seen in table 7.1, it cannot be used widely for the prioritization. More than 90% of the vulnerability instances in the dataset could not be associated to an EPSS score due to missing CVE IDs or missing EPSS scores due to CVEs before 2017, as described in section 6.4. For the nearly 10% of vulnerability instances that could be associated to an EPSS Score, figure 7.2 was created. It can be seen that the majority of vulnerabilities is below the EPSS and above the CVSS threshold, defined in section 6.4. Using the ordering of the prioritization methods suggested by the model, all vulnerabilities with high EPSS scores are prioritized before the large number of vulnerabilities without EPSS scores. As a last prioritization group the vulnerabilities with low EPSS scores are assessed afterwards. The suggested model will help to prioritize the vulnerabilities in these last two groups, since EPSS cannot be used on them.



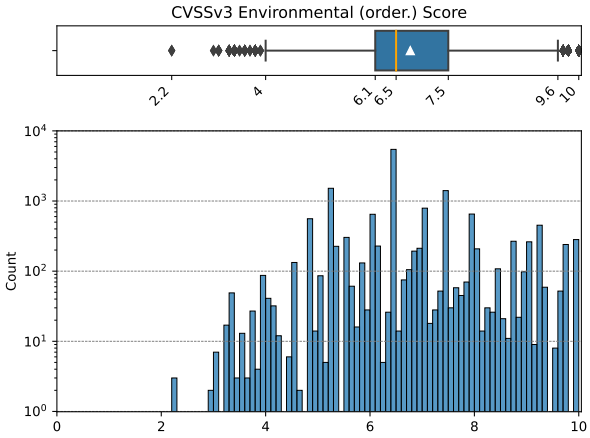
(a) CVSSv3 Base Score



(b) CVSSv3 Temporal Score (not including missing values)



(c) CVSSv3 Environmental Score



(d) CVSSv3 Environmental Score (with modified Temporal Score)

Figure 7.1: Box Plots and Histograms of CVSSv3 Scores of Vulnerability Instances retrieved from the Dataset

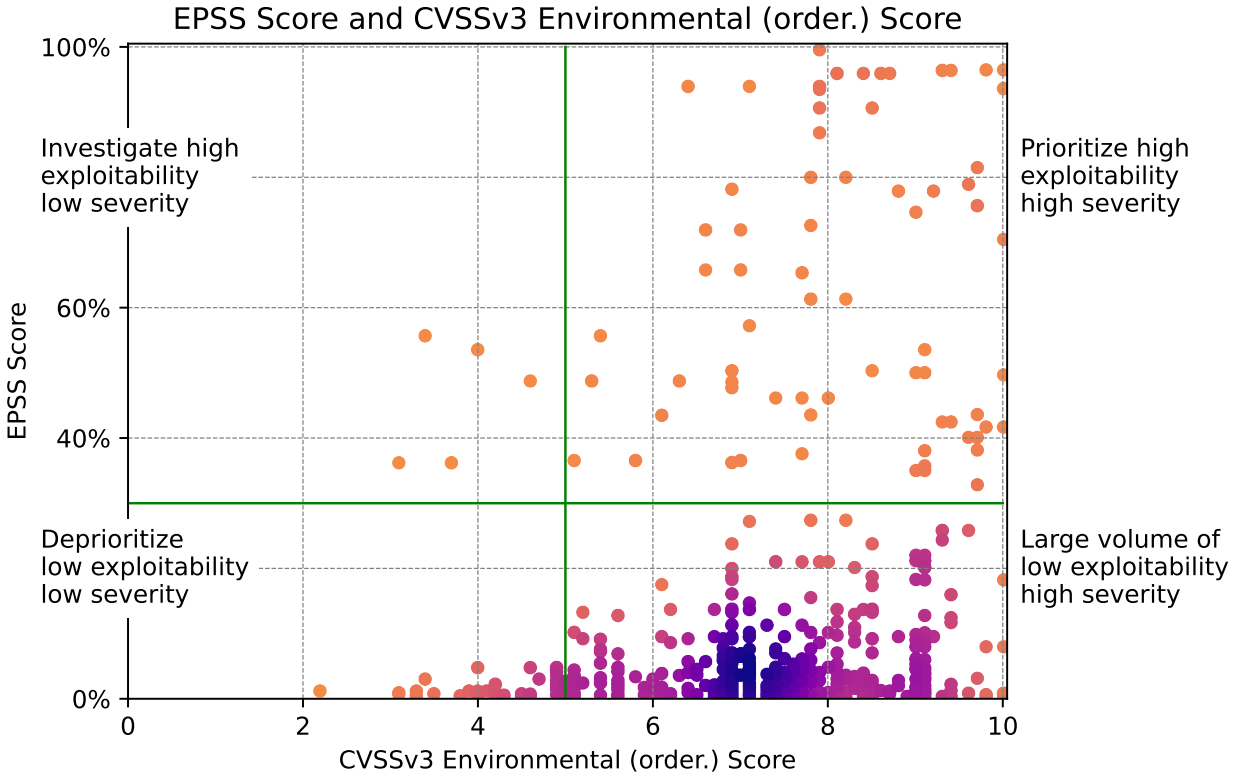


Figure 7.2: CVSSv3 Environmental Score (with modified Temporal Score) in relation to EPSS

8 Conclusion

This thesis presents a new approach for inferring the CVSSv3 Environmental Score from data available in vulnerability scanners and other organization specific data sources as well as vulnerability prioritization based upon this score. The model to create this information, was developed by studying the work done in various other improvements to CVSS and scoring models based upon or trying to replace the CVSS. Based upon data retrieved from a dataset, created with the vulnerability scanner Tenable Nessus, rules were defined that build a prototype of this model. To answer the research question, "Which data is required to estimate the Environmental Score of the CVSS?", the data requirements are discussed in chapter 4. The required data identified is information about systems to infer their roles in an organization, which requires device types and roles based on services of the systems to be identified. Furthermore, different security measurements taken on systems and in the whole organization and other risk reducing measurements can be used to improve the estimation. By also using information of the particular vulnerabilities itself, like classifications and common properties of the vulnerabilities, the estimation can be enhanced further. Requirements for the model were identified and data sources were examined for the model. A part of the data that can be deduced automatically from vulnerability scanners which is analyzed in depth in chapter 5.

By defining a model to infer the CVSS Environmental Score Metrics from data that is commonly available to organizations, which employ vulnerability scanners in their VM program, the proposed model can be used by organizations of any maturity. The research question "What data sources can be used to acquire this data and which of them can be deduced from available data in vulnerability scanners?" is discussed initially in chapter 4 and answered in detail in chapters 5 and 6. The initial rules, based on vulnerability scanners as an automatic datasource, presented in section 6.1, can be used by anyone after conducting a vulnerability scan. The proposed rules are building a foundation and show the concept on how to create the scoring model and will need to be complemented by creating more device types and roles, as shown in section 5.3, to create a holistic model that fits any organization. The rules in this thesis are created on Tenable Nessus data and are to be used with Nessus because of its popularity. To use the model with other vulnerability scanners, the data retrieval, which was performed in chapter 5, will need to be repeated for these tools. More datasources that

can be used for the model are defined in the later sections 6.2 and 6.3. They currently need to be manually implemented into the model.

