UNIVERSIDADE DE LISBOA
FACULDADE DE CIÊNCIAS
DEPARTAMENTO DE INFORMÁTICA

SecRush – New Generation Vulnerability Management Framework

Miguel Tomás Cabrita Santana

Mestrado em Segurança Informática

Dissertação orientada por:
Prof. Alan Oliveira de Sá

2022

Agradecimentos

Este foi mais um desafio concretizado! Desde cedo que soube que a área da segurança informática me interessava e queria aprofundar o tema. Não foi uma tarefa fácil e esta tese é fruto de muito esforço ao longo deste ano. Um ano cheio de vulnerabilidades para este trabalho, em todos os sentidos! Fui sacrificando algumas das coisas que gostaria de ter feito, mas todos os esforços têm recompensas, já que pude aprofundar a temática da gestão de vulnerabilidades, da qual fiquei bastante fã.

Durante este período, muitas pessoas se cruzaram comigo e fizeram parte desta grande aventura que é a dissertação do mestrado e tornaram esta experiência a melhor possível. Esta é sempre uma árdua tarefa de poder agradecer a todos os que contribuem positivamente para o nosso percurso sem nos esquecermos de ninguém.

Quero, em primeiro lugar, agradecer aos meus pais, os meus *sine qua non*, não só por terem financiado esta aventura, mas também por terem acompanhado de perto e estarem sempre presentes para qualquer coisa, mesmo que isso significasse ter de me ouvir horas a fio ao telefone com as habituais conversas existenciais! Não me posso também esquecer dos meus avós que valorizam imenso a educação que tenho tido oportunidade de receber e a motivação que me dão para continuar e ultrapassar desafios, mesmo quando a vontade é pouca. Um beijinho especial para a avó Lili que durante este percurso se tornou na estrela mais brilhante que ilumina os céus de noite.

Os amigos também foram extremamente importantes para a concretização deste grande projeto. Sem eles não haveriam tantos momentos de lazer, essenciais para ultrapassar os momentos difíceis, mas também para criar boas memórias. Um agradecimento especial a todos os amigos do Algarve (sem exceção) e aos amigos de faculdade por todo o carinho e amizade que mantemos até hoje.

Eternamente grato ao professor Alan por ter lido e relido vezes sem conta o manuscrito deste documento e pelas minhas insistências em desejar terminar o quanto antes. Sei que apesar da minha insistência, o brio profissional que leva permitiram que esta tese pudesse sair com a qualidade que a apresentamos. Acompanhou de perto todo este trabalho e fomentou sempre um espírito propício ao desenvolvimento desta tese e sempre com motivação para me guiar neste trabalho e juntos criarmos um bom projeto. Agradeço todas as inúmeras horas que passámos nas mais variadas formas de comunicação e o acompanhamento, também ele, quase 24/7, bem como todas as oportunidades para publicar o trabalho aqui desenvolvido.

Aos professores Dinis Pestana e Fernando Sequeira que contribuíram com excelentes comentários para a análise estatística e me ajudaram a desenvincilhar uma boa análise, bem como a pyxyp, que disponibilizou o acesso ao VulDB e que permitiu a recolha de informação bastante importante para este trabalho.

Por fim, um agradecimento especial ao Banco de Portugal, que me proporcionou este desafio do qual estou muito grato. Ao Luís Gonçalves, pela proposta inicial do tema que se revelou muito interessante, ao Carlos Moura, à Filipa Lima e ao Paulo Martins, por abraçarem este projeto e ao Pedro Silva, que acompanhou de perto este trabalho. Ao Eduardo Dias e à equipa da Plataforma Cliente por se terem disponibilizado a realizar os testes que conduziram a excelentes resultados. Também a toda a equipa da cibersegurança do Banco durante a minha passagem: Alexandre Barão, António Tavares, António Teixeira, Filipe Quaresma, Ida Corrêa, João Silva, Joaquim Santos, Manoel Neto, Marcelo Lima, Mário Anacleto, Rafael Ferreira, Ricardo Pires, Susana Fernandes e Tiago Neiva. Ao António Barros, um agradecimento especial pelas inúmeras conversas ciberseguras que foram tendendo para a gestão de vulnerabilidades e permitiram ideias interessantes neste projeto.

Esta tese é tanto minha como vossa.

A todos vós, sem exceção, o meu Muito Obrigado. Sejam felizes e mantenham-se ciberseguros!

Um bem-hajam.

Aos meus pais.

Resumo

Atualmente não existe uma forma de poder calcular o impacto direto que uma vulnerabilidade traz para a organização. A classificação dada é genérica e desprovida do risco associado para quem a possui. O CVSS é o sistema de classificação mais usado até à data mas é falho na sua forma de calcular atual. Várias críticas são apontadas [19][13] à pouca variabilidade da classificação e da impossibilidade de ser utilizado para priorizar vulnerabilidades de forma eficaz. Este sistema de classificação apresenta três tipos de métricas: métricas de base, métricas temporais e métricas ambientais. As métricas temporais são estanques e a classificação por elas dada é estática, não evoluindo com o tempo. Estas apenas são alteradas quando se verifica uma mudança em pelo menos uma das suas componentes. Já as métricas ambientais não avaliam a vulnerabilidade no contexto de uma organização com diversos ativos, com diferentes impactos na continuidade do negócio e diferentes visibilidades para a rede exterior. Ambas não representam o risco real da vulnerabilidade no contexto da organização. O risco que esta acarreta deve variar de acordo com este e outros fatores relevantes, tendo em conta as topologias dos sistemas informáticos e criticidade para a organização.

A somar a isto, os próprios planos de gestão de vulnerabilidades existentes encontram-se, por diversas vezes associados a empresas/produtos que vendem ferramentas de deteção de vulnerabilidades [6, 4, 10, 44, 15, 11, 32, 14, 29], no qual, os seus planos se centram em torno dos resultados apenas por elas produzido. Não só isto, como a literatura não tem apresentado outras perspetivas para um processo de gestão de vulnerabilidades genérico e passível de ser utilizado por qualquer organização, independentemente das ferramentas escolhidas para o concretizar. Mais recentemente, ainda impulsionado por estas empresas produtoras de ferramentas de deteção de vulnerabilidades, surgiu uma nova forma de gestão de vulnerabilidades baseadas em risco [37, 17, 38, 5, 39, 30, 31, 33]. Esta visão é baseada numa abordagem proactiva na mitigação de vulnerabilidades, orientada para as decisões baseadas no risco que estas trarão para a organização, tendo em conta todos os ativos expostos que cumprem ou não os requisitos de conformidade, através de uma descoberta contínua e dinâmica para exercer uma rápida ação. Assim, esta dissertação propõe a criação do SecRush, uma *framework* de gestão de vulnerabilidades de nova geração, orientada para o risco e capaz de ser aplicada em qualquer organização que necessite de endereçar as suas vulnerabilidades. O SecRush possui um processo, com um conjunto de fases que procedimentam a gestão de vulnerabilidades e o SecScore, um novo sistema de classificação de vulnerabilidades baseado no CVSS, com melhorias no aspeto temporal e ambiental, já que estes são os fatores que provocam um maior número de críticas a este sistema de classificação na literatura.