To improve upon the initial rating using a model of generic CVSS Security Requirement Metrics, section 6.2 defines possibilities to alter the scoring created by the model to be able to customize it to organizations policies and requirements. By also defining manual datasources that can be used to improve the automatic model, the scoring accuracy is optionally improved upon by organizations utilizing them. While altering the definition and metrics created, which are based on groups of systems, the amount of labor is kept low, while still allowing a high degree of flexibility to customize the results of the model. Section 6.3 is using the security system of an organization as another manual datasource and defines how to create rules from risk reducing measurements in place to further improve the scoring and create more accurate metrics for use in the CVSS Environmental Score.

By defining the proposed three step model, which is based on automatically retrieved data in the first step and manual data used to supplement the model in the second and third step, the research question "How is it possible to design a model that infers the Environmental Score of CVSS, that combines automatic and manual data." is answered thoroughly. The defined automatic model to create a scoring purely from vulnerability scanning results was furthermore implemented and an evaluation of it was conducted in chapter 7. This evaluation showed, that the CVSS Environmental Score inferred by the proposed model creates vulnerability scores, that are more diverse and therefore it is easier to order them for prioritization. The clustering of CVSSv3 scores is reduced and the deviation of the scores sees an increase by applying the CVSS Security Requirement Metrics automatically to the available Base and Temporal Scores using the model. By increasing the possibilities of different scores, based upon automatically retrieved CVSS Security Requirements of each system, it is easier to decide if a vulnerability possesses a greater impact than another, by comparing the more diverse numerical scores.

The model creates prioritization based upon the existing standard scoring model for vulnerabilities CVSS. To create a more sophisticated model, further prioritization methods are added to improve upon shortcomings presented in literature of CVSS. One of these improvements is the EPSS which is based upon problems in the CVSSv3 Temporal Score Metrics. The EPSS is trying to improve upon the idea of the CVSS Temporal Score and is presented in detail in section 2.4.1. By conducting a combination of the inferred CVSS Environmental Score and the EPSS Score, the prioritization of vulnerabilities can be further improved. A method to combine these two models and solve other shortcomings of the CVSSv3 Temporal Score is presented in section 6.4. The methods presented in this section also take changes that will be implemented with CVSSv4 into consideration, which are described in detail in section 2.3.3. More considerations in

prioritization of vulnerabilities are presented in section 6.5. They involve including system importance into the ranking, as well as exposure and the maturity of the fix for a particular vulnerability. By adding these prioritization mechanisms to the VM, the amount of resources available can be used more effectively, by closing vulnerabilities that are most likely to be exploited and are the most severe to an organization.

Limitations of the model are currently a possible bias towards the data, that is present in the dataset, analyzed in chapter 5. The dataset was comprised of vulnerability data from multiple organizations, in different industries, but certainly not all possible device types, OSs and roles were present in the dataset. To improve upon this limitation, the implementation of the model needs to be checked against the data that should be analyzed and the classes need to be supplemented as suggested in section 6.2. Generating a more complete set of classes for the inferring of the CVSSv3 Environmental Score is definitely something, that can be done based on the model suggested in this thesis. Furthermore, rules were only created for the initial prototype of the model using one major commercial vulnerability scanner, Tenable Nessus. Creating the rules for other vulnerability scanners should not be a problem itself, as shown in the analysis of other tools available data, but the detailed analysis of specific data reported by the scanners could need to be repeated, as discussed in chapter 5. If it is possible to create the rules based on the definitions of the SCAP, the standard could be used to create generalized rules that are directly interoperable between different implementations of vulnerability scanners. As discussed in section 5.3 the use of SCAP using the AID standard[24] for asset classification is not granular enough to be used in the model. The usage of CPE and SWID and their problems for usage in this model have been discussed for operating system and device identification in section 5.1 and for role identification in section 5.2. The conclusion was not to use it for the prototype implementation due to not having SWID data in the dataset at all and missing CPE entries and the requirement to create a much more complex rulebase due to CPE identifying specific software or hardware and not services directly. Another limitation of the work is the missing validation of the general CVSSv3 Security Requirement Metrics suggested in section 6.1. The values associated to the different classes are currently based on expert opinion and were not validated with organizations and their specific requirements. Creating a granular classification of different IT assets and defining empirical CVSSv3 Security Requirement Metrics for the general inferred score would make the model more valid in the automatic step of the model. For current use of the model, going through the classes and customizing these metric associations as suggested in section 6.2 should be performed.

By proposing the model in this thesis, a possibility for better vulnerability prioritization is created for all organizations on different maturity levels in security. Due to the improvements that can be implemented on top of the automatic model's base, the inferred scores are available for any organization and can be

improved and customized optionally, if the organization has the required data sources and capacities to do these improvements, which require manual work. The improvements that can be performed on the model are ranked into two steps, to first create a minimal, but effective way, to supplement the automatically inferred results. The second step improves upon the first and can implement the security system of an organization into the models scoring.

8.1 Future Work

The model described in this thesis is requiring more work to implement it completely for a use in any organization. To create an implementation of the model, more data from different vulnerability scans and scanners needs to be analyzed, to create more rules to classify systems. This has to be done on one hand to detect more device types and roles, which is required for better classification, and to be able to use this model with more vulnerability scanners. The current use of a possible limited dataset in the creation of the first proposed ruleset could be not implementing all device types found in organizations that have IT systems, which were not covered by the used dataset for the model creation. By evaluating the methods shown in sections 6.2, 6.3 and 6.5, the model could be improved further, based on the requests of real organizations security systems. This evaluation was not possible in the scope of this thesis, since organization specific data, like network segmentation, was not available for the evaluation and direct reviews with the organizations were not possible. The prioritization methods shown in sections 6.4 and 6.5 could also be incorporated into a numerical score, to be used directly as a single number. With that, the result of the EPSS would be combined directly with the CVSS score inferred. The possibilities on how to combine the severity of a vulnerability measured with the CVSS Score and the exploitability of a vulnerability measured with the EPSS score should be further examined to give organizations management a single metric that combines this information. In addition to that, the model could also benefit from incorporating the prioritization of systems in an organizational context directly into the numeric score, as described in section 6.5.