O processo proposto possui seis fases: Descoberta, Priorização, Reporte, Remediação, Verificação e Balanço. Na Descoberta são identificados todos os ativos da organização, bem como as vulnerabilidades a eles associadas. Na Priorização são classificadas e priorizadas as vulnerabilidades existentes

com o novo sistema de classificação proposto, ordenando as vulnerabilidades mais críticas primeiro. No Reporte, as vulnerabilidades são organizadas, documentadas e endereçadas aos donos dos ativos vulneráveis segundo um "rush" – um conjunto de vulnerabilidades a solucionar no espaço temporal definido pela organização (sendo o período mínimo de uma semana), com vista a tornar o processo de gestão de vulnerabilidades o mais ágil possível – semelhante a um *sprint* no SCRUM. Na Remediação aplicam-se as medidas e controlos necessários para a redução do risco de exposição da organização às vulnerabilidades reportadas na fase anterior, que está compartimentada pelas características que cada vulnerabilidade apresentará. Na Verificação, a equipa gestora do processo de gestão de vulnerabilidades garante que as medidas e controlos aplicados na fase anterior perfazem os requisitos para reduzir o risco de exposição da organização à vulnerabilidade, através de uma comunicação formal pelo responsável do ativo. Por fim, o Balanço pretende garantir uma informação sobre o estado atual de risco em torno das vulnerabilidades que a organização possui através da criação de relatórios táticos sobre as vulnerabilidades identificadas e remediadas, bem como o progresso que a organização tem vindo a sofrer na própria gestão das vulnerabilidades e ainda o risco global da empresa, providenciando aos executivos um sumário sobre o estado atual das vulnerabilidades.

O novo sistema de classificação, o SecScore, avalia a vulnerabilidade segundo as regras já estabelecidas pelo CVSS, com a adição da variação do tempo na fórmula com o hipotético aumento de risco da vulnerabilidade, que deve ser atualizada permanentemente para os casos onde as vulnerabilidades possuem um código de exploração ativo com classificação Alta. A somar a isto, e porque a vulnerabilidade encontra-se sempre inserida num contexto, esta deverá ser alvo de avaliação para averiguar a sua criticidade, através da priorização pela sua visibilidade para o exterior e importância para o negócio, para priorizar e distinguir máquinas que dispõem de serviços disponibilizados para o público, como um servidor, de uma estação de trabalho pessoal. A variação ao longo do tempo é definida por um índice previamente calculado de acordo com o perfil de risco da organização. Sobre um conjunto de 40 vulnerabilidades publicadas entre abril de 2021 e março de 2022 com o parâmetro temporal "Maturidade do Código de Exploração" com a classificação "Alta", efetuou-se uma análise estatística com o objetivo de calcular o índice temporal adequado à organização a afetar o SecScore. Para cada uma destas vulnerabilidades foi calculada a diferença entre a data em que o CVE é publicado até à data em que a empresa produtora do software vulnerável confirma a existência da vulnerabilidade e emite um aviso, muitas das vezes, acompanhado de uma forma de mitigação. Através do cálculo de um diagrama de extremos e quartis obtém-se três valores, que representam 25%, 50% e 75% da amostra ordenada, respetivamente. Estes valores estão em concordância com os habituais três níveis de apetite ao risco possíveis para as organizações. Para comparar o CVSS com o SecScore foram realizados testes sobre um cenário normal de priorização de vulnerabilidades no período de 12 de abril a 19 de abril de 2021, para simular uma semana normal de aplicação de mitigações. Escolheram-se seis vulnerabilidades para os testes, nos quais o CVE do mesmo já tinha sido publicado, embora a divulgação da mitigação de segurança pela empresa produtora do software ainda estivesse por divulgar. Com o CVSS, as vulnerabilidades recebiam classificações desprovidas de qualquer relação ou influência com o tempo que levariam a receber uma solução oficial, tornando-se mais difícil tomar uma decisão sobre que vulnerabilidades priorizar primeiro. Já com o SecScore, a maioria das vulnerabilidades vai tendencialmente decrescendo o seu risco ao longo do tempo.

Já sobre os resultados da avaliação do processo, com a execução deste e através das ferramentas de deteção de vulnerabilidades foi possível identificar e endereçar um total de 41544 vulnerabilidades no espaço temporal de duas semanas.

O sucesso dos resultados obtidos deve-se à inclusão dos conceitos ágeis da gestão de projetos que permitem uma produtividade muito maior e à automatização de algumas das fases do processo. O foco passou a ser a mitigação rápida das vulnerabilidades, ou seja, o trabalho de cibersegurança toma uma posição dominante para que este possa ser executado no menor tempo possível e assim abranger mais vulnerabilidades a serem mitigadas no mesmo espaço de tempo.

É compreensível, através de todos os *stakeholders* envolvidos de que há necessidade de mudar o rumo de como o CVSS está construído, graças às críticas que lhe são apresentadas. Este método proposto apresenta um ganho na priorização de vulnerabilidades, já que esta fica mais próxima ao ideal. A priorização abandona a análise estática do CVSS, passando a garantir a flutuação das suas características ao longo do tempo, especialmente com a priorização de formas mais graves de vulnerabilidades, como aquelas que possuem formas de exploração bastante desenvolvidas.

Palavras-chave: Vulnerabilidade, Gestão de Vulnerabilidades, Gestão de Vulnerabilidades Baseada em Risco, CVSS

Abstract

Vulnerabilities have been increasing over the years without signs of decreasing soon. With this exponential growth, it is important for organizations to define a vulnerability management plan to proceed with what should be done if they encounter a vulnerability. However, existing plans and metrics do not fit the current reality. Existing plans are not independent of vulnerability detection tools. The classification systems currently used (the most common is CVSS) fail to provide information on the variation of risk that a particular vulnerability entails for the organization. As this is not constant, being exceptionally high when there is a form of active exploitation, as well as its location in the network and business needs. SecRush presents itself as a new vulnerability management framework with a new risk-based vulnerability management process. It has a set of procedures inspired by agile methodologies to mitigate vulnerabilities and a new classification system - SecScore – able to provide a prioritization in context with the organization. SecScore varies its ranking through temporal factors (specific risk index depending on the organization's risk appetite and the availability of an exploit) and environmental factors (asset visibility to the external network and importance of the asset to the organization's mission). This project intends not only to contribute with a set of procedures independent of the security tools used but also to improve the currently existing classification systems for prioritization, which cannot adapt to the different contexts in which they are found.

Keywords: Vulnerability, Vulnerability Management, Risk-based Vulnerability Management, CVSS.

Contents

List of Figures

List of Tables

Acronyms

AI Artificial Intelligence

AC Attack Complexity

AR Availability Requirement

AV Attack Vector

AVI Attack, Vulnerability, Intrusion

DNS Domain Name System

CDC Centers for Disease Control and Prevention

CMDB Configuration Management Database

CPI Consumer Price Index

CR Confidentiality Requirement

CVE Common Vulnerabilities and Exposures

CVSS Common Vulnerability Scoring System

EPSS Exploit Prediction Scoring System

FIRST Forum of Incident Response and Security Teams

GDP Gross Domestic Product

IoT Internet of Things

IP Internet Protocol

IR Integrity Requirement

ISS Impact Sub-Score

IT Information Technology

LVU List of Vulnerable Users

MA Modified Availability

MAC Modified Attack Complexity

MAV Modified Attack Vector

MC Modified Confidentiality

ME Modified Exploitability

MI Modified Impact

MInt Modified Integrity

MISS Modified Impact Sub-Score

MPR Modified Privileges Required

MS Modified Scope

MUI Modified User Interaction

NIST National Institute of Standards and Technology

NVD National Vulnerability Database

NVL Normalized Vulnerability List

OS Operating System

OSINT Open Source Intelligence

PR Privileges Required

PVL Prioritized Vulnerability List

RBAC Role-Based Access Control

RC Report Confidence

RL Remediation Level

SIG Special Interest Group

UI User Interaction

URL Uniform Resource Locator

VMS Vulnerability Management System

Chapter 1

Introduction

Over time, vulnerabilities have grown exponentially in the digital world [12, 20], and with that, security efforts have accompanied this change in the security landscape in organizations. A vulnerability is a flaw in a computer system that can be exploited with malicious intent [43]. In software security, a vulnerability is a specific flaw or omission in software that allows attackers to perform a malicious action - to expose or alter confidential information, disrupt or destroy a system, or take control of a system or a computer software program [8]. For Michael Whitman [49], a vulnerability is an identified weakness in a controlled information asset resulting from the absence or inadequacy of controls. However, a vulnerability should be understood as a weakness (identified or not) present in an IT asset (in software or hardware) likely to be the target of a successful intrusion. The attacker that conducts the intrusion may be internal or external to the organization.