By modifying the CVSSv3.1 Security Requirements Metric Group and adding more values than the three that currently exist (excluding *Not Defined*), the classifications could be made more granular. Currently only the values *Low* and *High* actually modify the resulting CVSS Environmental Score. It could be beneficial to add values between the existing values to increase the number of ratings possible for the classifications shown in tables 6.1 and 6.2. This improvement would require a deviation from the CVSSv3.1 specification and was therefore not performed in this thesis. These modifications of the CVSS standard could be proposed to the CVSS SIG to implement them in CVSSv4. When CVSSv4 is released, the model will need to

be adapted to the new scoring metrics implemented in the standard. Some modifications that are already approved are prepared to be implemented into the model and discussed in section 6.4.
The suggested general CVSSv3 Security Requirements Metrics in section 6.1 should also be validated with experts and organizations to create a better average for the classes identified. By creating a valid general mapping of the requirements, that can be applied automatically to any organization, the requirements of each class can be further improved by better metrics associated with them.
Further improvements to the automatic inferring of the scores can be made, by implementing automatic network topology information to the model. This way topology of the networks can be used automatically by the model to improve the scoring, without the requirement to manually add this information as described in section 6.2. [115] implements such techniques but does not go into detail how it is performed. Probably special tools, in addition to the vulnerability scanner data, are required to retrieve this information.
To aid the manual parts of the model and include more risk management information, the findings and the SSVC, as presented in [61], could be used in addition to the model and EPSS for better prioritization and to further improve upon the prioritization of the vulnerabilities. These further improved methods could then be tested together with the organization specific improvements to the model mentioned in chapter 6.
With the EPSS available for more CVE IDs and the CVSSv3 available for all vulnerabilities found in an organization, the prioritization could be further improved drastically. The evaluation of the model in chapter 7 showed, that more than 90% of the vulnerability instances in the dataset could not be associated to an EPSS score. Therefore, the EPSS cannot be used for the prioritization of most vulnerabilities, even though the effectiveness of this scoring model, as shown in [23] would be important to the prioritization of vulnerabilities in organizations. By eliminating this gap, the prioritization of all vulnerabilities of the organization would benefit from these improvements. With the results shown in [55], the gap of vulnerabilities not having CVSSv3 scores could be reduced and historic vulnerabilities, which are still present in today's organizations, as found in the dataset analyzed in the scope of this thesis, could be prioritized using the new version of CVSS. Alternatively, the models result could be translated for use with CVSSv2.
Scoring all systems vulnerabilities based on the highest security requirements of a single role of a system is a pragmatic approach, as mentioned in section 6.1. This could be improved by deciding which vulnerabilities apply to certain server roles and do not affect other roles, as measured with the CVSS-S metric. When implementing this, it could be used to further prioritize vulnerabilities on systems and to do a more granular prioritization based on roles and services instead on a system basis as in the proposed model. By changing to this more granular approach, the model's complexity is increased, but the ranking of the vulnerabilities would better reflect the requirements defined for certain services.

List of Figures

List of Tables

List of Equations

Glossary

AD	Windows Server Active Directory
AD DC	AD Domain Controller
AID	Asset Identification
API	Application Programming Interface
ARF	Asset Reporting Format
CCE	Common Configuration Enumeration
CCSS	Common Configuration Scoring System
CMDB	Configuration Management Database
CNA	CVE Numbering Authority
CPE	Common Platform Enumeration
CRISM	Cyber Risk Scoring and Mitigation
CSOC	Cyber-Security Operations Center
CVE	Common Vulnerabilities and Exposures
CVE ID	CVE Identifier
CVSS	Common Vulnerability Scoring System
CVSS SIG	CVSS Special Interest Group
CVSS-A	CVSS Availability Metric
CVSS-AC	CVSS Attack Complexity Metric
CVSS-AR	CVSS Availability Requirement Metric
CVSS-AV	CVSS Attack Vector Metric
CVSS-C	CVSS Confidentiality Metric
CVSS-CR	CVSS Confidentiality Requirement Metric
CVSS-E	CVSS Exploit Code Maturity Metric

CVSS-I	CVSS Integrity Metric
CVSS-IR	CVSS Integrity Requirement Metric
CVSS-MA	CVSS Modified Availability Metric
CVSS-MAC	CVSS Modified Attack Complexity Metric
CVSS-MAV	CVSS Modified Attack Vector Metric
CVSS-MC	CVSS Modified Confidentiality Metric
CVSS-MI	CVSS Modified Integrity Metric
CVSS-MPR	CVSS Modified Privileges Required Metric
CVSS-MS	CVSS Modified Scope Metric
CVSS-MUI	CVSS Modified User Interaction Metric
CVSS-PR	CVSS Privileges Required Metric
CVSS-RC	CVSS Report Confidence Metric
CVSS-RL	CVSS Remediation Level Metric
CVSS-S	CVSS Scope Metric
CVSS-UI	CVSS User Interaction Metric
CVSSv1	CVSS Version 1
CVSSv2	CVSS Version 2
CVSSv3	CVSS Version 3
CVSSv3.0	CVSS Version 3.0
CVSSv3.1	CVSS Version 3.1
CVSSv4	CVSS Version 4.0
CWE	Common Weakness Enumeration
CWRAF	Common Weakness Risk Analysis Framework
CWSS	Common Weakness Scoring System
DHCP	Dynamic Host Configuration Protocol
DNS	Domain Name System
EDB	Exploit Database
EPSS	Exploit Prediction Scoring System

EPSS SIG	EPSS Special Interest Group
FIRST	Forum of Incident Response and Security Teams Inc.
FTP	File Transfer Protocol
ICS	Industrial Control Systems
IDS	Intrusion Detection System
IEC	International Electrotechnical Commission
IMAP	Internet Message Access Protocol
IoC	Indicator of Compromise
IP	Internet Protocol
IPMI	Intelligent Platform Management Interface
ISMS	Information Security Management System
ISO	International Organization for Standardization
IT	Information Technology
LDAP	Lightweight Directory Access Protocol
ML	Machine Learning
MSRC	Microsoft Security Response Center
NIST	National Institute of Standards and Technology of the US Department of Commerce
NTP	Network Time Protocol
NVD	US National Vulnerability Database
OCIL	Open Checklist Interactive Language
OS	Operating System
OT	Operational Technology
OVAL	Open Vulnerability and Assessment Language

plugin ID	Tenable Nessus Plugin Identification Number
PoC	Proof of Concept
POP3	Post Office Protocol Version 3
SCAP	Security Content Automation Protocol
SIP	Session Initiation Protocol
SMB	Server Message Block
SMTP	Simple Mail Transfer Protocol
SSH	Secure Shell
SSVC	Stakeholder-Specific Vulnerability Categorization
SWAT	Samba Web Administration Tool
SWID	Software Identification Tag
URI	Uniform Resource Identifier
VM	Vulnerability Management
VMC	Vulnerability Management Centre
VPR	Vulnerability Priority Rating
VRSS	Vulnerability Rating and Scoring System
WFN	Well-Formed CPE Name
WIVSS	Weighted Impact Vulnerability Scoring System
WMI	Windows Management Instrumentation
XCCDF	Extensible Configuration Checklist Description Format
XML	Extensible Markup Language
XPath	XML Path Language
XSS	Cross-Site-Scripting

Bibliography

[1] Caitlin Condon, *ProxyShell: More widespread exploitation of microsoft exchange servers*, https://www.rapid7.com/blog/post/2021/08/12/proxyshell-more-widespread-exploitation-of-microsoft-exchange-servers/, Accessed: 2021-08-17, Aug. 2021.

[2] Park Foreman, *Vulnerability Management*. Taylor & Francis Group, LLC, 2019, ISBN: 978-0-367-23514-7.

[3] Serkan Özkan, *CVE details, Browse vulnerabilities by date*, https://www.cvedetails.com/browse-by-date.php, Accessed: 2021-08-26, 2021.

[4] Forum of Incident Response and Security Teams, Inc., *Common vulnerability scoring system v3.1: Specification document*, https://www.first.org/cvss/v3.1/specification-document, Accessed: 2021-04-25, Jun. 2019.

[5] Assad Ali, Pavol Zavarsky, Dale Lindskog, and Ron Ruhl, "A software application to analyze the effects of temporal and environmental metrics on overall CVSS v2 score," in *2011 World Congress on Internet Security (WorldCIS-2011)*, 2011, pp. 109–113. DOI: 10.1109/WorldCIS17046.2011.5749893.

[6] Luca Allodi, Silvio Biagioni, Bruno Crispo, Katsiaryna Labunets, Fabio Massacci, and Wagner Medeiros dos Santos, "Estimating the assessment difficulty of CVSS environmental metrics: An experiment," Nov. 2017, pp. 23–39, ISBN: 978-3-319-70003-8. DOI: 10.1007/978-3-319-70004-5_2.

[7] Cyentia Institute and Kenna Security, "Prioritization to prediction, Volume 3: Winning the remediation race," Tech. Rep., Mar. 2019, Accessed: 2021-08-01. [Online]. Available: https://library.cyentia.com/report/report_002993.html.

[8] Andrew Magnusson, *Practical vulnerability management, a strategic approach to managing cyber risk*. No Starch Press, Inc, 2020, ISBN: 978-1-59327-989-9.

[9] The MITRE Corporation, *Corporate overview*, https://www.mitre.org/about/corporate-overview, Accessed: 2021-08-06, 2021.

[10] ——, *About CWE*, https://cwe.mitre.org/about/, Accessed: 2021-07-26, Mar. 2021.

[11] ——, *CVE terminology*, https://cve.mitre.org/about/terminology.html, Accessed: 2021-07-26, Mar. 2021.

[12] European Union Agency for Cybersecurity (ENISA), *Glossary, Vulnerabilities and exploits*, https://www.enisa.europa.eu/topics/csirts-in-europe/glossary/vulnerabilities-and-exploits, Accessed: 2021-08-06, 2021.

[13] Tenable, Inc., *Credentials in vulnerability management scans*, https://docs.tenable.com/tenableio/Content/Scans/Credentials.htm, Accessed: 2021-08-06, 2021.

[14] Hannes Holm, Teodor Sommestad, Jonas Almroth, and Mats Persson, "A quantitative evaluation of vulnerability scanning," *Information Management & Computer Security*, vol. 19, no. 4, pp. 231–247, Oct. 2011. DOI: 10.1108/09685221111173058.

[15] Karen Scarfone, Murugiah Souppaya, Amanda Cody, and Angela Orebaugh, "Technical guide to information security testing and assessment.," Tech. Rep., Sep. 2008. DOI: 10.6028/nist.sp.800-115.

[16] Kyle Coffey, R. Smith, Leandros Maglaras, and Helge Janicke, "Vulnerability analysis of network scanning on scada systems," *Security and Communication Networks*, vol. 2018, Feb. 2018. DOI: 10.1155/2018/3794603.

[17] Murugiah Souppaya and Karen Scarfone, "Guide to enterprise patch management technologies," Tech. Rep., Jul. 2013. DOI: 10.6028/nist.sp.800-40r3.

[18] Information Systems Audit and Control Association (ISACA), *COBIT® 2019 Framework: Introduction and Methodology*. 2018, ISBN: 978-1-60420-763-7. [Online]. Available: https://www.isaca.org/bookstore/bookstore-cobit_19-digital/wcb19fim.

[19] ——, *COBIT® 2019 Framework: Governance and Management Objectives*. 2018, ISBN: 978-1-60420-764-4. [Online]. Available: https://www.isaca.org/bookstore/bookstore-cobit_19-digital/wcb19fgm.

[20] National Institute of Standards and Technology, "Security and privacy controls for information systems and organizations," Tech. Rep., Sep. 2020. DOI: 10.6028/nist.sp.800-53r5.

[21] Peter Mell, Tiffany Bergeron, and David Henning, "Creating a patch and vulnerability management program," Tech. Rep., Nov. 2005. DOI: 10.6028/nist.sp.800-40ver2.

[22] The MITRE Corporation, *CVE frequently asked questions*, https://cve.mitre.org/about/faqs.html, Accessed: 2021-07-26, Mar. 2021.

[23] Jay Jacobs, Sasha Romanosky, Benjamin Edwards, Idris Adjerid, and Michael Roytman, "Exploit prediction scoring system (EPSS)," *Digital Threats: Research and Practice*, vol. 2, no. 3, Jun. 2021, ISSN: 2692-1626. DOI: 10.1145/3436242.

[24] John Wunder, Adam Halbardier, and David Waltermire, "Specification for asset identification 1.1," National Institute of Standards and Technology, Tech. Rep., Jun. 2011. DOI: 10.6028/nist.ir.7693.

[25] David Waltermire, Stephen Quinn, Harold Booth, Karen Scarfone, and Dragos Prisaca, "The technical specification for the security content automation protocol (SCAP) version 1.3," National Institute of Standards and Technology, Tech. Rep., Feb. 2018. DOI: 10.6028/nist.sp.800-126r3.

[26] David Waltermire and Jessica Fitzgerald-McKay, "Transitioning to the security content automation protocol (SCAP) version 2," National Institute of Standards and Technology, Tech. Rep., Sep. 2018. DOI: 10.6028/nist.cswp.09102018.

[27] Stephen D Quinn, Murugiah Souppaya, Melanie Cook, and Karen Scarfone, "National checklist program for IT products - guidelines for checklist users and developers," Tech. Rep., Feb. 2018. DOI: 10.6028/nist.sp.800-70r4.

[28] National Institute of Standards and Technology, *Security content automation protocol*, https://csrc.nist.gov/projects/Security-Content-Automation-Protocol, Accessed: 2021-07-18, Jul. 2021.

[29] Raydel Montesino and Stefan Fenz, "Information security automation: How far can we go?" In *2011 Sixth International Conference on Availability, Reliability and Security*, 2011, pp. 280–285. DOI: 10.1109/ARES.2011.48.

[30] Gerard T McGuire and Emily E Reid, "The state of security automation standards - 2011," The MITRE Corporation, Tech. Rep., 2011. [Online]. Available: https://www.mitre.org/sites/default/files/pdf/11_3822.pdf.

[31] Alexandre D'Hondt and Hussein Bahmad, "Understanding scap through a simple use case," *Hakin9 IT Security Magazine*, vol. 11, pp. 100–110, 2016. [Online]. Available: https://dial.uclouvain.be/memoire/ucl/en/object/thesis:8128/datastream/PDF_05/view.

[32] The MITRE Corporation, *Scoring CWEs, Common weakness scoring system (CWSS™)*, https://cwe.mitre.org/cwss/, Accessed: 2021-07-26, Sep. 2014.