The AVI (Attack, Vulnerability, and Intrusion) model [43] is commonly used to explain the importance and interconnection of vulnerabilities with systems, which must be fault tolerant. This model assumes vulnerability as an internal fault and attack as an interaction fault that, by joining both, directly results in intrusion and, therefore, in component failure.

It is easy to see that the vulnerability only materializes in an intrusion when an attack exploits a vulnerability. When it exists but is not exploited, there is no intrusion. However, the existence of a vulnerability, by itself, constitutes a risk for the systems that have it, so it must be removed as soon as possible. In the presence of multiple vulnerabilities, it is necessary to reduce the risk of exposure by taking into account: their criticality, the form of the intrusion, the importance of the affected assets, and the impact on the organization's operation. Hence, it becomes vital to create a vulnerability management plan that, in addition to helping to prioritize the vulnerabilities to be mitigated, defines the entire action process, from detecting vulnerabilities to confirming their correction and mitigation.

For the sake of standardization, vulnerabilities are globally identified through a Common Vulnerabilities and Exposures (CVE) code. This code is assigned primarily by MITRE but also by a CVE Numbering Authority (CNA) [26, 27], an international group of vendors and researchers from numerous countries. Each CVE can be found in different databases with different Common Vulnerability Scoring System[1] (CVSS) [23] scores. This difference can occur due to different risk interpretations, depending on who assesses the vulnerability. This CVSS score classifies a vulnerability on a scale from 1 to 10, with 10 being a critical vulnerability. It has a set of pre-established metrics to help analysts classify the

[1] CVSS is maintained by FIRST through its Special Interest Group (SIG) [22]

vulnerabilities in a standardized way.

In addition, these classification systems have a very one-sided view, given that the metric obtained may not be the same in all organizational contexts. Just think that the vulnerability has an exploit code published on the internet, and any attacker can reproduce it on every organization. It needs to be prioritized above the others that do not present the same criticality. Also, the asset holding the vulnerability may not have the same external visibility (exposure of the IT asset to the network outside the organization), so the risk it represents for the organization will also differ. The risk this entails should vary according to this and other relevant factors for organizations, considering the topologies and asset organization structures. These factors are something that CVSS lacks in its evaluation and need to be improved, as it is the most used prioritization scoring system.

Although the existence of other classification systems [19], created to improve the prioritization and classification of vulnerabilities, these end up not succeeding as they are more theoretical and difficult to reproduce. Also, there are specific certifications of high weight worldwide. These certifications are the case of the banking sector [7, 13], which requires the use of CVSS to classify and prioritize vulnerabilities. Besides the certification requirements, the main problem with CVSS is its ability to continually evaluate the characteristics that change over time (temporal metrics) and the vulnerability inside its context in the organization (environmental metrics).

Any vulnerability management process must be specific to the organization, as each has its priorities and the process must address them [29]. In the banking scenario, for example, a vulnerability in the transfer service's operating system could jeopardize its main activity and bring a tremendous financial impact. However, the same vulnerability in the email marketing service of a supermarket chain will not wholly stop its operation, although it will still be a vulnerability to be addressed soon, never with the same urgency as a bank.

However, vulnerability management plans do not have a follow-up that focuses more on adapting to the current reality, so much of the information becomes obsolete and inconsistent. The vulnerability management plans that are publicly available are attached to a vulnerability detection tool and, most of the time, are only available through their paid platform. Vulnerability management plans are evolving to a more recent approach to introduce risk assessment in their prioritization. This approach brings a more comprehensive analysis of the real risk that vulnerabilities pose to organizations.

This work proposes a solution for the organization's vulnerability problem with a new vulnerability management framework that includes a new risk-based process and modified CVSS's temporal and environmental metrics. The motivation of this work is presented in Section 1.1 and the goals in Section 1.2. Also, the contributions of this work are in Section 1.3, and this document's structure is in Section 1.4.

1.1 Motivation

There is currently no way to calculate a vulnerability's direct impact on the organization. The classification is generic and devoid of the associated risk for those with it. CVSS itself is flawed in its current way of calculating. Several criticisms are made [13, 19] of the low variability of classification and the impossibility of being used to prioritize vulnerabilities effectively. The temporal metrics are watertight, and their classification is static, not evolving over time.

Additionally, environmental metrics do not assess a vulnerability in the context of an organization with several assets, with different impacts on business continuity and visibility to the external network. Both do not represent the actual risk of vulnerability in the organization's context. The risk it entails must vary according to relevant factors, taking into account the topologies of the computer systems and criticality for the organization.

CVSS is widely used for prioritizing vulnerabilities to mitigate in vulnerability management systems. Therefore, it is imperative to improve the current CVSS calculation to allow its use to be maintained. The challenge of creating a new formula or improving an existing one is to be able to transform the static calculation into a dynamic calculation. This improvement should not only characterize the vulnerability in the context of the asset and the organization but also the temporal aspects inherent to its evolution, particularly regarding the maturity of the code for its exploitation and the availability of a mitigation. So this work's first motivation goes toward improving the CVSS scoring system. Taking measures as soon as possible can define a greater or lesser degree of an organization being the target of a cyberattack since the time from the creation of a CVE and its publication in the NVD [28] (a public vulnerability database) is still high. However, it has decreased immensely over the years [34].

In addition to this, the existing vulnerability management plans are often associated with companies/products that sell vulnerability detection tools [4, 6, 10, 11, 14, 15, 29, 32, 44], in which their plans are centered around its detection tools. Not only that, but the literature has not presented other perspectives for a generic vulnerability management process that can be used by any organization, regardless of the tools chosen to implement it. More recently, still driven by these companies producing vulnerability detection tools, a new form of risk-based vulnerability management has emerged [5, 17, 30, 31, 33, 37, 38, 39]. This new system is based on a proactive approach to mitigate vulnerabilities, oriented toward the risk-based decisions they will bring to the organization. They will be taking into account all exposed assets that meet or do not meet compliance requirements through continuous and dynamic discovery to take quick action. The second motivation for this work results from seeking a complete and comprehensive vulnerability management plan that fits most organizations. By merging the best practices of the revised plans, it is possible to obtain a risk-based management plan that combines the advantages of the different existing models.

1.2 Goals

This dissertation's general objective is to create a new generation vulnerability management framework taking into account the temporal and environmental aspects during the vulnerability evaluation. The specific objectives are defined below:

I) Development of a new classification method capable of taking into account all the relevant factors in the current security of organizations, with the reformulation of the CVSS calculation method, to better address the temporal and environmental aspects in the vulnerabilities' classification;

II) Development of a common repository of vulnerabilities, with the respective web interface;

III) Design of a vulnerability management process, adopting the new classification proposed in I) and the repository developed in II).