[33] ——, *Introduction to CWRAF™*, https://cwe.mitre.org/cwraf/introduction.html, Accessed: 2021-07-26, Apr. 2013.

[34] Brant A Cheikes, David Waltermire, and Karen Scarfone, "Common platform enumeration: Naming specification version 2.3," National Institute of Standards and Technology, Tech. Rep., Aug. 2011. DOI: 10.6028/nist.ir.7695.

[35] Mary C Parmelee, Harold Booth, David Waltermire, and Karen Scarfone, "Common planform enumeration: Name matching specification version 2.3," National Institute of Standards and Technology, Tech. Rep., Aug. 2011. DOI: 10.6028/nist.ir.7696.

[36] Paul Cichonski, David Waltermire, and Karen Scarfone, "Common platform enumeration: Dictionary specification version 2.3," National Institute of Standards and Technology, Tech. Rep., Aug. 2011. DOI: 10.6028/nist.ir.7697.

[37] David Waltermire, Paul Cichonski, and Karen Scarfone, "Common platform enumeration: Applicability language specification version 2.3," National Institute of Standards and Technology, Tech. Rep., Aug. 2011. DOI: 10.6028/nist.ir.7698.

[38] National Institute of Standards and Technology, *National vulnerability database (NVD), Official common platform enumeration (CPE) dictionary*, http://nvd.nist.gov/cpe.cfm, Accessed: 2021-07-27, 2021.

[39] Luis Alberto Benthin Sanguino and Rafael Uetz, "Software vulnerability analysis using CPE and CVE," May 2017. arXiv: 1705.05347.

[40] "Information technology – software asset management – part 2: Software identification tag," International Organization for Standardization, Standard ISO/IEC 19770-2:2015(en), Oct. 2015. [Online]. Available: https://www.iso.org/standard/65666.html.

[41] David Waltermire, Brant A Cheikes, Larry Feldman, and Greg Witte, "Guidelines for the creation of interoperable software identification (SWID) tags," National Institute of Standards and Technology, Tech. Rep., Apr. 2016. DOI: 10.6028/nist.ir.8060.

[42] David Waltermire and Brant A Cheikes, "Forming common platform enumeration (CPE) names from software identification (SWID) tags," National Institute of Standards and Technology, draft, Dec. 2015. [Online]. Available: https://csrc.nist.gov/publications/detail/nistir/8085/draft.

[43] The MITRE Corporation, *CVE numbering authorities*, https://cve.mitre.org/cve/cna.html, Accessed: 2021-07-26, Jul. 2021.

[44] Forum of Incident Response and Security Teams, Inc., *Common vulnerability scoring system v3.0*, https://www.first.org/cvss/v3-0/, Accessed: 2021-04-25, Jun. 2015.

[45] ——, *Common vulnerability scoring system SIG – current initiatives – list of potential improvements for CVSS v4.0*, https://www.first.org/cvss/#Current-initiatives and https://bit.ly/cvssv4-workitems, Accessed: 2021-07-30, Jun. 2020.

[46] ——, *Common vulnerability scoring system v3.1: User guide*, https://www.first.org/cvss/v3.1/user-guide, Accessed: 2021-04-25, Jun. 2019.

[47] ——, *A complete guide to the common vulnerability scoring system version 2.0*, https://www.first.org/cvss/v2/guide, Accessed: 2021-07-18, Jun. 2007.

[48] National Institute of Standards and Technology, *National vulnerability database (NVD), Vulnerability metrics*, https://nvd.nist.gov/vuln-metrics/cvss, Accessed: 2021-07-28.

[49] Tenable, Inc., *Tenable documentation risk metrics*, https://docs.tenable.com/nessus/Content/RiskMetrics.htm, Accessed: 2021-07-28, 2021.

[50] Forum of Incident Response and Security Teams, Inc., *Common vulnerability scoring system v1 archive*, https://www.first.org/cvss/v1/, Accessed: 2021-07-18, Apr. 2005.

[51] ——, *Common vulnerability scoring system v2: History*, https://www.first.org/cvss/v2/history, Accessed: 2021-07-18, Jun. 2007.

[52] National Institute of Standards and Technology, *National vulnerability database (NVD), Search vulnerability database*, https://nvd.nist.gov/vuln/search, Accessed: 2021-07-18, 2021.

[53] Doudou Fall and Youki Kadobayashi, "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 - ICISSP*, INSTICC, SciTePress, 2019, pp. 405–411, ISBN: 978-989-758-359-9. DOI: 10.5220/0007387704050411.

[54] Horváth Attila, Péter Máté Erdősi, and Ferenc Kiss, "The common vulnerability scoring system (CVSS) generations – usefulness and deficiencies," in *It és hálózati sérülékenységek társadalmi-gazdasági hatásai*. Jan. 2016, pp. 137–153. [Online]. Available: https://infota.org/projektek/kutatasi-projektek/it-es-halozati-serulekenysegek-tovagyuruzo-tarsadalmi-gazdasagi-hatasai/.

[55] Maciej Nowak, Michał Walkowski, and Sławomir Sujecki, "Machine learning algorithms for conversion of cvss base score from 2.0 to 3.x," in *Computational Science – ICCS 2021*, Cham: Springer International Publishing, 2021, pp. 255–269, ISBN: 978-3-030-77967-2. DOI: 10.1007/978-3-030-77967-2_21.

[56] Marc Ruef, *How to convert risk scores (CVSSv1, CVSSv2, CVSSv3, OWASP risk severity)?* Stackoverflow https://security.stackexchange.com/a/127447, Accessed: 2021-07-26, Jun. 2019.

[57] Tenable, Inc., *CVSS severity changes for plugins in tenable products*, https://community.tenable.com/s/article/CVSS-Severity-Changes-for-plugins-in-Tenable-products, Accessed: 2021-07-26, May 2021.

[58] Dave Dugal and Dale Rich, *Common vulnerability scoring system - the state of CVSS to come*, presentation slides: https://www.first.org/resources/papers/sig-may-jun2021/CVSS-v4-FIRST-SIG-Update-2021.pdf helt on [59], Accessed: 2021-07-30, May 2021.

[59] Shawn Richardson, Sasha Romanosky, Jay Jacobs, Dave Dugal, Dale Rich, Swathi Joshi, and Emer O'Neill, *FIRST SIG updates: EPSS, CVSS, ethics, women of FIRST | thursday. may 13*, session recording: https://youtu.be/AHKOtj7sk5o, EPSS at 0:05:25, CVSSv4 at 1:10:38, Accessed: 2021-07-30, May 2021.

[60] Francois Boechat, Gabriel Ribas, Lucas Senos, Miguel Bicudo, Mateus Schulz Nogueira, Leandro Pfleger de Aguiar, and Daniel Sadoc Menasche, "Is vulnerability report confidence redundant? pitfalls using temporal risk scores," *IEEE Security Privacy*, vol. 19, no. 4, pp. 44–53, 2021. DOI: 10.1109/MSEC.2021.3070978.