1.3 Contributions

This dissertation proposes a new vulnerability management framework, of which the following contributions are listed:

I) A reformulation of the CVSS classification, capable of identifying and prioritizing vulnerabilities based on the importance of the vulnerable asset to the business and its external visibility. The reformulation of the classification also managed to make this classification more dynamic, allowing the vulnerability score to vary over time as its exploitation likelihood grows. This contribution is published in [35];

II) A new generation, risk-based vulnerability management process with six phases that track each vulnerability from its discovery to the end of its mitigation confirmation. This process promotes the rapid resolution of vulnerabilities by being inspired by agile project management methodologies. The process was built by merging different existing processes in order to bring together in a framework with the existing best practices on vulnerability management;

III) A vulnerability repository for near-real-time monitoring of vulnerabilities detected in the organization. This repository allows the listing of the various vulnerable assets associated with their asset owners, as well as monitoring and changing the vulnerability's resolution status. Due to the limited time imposed for this dissertation, only a subset of assets was considered for vulnerability analysis, namely desktops. Servers, IoT devices, or other vulnerable organization assets were not considered. The developed platform has a TRL of 3.

1.4 Document Structure

This document is organized into five chapters. Chapter 2 explains the related works, focusing on vulnerability management processes and vulnerability prioritization. Chapter 3 discusses the proposed framework in detail. Chapter 4 presents the results and a discussion about the experiments used to validate the proposed framework, including the modified CVSS calculation method and the vulnerability management tool. Finally, Chapter 5 presents the conclusions of this work and possible future works.

Chapter 2

Related Works

This chapter carries out a bibliographic analysis of vulnerability management, focusing on vulnerability management processes (addressed in Section 2.1), vulnerability prioritization (addressed in Section 2.2), risk-based vulnerability management (addressed in Section 2.3) and vulnerability management systems (addressed in Section 2.4).

2.1 Vulnerability Management Process

A vulnerability management process presupposes the creation of stages and mechanisms for effective management of the vulnerabilities that the organization may come across. These processes have undergone some changes over time to foresee all possible scenarios. However, the literature produced by companies in information technology and computer security (commonly known as whitepapers) is not consensual. Depending on the product they intend to sell and how the vulnerability management process can be integrated into them, the number of stages that the process imposes varies.

These processes are generally modeled in four, five, or six stages. To facilitate the organization and comparison between the different works presented here, this section is organized in the following subsections according to the number of stages in each of them.

2.1.1 Vulnerability Management Process in Four Stages

In the four-stage process, presented by Exabeam [14] and Rapid7 [32], the following stages are listed: "Identify", to identify vulnerabilities; "Evaluate", to determine the impact of each of the vulnerabilities; "Treat", to correct the targeted vulnerability; and, "Report".

This model is quite simple and rudimentary, as it does not consider many of the current problems that vulnerabilities entail, namely the impact of fixing all vulnerabilities, nor the confirmation that the risk has been mitigated by fixing the vulnerability. Furthermore, in its evaluation stage, this model does not reflect the existence or not of exploits.

The Exabeam process [14] was the shortest of all analyzed. In addition, the process already refers to other metrics to be analyzed with the CVSS classification, such as threat intelligence or the organization's business risk information, to evaluate better and prioritize vulnerabilities. This conjunction allows a more holistic and appropriate view of the organization. Moreover, the final Report stage has as its primary objective the creation of the idea of "Lessons Learned", something beneficial for situations where the team reencounters similar problems. However, there is only one "Report" stage, without any stage for

delegation of mitigation and transmission of the necessary information to the vulnerable asset owner. This “Report” stage is equivalent to a “Lessons Learned” stage, where the team reports vulnerabilities after remediation to help improve the organization’s security and responses in the future. The reporting of this information is also important. However, there needs to be a stage where the team starts delivering the mitigation tasks to the responsible vulnerable asset owners. Even the identification of vulnerabilities, which is done in the “Discovery”, is not done continuously, only once or on request. The Rapid7 [32] process is identical with the same stages name and order, and the stages present the same tasks.

2.1.2 Vulnerability Management Process in Five Stages

The five-step process is the most common and transversal and also the most adopted by technology companies that develop software for computer security [4], or even academic institutions [25, 41]. In this process, the stages of “Discovery”, “Assessment” or “Prioritization”, “Report” (sometimes alternated with “Planning”), “Correction”, “Verification” are listed.

This model encompasses a more holistic view of vulnerabilities since, in addition to confirming vulnerability mitigation, they have the formal prioritization of vulnerabilities, which is highly crucial to attack the highest priority vulnerabilities and, consequently, more critical to maintain the organization’s business continuity. It is also worth noting the importance of reporting discovered and prioritized vulnerabilities before taking corrective measures. In a large organization, it is not always the responsibility of the same company unit to discover vulnerabilities and manage the system to resolve and mitigate detected vulnerabilities.

In the Team Ascend [4] process, the “Discovery” stage tries to get all connected devices continuously, which allows you to identify all the assets present on the network without carrying out a manual inventory. There is a focus on gathering as much information as possible about the equipment. This focus allows knowing its importance, the tasks performed, and the identification of criticality for the business. One of the relevant positive points is the “Report” stage before “Remediation”, which allows an organization to create a report with instructions for quick remediation (step-by-step). However, this process lacks a final reporting stage for management and does not refer to the ongoing discovery of assets and vulnerabilities.

For the Fordham University process [41], a minimum deadline for addressing vulnerabilities is defined, which allows the establishment of a deadline for each vulnerability criticality. However, the “Discovery” process only follows technological tools, one of which is Qualys, which defines the priority of the security levels to be addressed. In addition, mitigation always needs approval, which can be seen as an increased risk in more urgent situations.

The University of Miami [25] process was the least detailed of all those analyzed, derived mainly from the “Discovery” stage, which is periodic and not continuous, and the actions to be taken according to the possible scenarios are not parameterized.

2.1.3 Vulnerability Management Process in Six Stages

The six-stage process is the most extensive, having been the one presented by relevant international organizations, whether in public administration, such as the CDC [9], or in significant technology and

telephone companies, such as AT&T [6]. This process describes the stages of “Discovery”, “Prioritization”, “Assessment”, “Remediation”, “Verification” and “Report”.

The “Report” stage may come at the end of [6] or in the middle of the process [9]. This difference will depend on the organization that proposed the process. It all boils down to the importance and activity developed at this stage within each vulnerability management process.

In this model, there is already a clear distinction between prioritization and assessment. The former prioritizes all vulnerabilities discovered in the first stage of the process. The latter aims to establish a threshold value for the risk. In this case, any vulnerability exceeding the defined threshold should be mitigated according to the prioritization process. For vulnerabilities that do not reach this risk value, they must be abandoned and not addressed, as the organization assumes the entire risk that the vulnerability may entail, should it become successful.

One of the significant advantages of this six-stage process by AT&T [6] is that the “Discovery” is a continuous, regular and automatic stage. This automation, in addition to saving human resources in carrying out the tasks, also makes the information produced more up-to-date and adequate to the temporal reality when consulting the data due to the ease of performing tasks over the network than when performed manually by humans. The other advantage would be the inclusion of a “Verification” stage after the “Remediation” stage, as it is vital to ensure that the controls and the resolution carried out are correctly implemented to mitigate and eliminate the risk of treated vulnerability. As the authors of this AT&T process demonstrate, the “Prioritization” stage prioritizes assets in groups. The prioritization is based on its business criticality. This prioritization is because not all assets are of equal importance to the organization, or their absence does not compromise business continuity. Also, the risk level is continuously evaluated. This continuous evaluation ensures the limit for which the risks of the detected vulnerabilities are accepted or treated. The evaluation also ensures that the agreement with the possible changes to the organization’s risk appetite is always present. However, this process has a significant flaw: failing to monitor vulnerabilities without a patch. These types of vulnerabilities, as they can pose a high risk to the organization, must have a form of action previously stipulated in the process. Still, the “Report” stage produces artifacts only aimed at executive managers, without any concern for middle managers, which can become flawed for large organizations.