[61] Jonathan M Spring, Allen Householder, Eric Hatleback, Art Manion, Oliver Madison, Vijay Sarvepalli, Laurie Tyzenhaus, and Charles Yarbrough, "Prioritizing vulnerability response: A stakeholder-specific vulnerability categorization (version 2.0)," Apr. 2021. [Online]. Available: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=653459.

[62] Jay Jacobs, Sasha Romanosky, Idris Adjerid, and Wade Baker, "Improving vulnerability remediation through better exploit prediction," *Journal of Cybersecurity*, vol. 6, no. 1, Sep. 2020, ISSN: 2057-2085. DOI: 10.1093/cybsec/tyaa015.

[63] Forum of Incident Response and Security Teams, Inc., *The EPSS model*, https://www.first.org/epss/model, Accessed: 2021-07-31, 2021.

[64] ——, *EPSS user guide, Using epss for better vulnerability management*, https://www.first.org/epss/user-guide, Accessed: 2021-07-31, 2021.

[65] ——, *EPSS data*, https://www.first.org/epss/data_stats, Accessed: 2021-08-19, 2021.

[66] Microsoft, *Security update severity rating system*, https://www.microsoft.com/en-us/msrc/security-update-severity-rating-system, Accessed: 2021-07-28, 2021.

[67] ——, *Microsoft exploitability index*, https://www.microsoft.com/en-us/msrc/exploitability-index, Accessed: 2021-07-28, 2021.

[68] Awad A Younis and Yashwant K Malaiya, "Comparing and evaluating cvss base metrics and microsoft rating system," in *2015 IEEE International Conference on Software Quality, Reliability and Security*, Aug. 2015, pp. 252–261. DOI: 10.1109/QRS.2015.44.

[69] Christopher Kissel and Frank Dickson, "Worldwide device vulnerability management market shares, 2019: Finding the transitional elements between device assessment scanning and risk-based remediation," IDC Corporate USA, Tech. Rep., May 2020.

[70] Wei Tai, *What is VPR and how is it different from CVSS?* https://www.tenable.com/blog/what-is-vpr-and-how-is-it-different-from-cvss, Accessed: 2021-07-28, Apr. 2020.

[71] Jonathan M Spring, Eric Hatleback, Allen Householder, Art Manion, and Deana Shick, "Towards improving CVSS," Dec. 2018. [Online]. Available: https://resources.sei.cmu.edu/library/asset-view.cfm?assetid=538368.

[72] Red Hat, Inc., *Severity ratings*, https://access.redhat.com/security/updates/classification, Accessed: 2021-08-06, 2021.

[73] Qualys Inc., *Qualys VM, PC, SCA, Severity levels*, https://qualysguard.qg2.apps.qualys.com/qwebhelp/fo_portal/knowledgebase/severity_levels.htm, Accessed: 2021-09-07, 2021.

[74] ——, *Qualys severity score vs CVSS scoring*, https://success.qualys.com/discussions/s/article/000002759, Accessed: 2021-09-07, Mar. 2019.

[75] ——, *Qualys VM, PC, SCA, CVSS scoring*, https://qualysguard.qg2.apps.qualys.com/qwebhelp/fo_portal/setup/cvss_scoring.htm, Accessed: 2021-09-07, 2021.

[76] Rapid7, *Know nexpose? it's time to meet InsightVM.* https://www.rapid7.com/products/nexpose/insightvm-comparison/, Accessed: 2021-09-07, 2021.

[77] ——, *Quantifying risk with insightvm*, https://www.rapid7.com/globalassets/_pdfs/product-and-service-briefs/rapid7-solution-brief-quantifying-risk-insightvm.pdf, Accessed: 2021-09-07, 2020.

[78] ——, *Nexpose documentation, Risk strategies*, https://docs.rapid7.com/nexpose/working-with-risk-strategies-to-analyze-threats/, Accessed: 2021-09-07, 2021.

[79] ——, *PCI, CVSS, & risk scoring frequently asked questions*, https://help.rapid7.com/nexpose/en-us/Files/Risk_scoring_FAQ.html, Accessed: 2021-09-07, 2021.

[80] Gabriela Roldán-Molina, Mario Almache-Cueva, Carlos Silva-Rabadão, Iryna Yevseyeva, and Vitor Basto-Fernandes, "A comparison of cybersecurity risk analysis tools," *Procedia Computer Science*, vol. 121, pp. 568–575, 2017, CENTERIS 2017 - International Conference on ENTERprise Information Systems / ProjMAN 2017 - International Conference on Project MANagement / HCist 2017 - International Conference on Health and Social Care Information Systems and Technologies, CENTERIS/ProjMAN/HCist 2017, ISSN: 1877-0509. DOI: 10.1016/j.procs.2017.11.075.

[81] Tenable, Inc., *A comparison of tenable and rapid7 approaches to vulnerability prioritization*, https://www.tenable.com/sites/drupal.dmz.tenablesecurity.com/files/uploads/documents/whitepapers/TEN_Rapid7PriorPaper_Final.pdf, Accessed: 2021-09-08, 2019.

[82] IBM X-Force, *IBM X-Force exchange vulnerabilities*, https://exchange.xforce.ibmcloud.com/activity/list?filter=Vulnerabilities, Accessed: 2021-09-08, 2021.

[83] Qixu Liu, Yuqing Zhang, Ying Kong, and Qianru Wu, "Improving VRSS-based vulnerability prioritization using analytic hierarchy process," *Journal of Systems and Software*, vol. 85, no. 8, pp. 1699–1708, 2012, ISSN: 0164-1212. DOI: 10.1016/j.jss.2012.03.057.

[84] Qixu Liu and Yuqing Zhang, "VRSS: A new system for rating and scoring vulnerabilities," *Computer Communications*, vol. 34, no. 3, pp. 264–273, 2011, Special Issue of Computer Communications on Information and Future Communication Security, ISSN: 0140-3664. DOI: 10.1016/j.comcom.2010.04.006.

[85] Tenable, Inc., *CVSS scores in tenable plugins*, https://community.tenable.com/s/article/CVSS-Scores-in-Tenable-Plugins, Accessed: 2021-08-22, Dec. 2020.

[86] Pontus Johnson, Robert Lagerström, Mathias Ekstedt, and Ulrik Franke, "Can the common vulnerability scoring system be trusted? a bayesian analysis," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 6, pp. 1002–1015, 2018. DOI: 10.1109/TDSC.2016.2644614.

[87] Luca Allodi and Fabio Massacci, "A preliminary analysis of vulnerability scores for attacks in wild: The ekits and sym datasets," ser. BADGERS '12, Raleigh, North Carolina, USA: Association for Computing Machinery, 2012, pp. 17–24, ISBN: 978-1-4503-1661-3. DOI: 10.1145/2382416.2382427.

[88] Luca Allodi, Woohyun Shim, and Fabio Massacci, "Quantitative assessment of risk reduction with cybercrime black market monitoring," in *2013 IEEE Security and Privacy Workshops*, 2013, pp. 165–172. DOI: 10.1109/SPW.2013.16.

[89] Luca Allodi and Fabio Massacci, "Comparing vulnerability severity and exploits using case-control studies," *ACM Transactions on Information and System Security*, vol. 17, no. 1, Aug. 2014, ISSN: 1094-9224. DOI: 10.1145/2630069.