In the six-stage process of CDC [9], in the “Discovery” stage, the inventory of all assets with their respective vulnerabilities is carried out. This approach can be pretty interesting and relevant to this process, as it allows vulnerabilities to be automatically prioritized. Factors such as the risk they produce or the affected assets are essential so that later, in the “Remediation” stage, this is done based on the loss of risk that this produces for each group of business assets. For the same reasons as AT&T [6], the periodic review of the risk threshold can be good for updating the organization’s risk appetite. However, the “Discovery” stage of the CDC process [9] is cyclic and not continuous, which does not allow the permanent update of existing vulnerabilities, and this update is subject to change only when a cycle is closed. Furthermore, the prioritization refers to assets and not vulnerabilities; this will not make much sense, as the process focuses on vulnerabilities, which in turn may affect the assets and not the other way around, as what can affect the targeted assets are, in fact, the vulnerabilities. The described “Report” stage is not intended to guarantee any help to the stage that follows it, making it a waste of time since the content constructed here did not serve to improve and facilitate the subsequent work.

Table 2.1: Summary of the CVSS Metrics

Base Metrics	Exploitability Metrics	Attack Vector (AV) Attack Complexity (AC) Privileges Required (PR) User Interaction (UI)
	Scope	
	Impact Metrics	Confidentiality (C) Integrity (I) Availability (A)
Temporal Metrics	Exploit Code Maturity (E) Remediation Level (RL) Report Confidence (RC)	
Environmental Metrics	Security Requirements (CR, IR, AR) Modified Base Metrics	

2.2 Vulnerability Prioritization

The prioritization of vulnerabilities is a problem continuously presented in the literature, where several solution proposals exist.

Miura-Ko *et al.* [21] presented a proposal based on computing infrastructure, in which they presented three methods to prioritize vulnerabilities and assets : density prioritization - the nodes are prioritized by the number of neighbors they have; source prioritization - a higher priority is given to nodes that serve as a source for an attack; and, finally, type prioritization - vulnerabilities are categorized as low/medium/high risk, and priority is given to high risk.

However, it was necessary to assume that the known vulnerabilities revealed how easily a node could be attacked and penetrated. The global importance of a node is indicative of the damage it could entail if a vulnerability is exploited, and also the network topology, which also influences the vulnerability of a particular node, given that more vulnerable nodes make neighboring nodes more vulnerable.

Currently, prioritization has been achieved through the CVSS classification, which describes a numerical value intended to assist organizations in assessing vulnerabilities to be corrected first [22].

The numerical value obtained is based on three major types of metrics: base metrics, temporal metrics, and environmental metrics, as summarized by the Table 2.1, according to the FIRST specification [23], and detailed in Appendix A.

The CVSS classification, however, is considered outdated and ineffective nowadays [13, 19, 36] because it lacks several criteria that are important for organizations and are not reflected in the final classification, making the value obtained little coherent with the organizational reality.

2.2.1 CVSS: Common Vulnerability Scoring System

CVSS presents a classification and prioritization system capable of reflecting the severity of vulnerabilities. The final numerical classification it produces can be qualitatively represented in four categories: low, medium, high, and critical [22]. It has three groups of extensive metrics: base metrics, temporal metrics, and environmental metrics [23].

In the base metrics, those that unequivocally characterize a vulnerability get split into exploitability

metrics (Attack Vector, Attack Complexity, Privileges Required, and User Interaction), impact metrics (Confidentiality, Integrity, and Availability Impact), and Scope.

Temporal metrics measure the current state of exploit techniques or code availability, the existence of patches or a workaround, or the certainty of the vulnerability description. It includes three metrics: Exploit Code Maturity (E), Remediation Level (RL), and Report Confidence (RC).

Environmental metrics allow the customization of the CVSS score, depending on the importance of the affected IT asset to a user's organization, measured in terms of implemented complementary and alternative security controls, confidentiality, integrity, and availability. These include security requirements (Confidentiality, Integrity, and Availability Requirements) and Modified Base Metrics, which allow individual base metrics to be replaced by the specific characteristics of a user's environment.

Today, the CVSS assesses the time factor through the previously mentioned static metrics (Exploit Code Maturity, Remediation Level, and Report Confidence). This assessment can not always be accurate since, over time, the vulnerability context can evolve and represent a greater (or lesser) risk [13]. The value should vary and increase when a vulnerability poses a greater risk to the organization. This situation occurs when the vulnerability has an exploit code whose maturity is high or unknown. When high, it can bring very high risk to the organization, and it is more likely to be attacked. The absence of information on the maturity of this code should always be treated as high maturity (E) since, in principle, it is considered the worst-case scenario, in line with the existing coefficients of the CVSS.

On the other hand, the environmental metrics possess more information and can reassess the information security properties for each vulnerability. However, it is not enough since it is still necessary to prioritize vulnerabilities for their visibility to a user of the external network (the network where most cyberattacks occur). It is also necessary to prioritize vulnerabilities by their degree of importance to the organization, namely the criticality that assets have for the maintenance of the business and operation of its activity.

Considering these CVSS constraints, Howland presents that "there are no indicators of effectiveness tests to be carried out" [13] for CVSS and that "the standard is poorly maintained". In his article, where he criticizes CVSS as a way of prioritizing vulnerabilities and how it produces values for the criticality of vulnerabilities, he presents the importance of the vulnerability context. The change to CVSS in version 3.1 no longer considers it a risk score, making it a "clear attempt to shirk CVSS's responsibility as a means of measuring risk, not because it is not a risk score, but because it performs poorly", as stated by Howland. His work provides an example of a pharmaceutical company that assigns a "High" rating to two assets that support different business areas for the Availability parameter. However, one asset belongs to the drug manufacturing operation and another to a web server component. In a real context, there would be a natural prioritization to prefer business continuity through the core business. With CVSS, this becomes opaque, and both vulnerabilities would take the same category. The present work considers the need to adapt the CVSS in order to better address the aforementioned constraints identified in the literature. To this end, it proposes an improvement in the computation of the CVSS temporal and environmental components.

Spring et al. [36] confirm that there is no evidence that CVSS takes into account relevant contextual information, such as the type of data processed, the system's operation, the context of the use of the vulnerable asset, and the material consequences of an attack in case of success. According to them, since

2007, the major criticisms presented have been grouped as “Failure to account for context (both technical and human-organizational)”, “Failure to account for material consequences of vulnerability (whether life or property is threatened)” and “Operational scoring problems (inconsistent or clumped scores, algorithm design quibbles)”.

The discredit given to the rating [38, 39, 45] is due to the fact that currently, there are 76% of vulnerabilities with a CVSS rating of 7 or higher, especially without published code to exploit the vulnerability [37]. Despite the same high-risk classification, some of these vulnerabilities may represent little risk to the business, making it important to evaluate each vulnerability in its context.