[90] Jonathan M Spring, Eric Hatleback, Allen Householder, Art Manion, and Deana Shick, "Time to change the CVSS?" *IEEE Security Privacy*, vol. 19, no. 2, pp. 74–78, 2021. DOI: 10.1109/MSEC.2020.3044475.

[91] Hannes Holm and Khalid Khan Afridi, "An expert-based investigation of the common vulnerability scoring system," *Computers & Security*, vol. 53, no. C, pp. 18–30, Sep. 2015, ISSN: 0167-4048. DOI: 10.1016/j.cose.2015.04.012.

[92] Jukka Ruohonen, "A look at the time delays in CVSS vulnerability scoring," *Applied Computing and Informatics*, vol. 15, no. 2, pp. 129–135, 2019, ISSN: 2210-8327. DOI: 10.1016/j.aci.2017.12.002.

[93] Luca Allodi, Marco Cremonini, Fabio Massacci, and Woohyun Shim, "The effect of security education and expertise on security assessments: The case of software vulnerabilities," 2018. arXiv: 1808.06547.

[94] Christian Fruhwirth and Tomi Männistö, "Improving CVSS-based vulnerability prioritization and response with context information," in *2009 3rd International Symposium on Empirical Software Engineering and Measurement*, 2009, pp. 535–544. DOI: 10.1109/ESEM.2009.5314230.

[95] Deven Pandya and Dr. N.J. Patel, "To study and analyze the impact of confidentiality, integrity, and availability (CIA) on common vulnerability scoring system (CVSS) base score," *International Journal of Advanced Research in Computer Science*, vol. 9, no. 1, pp. 883–886, 2018, ISSN: 0976-5697. DOI: 10.26483/ijarcs.v9i1.5380. [Online]. Available: http://ijarcs.info/index.php/Ijarcs/article/view/5380.

[96] Michał Walkowski, Maciej Krakowiak, Jacek Oko, and Sławomir Sujecki, "Efficient algorithm for providing live vulnerability assessment in corporate network environment," *Applied Sciences*, vol. 10, p. 7926, Nov. 2020. DOI: 10.3390/app10217926.

[97] ——, "Distributed analysis tool for vulnerability prioritization in corporate networks," in *2020 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, 2020, pp. 1–6. DOI: 10.23919/SoftCOM50211.2020.9238286.

[98] DSecureMe, *Vulnerability management center (VMC) documentation*, https://github.com/DSecureMe/vmc-docs, Accessed: 2021-07-29, 2021.

[99] Georgios Spanos, Lefteris Angelis, and Dimitrios Toloudis, "Assessment of vulnerability severity using text mining," in *Proceedings of the 21st Pan-Hellenic Conference on Informatics*, ser. PCI 2017, Larissa, Greece: Association for Computing Machinery, 2017. DOI: 10.1145/3139367.3139390.

[100] Yongjae Lee and Seungwon Shin, "Toward semantic assessment of vulnerability severity: A text mining approach," in *Proceedings of the CIKM 2018 Workshops co-located with 27th ACM International Conference on Information and Knowledge Management (CIKM 2018), 1st International Workshop on EntitY REtrieval (EYRE 2018)*, vol. Vol-2482, 2018. [Online]. Available: http://ceur-ws.org/Vol-2482/paper8.pdf.

[101] Masoud Khosravi-Farmad, Razieh Rezaee, and Abbas Ghaemi Bafghi, "Considering temporal and environmental characteristics of vulnerabilities in network security risk assessment," in *2014 11th International ISC Conference on Information Security and Cryptology*, 2014, pp. 186–191. DOI: 10.1109/ISCISC.2014.6994045.

[102] Omer Keskin, Nick Gannon, Brian Lopez, and Unal Tatar, "Scoring cyber vulnerabilities based on their impact on organizational goals," in *2021 Systems and Information Engineering Design Symposium (SIEDS)*, 2021, pp. 1–6. DOI: 10.1109/SIEDS52267.2021.9483741.

[103] HyunChul Joh and Yashwant K Malaiya, "Defining and assessing quantitative security risk measures using vulnerability lifecycle and CVSS metrics," in *International Conference on Security and Management (SAM'11)*, 2011. [Online]. Available: https://www.cs.colostate.edu/~malaiya/p/johrisk11.pdf.

[104] Siv Hilde Houmb and Virginia N L Franqueira, "Estimating ToE risk level using CVSS," in *2009 International Conference on Availability, Reliability and Security*, 2009, pp. 718–725. DOI: 10.1109/ARES.2009.151.

[105] Pengsu Cheng, Lingyu Wang, Sushil Jajodia, and Anoop Singhal, "Aggregating CVSS base scores for semantics-rich network security metrics," in *2012 IEEE 31st Symposium on Reliable Distributed Systems*, 2012, pp. 31–40. DOI: 10.1109/SRDS.2012.4.

[106] Ruyi Wang, Ling Gao, Qian Sun, and Deheng Sun, "An improved CVSS-based vulnerability scoring mechanism," in *2011 Third International Conference on Multimedia Information Networking and Security*, 2011, pp. 352–355. DOI: 10.1109/MINES.2011.27.

[107] Georgios Spanos, Angeliki Sioziou, and Lefteris Angelis, "WIVSS: A new methodology for scoring information systems vulnerabilities," in *Proceedings of the 17th Panhellenic Conference on Informatics*, ser. PCI '13, Thessaloniki, Greece: Association for Computing Machinery, 2013, pp. 83–90. DOI: 10.1145/2491845.2491871.

[108] Georgios Spanos and Lefteris Angelis, "Impact metrics of security vulnerabilities: Analysis and weighing," *Information Security Journal: A Global Perspective*, vol. 24, no. 1-3, pp. 57–71, 2015. DOI: 10.1080/19393555.2015.1051675.

[109] Linus Karlsson, Pegah Nikbakht Bideh, and Martin Hell, "A recommender system for user-specific vulnerability scoring," in *Risks and Security of Internet and Systems*, Cham: Springer International Publishing, 2020, pp. 355–364, ISBN: 978-3-030-41568-6. DOI: 10.1007/978-3-030-41568-6_23.

[110] Katheryn A Farris, Ankit Shah, George Cybenko, Rajesh Ganesan, and Sushil Jajodia, "VULCON: A system for vulnerability prioritization, mitigation, and management," *ACM Transactions on Privacy and Security*, vol. 21, no. 4, Jun. 2018, ISSN: 2471-2566. DOI: 10.1145/3196884.

[111] Gabriela Roldán-Molina, Mario Almache-Cueva, Carlos Silva-Rabadão, Iryna Yevseyeva, and Vitor Basto-Fernandes, "A decision support system for corporations cybersecurity management," in *2017 12th Iberian Conference on Information Systems and Technologies (CISTI)*, 2017, pp. 1–6. DOI: 10.23919/CISTI.2017.7975826.

[112] BeyondTrust Corporation, *Vulnerability management*, https://www.beyondtrust.com/vulnerability-management, Accessed: 2021-09-07, 2020.