Luca Allodi [1] justifies this phenomenon by presenting evidence that the exploitation of vulnerabilities is described by a heavy-tailed statistical distribution and hypothesizes that the vulnerability distribution may follow the Power Law model, comparing it with the Log-Normal and Exponential. The vulnerabilities that his investigation focused on were just vulnerabilities exploited in the wild. In addition, he verified that a small fraction of vulnerabilities is responsible for a large majority of attacks against software, confirming what is present in [39]. The hypothesis presented was inconclusive due to the low number of vulnerabilities present in the dataset, despite being, according to the author, the most comprehensive at the time of publication. Therefore, it becomes a priority to define alternative ways of calculating an organization’s vulnerability risk.

2.2.2 EPSS: Exploit Prediction Scoring System

The literature also presents some proposals for new classification systems. However, the one that stands out the most is the Exploit Prediction Scoring System (EPSS) [19]. This new scoring system is presented as being more “simple to implement”, more understandable, objective, and transparent. The data used are all public and allow the exploitability’s prediction of a vulnerability within 12 months after the confirmation of the exploit from the software vendor. It is the first open and data-driven framework for vulnerability assessment.

According to the authors, precision measures the efficiency of the remediation strategy. It is computed as the number of true positives divided by the level of effort. The level of effort is the sum between true positives and false positives. Compared to CVSS, EPSS’s efficiency is extremely high for the highest-rated vulnerabilities. Ratings starting at 4 on the CVSS scale, the effort is reduced to amounts greater than 60%. At the last level of the same scale (10), it reaches a reduction of 86%.

However, even though the calculation of the coefficients and the formula are entirely transparent, their understanding and applicability require a robust knowledge of statistics, as the metrics the authors use to prove their improved capacity against CVSS are advanced.

In addition, its implementation depends on creating and maintaining machine learning models to obtain better results in predicting the exploitability of each vulnerability. A mandatory CVE identifies each vulnerability. This dependency makes it difficult to use and apply zero-day vulnerabilities, internal vulnerabilities, misconfigurations, or no official identification of a CVE. This new form of calculation presented by the authors is helpful for companies that only use third-party software and can prioritize the patches to be applied with a CVE identifier linked to it, provided that the data collected corresponds to the organization’s reality. There is the possibility of dealing with and distinguishing threats that are not so obvious to the organization’s business. The authors indicate that this is an “open, data-driven

effort". However, there is little of being open since it does not document which sources were used. More specifically, the authors do not specify the data used to train the machine learning models, the parameters used, and the code necessary to generate the exploitability prediction models. All the essential operational details are a "black box" of information that remains to be clarified, and they are necessary to put this new form of calculation into practice.

Like CVSS, EPSS also has a Special Interest Group (SIG) in FIRST [24], which follows the development of this new alternative model to CVSS.

2.2.3 Other scoring systems

Debricked, a Swedish cybersecurity solutions company, has a scoring system in one of its products for analyzing secure code in Git repositories, debAI [3]. Note that this is a different approach when compared to CVSS, in which the score of a vulnerability remains the same for any organization, regardless of its context. In this classification, the priority levels of "Confidentiality", "Integrity", and "Availability" can be changed, as well as the "Attack Vector" (where it is indicated which of the possible values the organization pays more attention to), "Exploitability" and "Required Privileges". All parameters except the "Attack Vector" can take values between 0 and 10.

Jacobs *et al.* [18] presented a proposal to prioritize vulnerabilities by predicting their exploit. Unlike other literature, the analyzed variable is "exploits in the wild" instead of "published exploits" in line with [1]. The proposed model presented three practical strategies for vulnerability remediation: CVSS scores, published exploits, and vulnerability attributes. They evaluated each strategy using a rules-based approach and a machine learning model. Ultimately, they built a machine learning model with the obtained dataset. The proposal presented good accuracy, efficiency, and coverage results compared to CVSS. However, it does not demonstrate an easy reproducibility for any organization to apply in its vulnerability management process. In addition, the transparency of the construction of the machine learning model and the data used to train the model are other negative factors. Only the sources used are mentioned, but all the processing and how the authors got there are completely opaque.

2.3 Risk-Based Vulnerability Management

The most recent articles indicate a change in the vulnerability management paradigm. The old approach followed a prioritization based on the value of the CVSS classification, and now, the focus shifts to the impact that vulnerabilities have on business continuity and, therefore, the risk they produce in organizations.

According to [37], a risk-based vulnerability process can be described generically in five stages:

- **Discovery:** The discovery stage consists of the discovery of assets and the vulnerabilities of these same assets, in a continuous way, similar to the discovery stage of the configuration and management of systems, which only identifies the assets connected to the network [42].

- **Assess in context:** Assessing each vulnerability allows the organization to assess them in context with the business as well as the overall threat landscape. In addition to classifying them in context,

the potential for exploitation is also evaluated. Then, the likelihood of the intrusion occurring (exploit) is determined, and the impact on the business is considered.

- **Prioritization:** Through automation and possibly artificial intelligence (AI) solutions, prioritizing vulnerabilities is facilitated when determining which ones pose the highest risk for the organization.
- **Remediation:** Remediation presupposes sending a short list of vulnerabilities to be corrected by the vulnerable asset owners. It is even articulated with these asset owners what is the most critical thing to be completed first.
- **Measurement:** It forces the metrics calculation to know what to improve and what positive or negative changes have occurred.

The vulnerability management process proposed in this work can be classified as risk-based. However, the proposed process has six stages, unlike the definition found in [37]. The additional stage aims at delegating the responsibility of the mitigation actions to the vulnerable asset owners and informing them.

2.4 Vulnerability Management Systems (VMSs)

Some solutions for vulnerability management systems were presented recently. The VMC [47] was the closest to the objectives of this work. The authors of this tool even carried out an analysis of selected metrics to facilitate the detection of critical vulnerabilities [46]. This tool integrates with vulnerability detection tools directly for greater automation; it connects to a Configuration Management Database (CMDB), where you can view all the organization's assets and integrate the asset management process with vulnerability management. In addition, VMC already has dashboards and reports ready for a clear overview of vulnerability management. However, despite being very promising, this tool proved to be quite challenging to deploy, and the integrations were very time-consuming for the limited time of this project, so this work followed up with a customized solution.

There were two more VMS solutions, ArcherySec [2] and Orchestron [48]. Both were good tools, but they focused on application security and not on managing all possible organizational vulnerabilities. Only ArcherySec gets maintained until the publication of this dissertation. Orchestron does not receive a code update since 2020.

Chapter 3

Proposed Framework

The simultaneous use of several vulnerability management tools produces a technology overload, ultimately resulting in unaddressed vulnerabilities, lack of prioritization, high costs, disruptions, and management challenges [40]. With this in mind, the vulnerability management plan presented in this chapter is a generic plan suitable for any institution that intends to manage its vulnerabilities. As discussed in Chapter 2, the CDC [9] plan seems, at the moment, to be the most appropriate to establish itself as a basis for this project. Not only because it is the most complete in the stages it presents but because of its suitability regarding the order of its stages to most organizations. By adopting the CDC plan, the model will have six stages. This plan aims to improve some aspects of its microstructure that lack activities to create a full risk-based vulnerability management. Regardless of the processes reviewed and their extension of stages, most of them provided positive characteristics in their contexts. These characteristics improve the construction of a comprehensive model with a flexible structure for any organization and meet the necessary conditions.

3.1 Process

The proposed vulnerability management process has six stages, in the following order, according to Figure 3.1. Each stage is considered a domain and has a set of tasks to be carried out. They cannot start if the previous stage has not terminated and provided an output. In the following subsections, each stage will be fully detailed.