[113] Andrew Mokotoff, Zachary Robbins, and Barrett Wolfson, "Visualizing contextual information to prioritize network vulnerability management," Worcester Polytechnic Institute, Tech. Rep., Mar. 2017. [Online]. Available: https://digital.wpi.edu/concern/student_works/t435gf227.

[114] Virginia Modeling, Analysis and Simulation Center (VMASC) at Old Dominion University, *Cyber RIsk Scoring and Mitigation (CRISM)*, https://itsapps.odu.edu/crism/index.php, Accessed: 2021-08-16, 2021.

[115] Sachin Shetty, Michael McShane, Linfeng Zhang, Jay P Kesan, Charles A Kamhoua, Kevin Kwiat, and Laurent L. Njilla, "Reducing informational disadvantages to improve cyber risk management†," *The Geneva Papers on Risk and Insurance - Issues and Practice*, vol. 43, no. 2, pp. 224–238, Feb. 2018. DOI: 10.1057/s41288-018-0078-3.

[116] Luca Allodi, "Risk-based vulnerability management, Exploiting the economic nature of the attacker to build sound and measurable vulnerability mitigation strategies," Ph.D. dissertation, DISI - University of Trento, Apr. 2015. [Online]. Available: http://eprints-phd.biblio.unitn.it/1431/.

[117] Jenny Martinsson, *The use of vulnerability data for risk assessment*, http://lup.lub.lu.se/student-papers/record/9052929, Student Paper, Department of Electrical and Information Technology, Lund University, Jun. 2021.

[118] Jonathan M Spring, Eric Hatleback, Allen Householder, Art Manion, and Deana Shick, "Prioritizing vulnerability response: A stakeholder-specific vulnerability categorization," Nov. 2019. [Online]. Available: https://resources.sei.cmu.edu/library/asset-view.cfm?assetID=636379.

[119] The Open Group, *Open fair and quantitative risk analysis*, https://www.opengroup.org/forum/security-forum-0/openfairandquantitativerisk, Accessed: 2021-07-29, 2021.

[120] ——, "The open group standard - risk analysis (O-RA), version 2.0," Tech. Rep., Nov. 2020. [Online]. Available: https://publications.opengroup.org/c20a.

[121] ——, "The open group standard - risk taxonomy (O-RT), version 3.0," Tech. Rep., Nov. 2020. [Online]. Available: https://publications.opengroup.org/c20b.

[122] Andrew Josey, Jim Hietala, and Jack Jones, "An introduction to the open fair body of knowledge - a taxonomy and method for risk analysis," The Open Group, Tech. Rep., Jun. 2014. [Online]. Available: https://publications.opengroup.org/w148.

[123] Roman Wirtz and Maritta Heisel, "CVSS-based estimation and prioritization for security risks," in *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2019)*, SciTePress, 2019, pp. 297–306, ISBN: 978-989-758-375-9. DOI: 10.5220/0007709902970306.

[124] Tenable, Inc., *What is the nessus knowledgebase (KB)?* https://community.tenable.com/s/article/What-is-the-Nessus-Knowledgebase-KB, Accessed: 2021-08-16, Mar. 2021.

[125] Greenbone Networks GmbH, *History of the OpenVAS project*, https://greenbone.github.io/docs/background.html#history-of-the-openvas-project, Accessed: 2021-08-18, 2021.

[126] ——, *Greenbone security manager user manual, With Greenbone OS 21.04*, https://docs.greenbone.net/GSM-Manual/gos-21.04/en/GSM-Manual-GOS-21.04-en.pdf, Accessed: 2021-09-07, Aug. 2021.

[127] ——, *Greenbone security manager live demo system*, https://secinfo.greenbone.net/nvts, Accessed: 2021-09-07, 2021.

[128] Qualys Inc., *Qualys VM, PC, SCA, Vulnerability categories*, https://qualysguard.qg2.apps.qualys.com/qwebhelp/fo_portal/knowledgebase/vulnerability_categories.htm, Accessed: 2021-09-07, 2021.

[129] ——, *Qualys Global AssetView*, https://www.qualys.com/apps/global-assetview/, Accessed: 2021-09-07, 2021.

[130] Nathan Palanov, *Vulnerability categories and severity levels: "informational" vulnerabilities vs. true vulnerabilities, Risk strategies*, https://www.rapid7.com/blog/post/2016/12/15/vulnerability-categories-and-severity-levels-informational-vulnerabilities-vs-true-vulnerabilities/, Accessed: 2021-09-07, Dec. 2016.

[131] Rapid7, *Nexpose documentation, Performing filtered asset searches*, https://docs.rapid7.com/nexpose/performing-filtered-asset-searches/, Accessed: 2021-09-07, 2021.

[132] ——, *InsightVM API (V3), Asset*, https://help.rapid7.com/insightvm/en-us/api/index.html#tag/Asset, Accessed: 2021-09-07, 2021.

[133] Jesus Garcia Galan, *Asset detection with nessus scanners: The first step in assessing cyber risk*, https://tenable.com/blog/asset-detection-with-nessus-scanners-the-first-step-in-assessing-cyber-risk, Accessed: 2021-07-29, Feb. 2021.

[134] Tenable, Inc., *Nessus plugin families*, https://www.tenable.com/plugins/nessus/families, Accessed: 2021-08-16, 2021.

[135] Paulino Calderon, *Nmap: Network Exploration and Security Auditing Cookbook*, Second Edition. Birmingham: Packt Publishing Ltd, 2017, ISBN: 978-1-78646-745-4.

[136] Luciano Saltimbanco, Cody Dumont, Mateusz Wolanski, Brie Barrier, Lee S., Brie Barrier, and Venkatesh H., *How check if a windows server is a domain controller?* Tenable, Inc. Community, https://community.tenable.com/s/question/0D5f200005xzOpS/how-check-if-a-windows-server-is-a-domain-controller, Accessed: 2021-08-16, Feb. 2019.

[137] ubuntu Deutschland e. V., *Rsync*, https://wiki.ubuntuusers.de/rsync/, Accessed: 2021-08-17, 2021.

[138] Microsoft, *What is SharePoint?* https://support.microsoft.com/en-us/office/what-is-sharepoint-97b915e6-651b-43b2-827d-fb25777f446f, Accessed: 2021-08-17, 2021.

[139] Tenable, Inc., *Do not scan operational technology devices, Nessus plugin ID 109142*, https://www.tenable.com/plugins/nessus/109142, Accessed: 2021-08-18, 2021.

[140] National Institute of Standards and Technology - National Vulnerability Database, *CVE-2020-11023 detail*, https://nvd.nist.gov/vuln/detail/CVE-2020-11023, Accessed: 2021-09-08, Jul. 2021.

[141] Frederik Michel Dekking, Cornelis Kraaikamp, Hendrik Paul Lopuhaä, and Ludolf Erwin Meester, *A Modern Introduction to Probability and Statistics*. Springer London, 2005, ISBN: 1-85233-896-2. DOI: 10.1007/1-84628-168-7.