3.1.1 Discovery

The vulnerabilities that affect an organization's services and corresponding assets are pervasive and variable. Thus, in the initial stage, Discovery will allow better identification of vulnerabilities and provide a quick resolution since the vulnerable asset is already identified.

It is essential to be aware of the different components of information systems and their configuration. Without suitable discovery mechanisms, it is not possible to successfully and effectively manage and administer a vulnerability management process. These mechanisms are crucial for its excellent performance, and because it is the initial stage, it will affect all the others.

Discovery is the process of identifying all assets managed by an organization and associating them with each of these assets' vulnerabilities. These assets can be of two types, physical assets (for example,

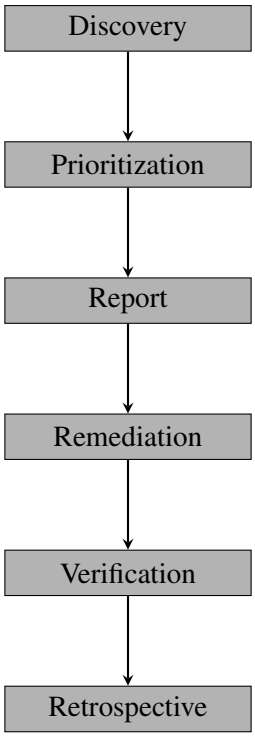


Figure 3.1: Representation of the process stages

computers or servers) or virtual assets (for example, machines virtualized on servers, either on an organization's premises or in the cloud). The information collected should focus on both types of information: the identification of all hardware and software assets and the topology of these assets – that is, how they are interconnected. The discovery procedure should be the same for all machines. The responsibility should be associated with whoever manages these assets in the organization (aka "owner of the asset"). Figure 3.2 presents a representation of the Discovery stage. Table 3.1 specifies the inputs and outputs typically expected in this stage.

Table 3.1: Input and output of the Discovery stage

Input	N/A
Output	Normalized Vulnerability List (NVL). List of Vulnerable Users (LVU).

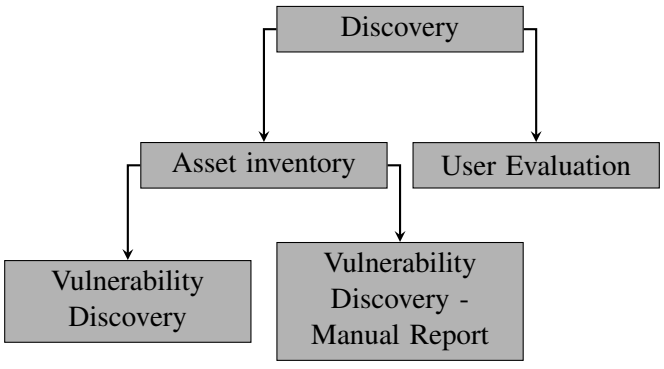


Figure 3.2: Representation of the Discovery stage

In this first stage, the cybersecurity and IT teams must collect information about which devices are

connected to the network. For the assets that belong to the organization, it is necessary to collect information about them[1], the software and hardware that those machines have and their respective versions so that it is possible to identify vulnerabilities, as described in Table 3.2.

After this information is cataloged, data collection must be continuous. The continuous collection means always, at all times, and not periodically with defined time intervals. The continuous collection makes it possible to deal with rapid changes and, consequently, reduce the time in which possible vulnerabilities may be in assets that would only be detected in a future analysis. In this project, which is based on the CDC [9] macrostructure, the idea of making the process continuous, regular, and automatic derives from the design of the equivalent stage in the AT&T process [6].

Table 3.2: Asset inventory

Sub-activity	Asset inventory
Scheduling	Continuous activity
Description	All network assets must be inventoried, preferably in an automated and continuous way, so that new assets can be detected on the network in the shortest possible time and, consequently, be able to adapt to the change (in the topology) quickly.
Output	Asset's list on the organization's network

Then, still in the first stage, and based on the information already collected, Vulnerability Discovery must be initiated, as described in Table 3.3. For this purpose, vulnerabilities must be identified by the vulnerability detection systems and centralized in a single repository. This repository will also include information collected based on communications or notifications from the manufacturers, emails from software vendors, secure code analysis, penetration tests, internal audit findings, automated scanning tools, risk management, reports produced, and Open Source Intelligence (OSINT). The entire list of vulnerabilities with the associated assets will be passed on to the next stage of the vulnerability management process, the Prioritization.

Table 3.3: Vulnerability Discovery

Sub-activity	Vulnerability Discovery
Scheduling	Continuous activity
Input	Assets list on the organization's network
Description	Based on the asset list, conduct continuous tests that allow the identification of vulnerabilities in each asset. This information can be collected manually or with the help of commercial or open-source tools. If the organization has more than one of those mentioned above, the information must be standardized to a central database, with identical fields to be prioritized later.
Output	Normalized Vulnerability List (NVL)

There may be vulnerabilities that cannot be discovered by automated means, and the organization is subject to the information it receives from vendors/providers or software manufacturers. In this situation, because there are several IT assets, and information about their vulnerabilities does not always go to the same channel, the asset owner is responsible for communicating this information to the entity

[1] Using SNMP or similar protocols.

responsible for the vulnerability management process and, consequently, carrying out the sub-activity 3.4 - Vulnerability Discovery - Manual Report.

Table 3.4: Vulnerability Discovery - Manual Report

Sub-activity	Vulnerability Discovery - Manual Report
Scheduling	Punctual activity
Input	Assets list on the organization's network
Description	It may be impossible to automate data collection, and this may be the case for audit findings, risk management, internal reports, penetration tests, code reviews, emails from service providers, or notifications from software manufacturers. In these cases, the asset owner must publish the information into a central database and, like the other vulnerabilities, get it prioritized.
Output	Normalized Vulnerability List (NVL)

Finally, to ensure the safety of those considered the most critical assets of the organization - the people - the entity responsible for the vulnerability management process conducts at least a quarterly awareness activity. User Evaluation aims to assess the ability of an organization's employees to be aware of the dangers of the digital world and sensitize them to possible computer attacks intended to steal data, money, or both. Table 3.5 describes this sub-activity in detail.

Table 3.5: User Evaluation

Sub-activity	User Evaluation
Scheduling	Quarterly activity
Input	Assets list on the organization's network
Description	It is important to pay attention and importance to the Social Engineering to which users are subject since, despite being the organization's most important assets, they are the most vulnerable to this type of attack. Thus, users must be evaluated against different types of possible attacks to find out if, in a real situation, they would know how to react correctly (for example, clicking on links from unknown or dubious sources). If not, employees must undergo practical training to prevent real-scale hazards.
Output	List of Vulnerable Users (LVU)

3.1.2 Prioritization

Prioritization is the second stage of the vulnerability management process and aims to prioritize the vulnerabilities found at the end of the Discovery stage. Many current systems already allow a link between the existing vulnerability and its registration in an open catalog. An example is the Common Vulnerabilities and Exposures (CVE) of the "MITRE Corporation".

However, the growing number of high-risk vulnerabilities ends up discrediting the current process of attributing risk to vulnerabilities. Therefore, it is necessary to assign a new risk classification for the vulnerabilities that could affect the organization through a classification system. This classification system takes into account the different information collected in different systems of the organization to classify each vulnerability according to the risk it produces.

At this stage, the vulnerability management process presented here will have its classification system,

which will take into account the various information collected in the different systems of the organization to classify each vulnerability in order to reflect the risk that it produces specifically in the organization where the vulnerability is present. This idea is based on the work also presented by AT&T [6], which prioritizes the organization's assets by groups, similar to organizing assets by recovery groups. However, the interest in this stage becomes even broader since, in addition to the asset's criticality, vulnerability is analyzed in context with the asset itself and the system as a whole. Table 3.6 specifies the inputs and outputs typically expected in this stage.

Table 3.6: Input and output of the Prioritization stage

Input	Normalized Vulnerability List (NVL).
Output	Prioritized Vulnerability List (PVL).

Previously identified and classified vulnerabilities should be ordered based on their risk to the organization. Currently, the ordering follows the CVSS [22] classification in most organizations. However, as Exabeam [14] proposes, it is important to go further since this classification has already proven inefficient. The ordering obtained here is based on a specific classification system that is defined by its own equations, described in Section 3.2. These equations will calculate the risk of vulnerability (singular) for the organization. For that, it must be applied to all vulnerabilities. The result obtained is a list of vulnerabilities ordered to be remedied. Table 3.7 describes this sub-activity.

Table 3.7: Automatic Prioritization

Sub-activity	Automatic Prioritization
Scheduling	Continuous activity
Input	Normalized Vulnerability List (NVL)
Description	Vulnerabilities are prioritized based on their risk to the organization. The ordering is based on a specific classification system defined by equations in Section 3.2.
Output	Prioritized Vulnerability List (PVL)

3.1.3 Report

The Report stage is the third stage of the vulnerability management process. It intends to gather the vulnerabilities already prioritized by the previous stage - Prioritization - to organize and document them so that the teams responsible for the targeted assets can act and solve the vulnerabilities. Figure 3.3 presents a representation of the Report stage. Table 3.8 specifies the inputs and outputs typically expected in this stage.

Table 3.8: Input and output of the Report stage

Input	Prioritized Vulnerability List (PVL).
Output	Report for the Asset Owner. Emergency Report for the Asset Owner.

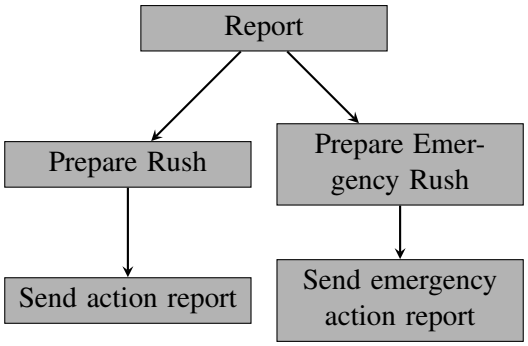


Figure 3.3: Representation of the Report stage

This stage introduces the concept of rush, a set of vulnerabilities to be resolved within a specified timeframe measured in weeks to make the current vulnerability management process as agile as possible. Each rush will be associated with a group/department, so the assignment of tasks is done only with the vulnerabilities whose owner(s) of the asset(s) are in the group/department. Table 3.9 describes the sub-activity of the rush preparation.

Table 3.9: Prepare Rush

Sub-activity	Prepare Rush
Scheduling	Weekly activity
Input	Prioritized Vulnerability List (PVL)
Description	Each period, each team will have allocated a set of vulnerabilities to carry out the rush. This list will be based on the Prioritized Vulnerability List and strip the most critical vulnerabilities from a particular group/department. For consistency, all periods should have the same number of new vulnerabilities to mitigate. All the vulnerabilities that may have been left in the previous rush are back in the new one.
Output	Vulnerabilities to add in the action report

For situations of extreme urgency, when the period of a normal rush is insufficient to resolve and reduce the organization's exposure to the risk of one or more specific vulnerabilities, an emergency rush must be prepared, which has increased priority, overriding a normal rush, as described in Table 3.10.

Table 3.10: Prepare Emergency Rush

Sub-activity	Prepare Emergency Rush
Scheduling	Punctual activity
Input	Prioritized Vulnerability List (PVL)
Description	In exceptional situations, an emergency rush can be prepared when the criticality of the vulnerability(s) is so high that it can severely affect the organization's activity if corrected within a period equal to that of a normal rush.
Output	Vulnerabilities to add in the emergency action report

Each rush will materialize in a report containing identified vulnerabilities and the highest priority to be resolved. These reports should be as succinct as possible so that the document is easy to put into practice and valuable to anyone addressing it. Each report, like the rush, only contains the vulnerabilities of the assets assigned to each asset owner, with a recommended step-by-step risk mitigation plan. This

is described in Table 3.11, for a regular rush, and in Table 3.12 for an emergency rush.

Table 3.11: Send action report

Sub-activity	Send action report
Scheduling	Periodic activity
Input	Vulnerabilities to add in the action report
Description	Prepare a report based on the vulnerabilities indicated by the rush, which should include the recommended plan to mitigate the risk step-by-step.
Output	Action Report for the specific group/department

As before, the emergency report contains the vulnerability(s) identified in the emergency rush and will override the normal report.

Table 3.12: Send emergency action report

Sub-activity	Send emergency action report
Scheduling	Punctual activity
Input	Vulnerabilities to add in the action report
Description	Prepare a report based on the vulnerabilities indicated by the emergency rush, which should include the recommended plan to mitigate the risk step-by-step.
Output	Emergency Action Report for the specific group/department

3.1.4 Remediation

Remediation is the fourth stage of the vulnerability management process and is intended to mitigate and reduce the risk of an organization's exposure to vulnerabilities.

This stage, represented in Figure 3.4, comprises four types of vulnerabilities, critical and normal with a patch, without a patch, and the ones for misconfiguration. As for the first two, these may have a patch, so their operationalization will differ. Table 3.13 specifies the inputs and outputs typically expected in this stage.

Table 3.13: Input and output of the Remediation stage

Input	Action Report for the specific group/department. Emergency Action Report for the specific group/department.
Output	Remedied Vulnerability(ies).

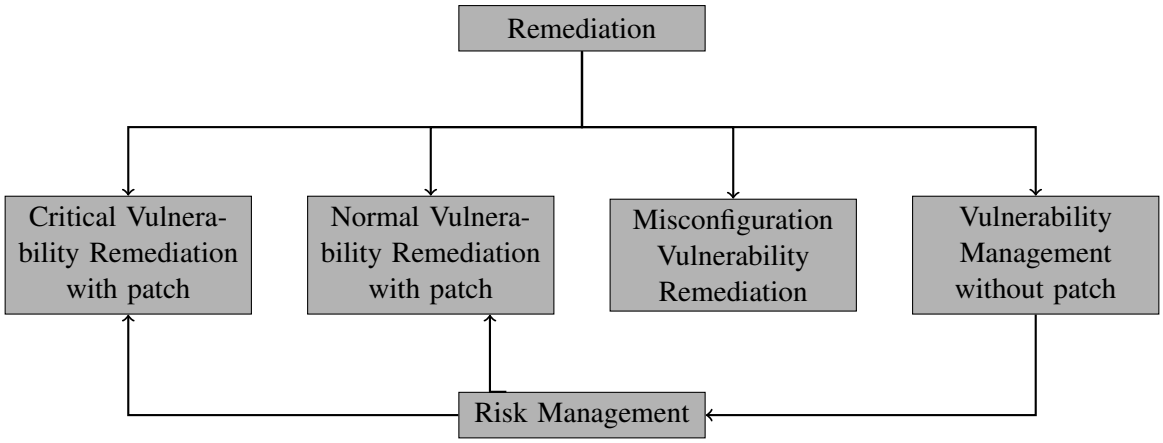


Figure 3.4: Representation of the Remediation stage

Table 3.14: Critical Vulnerability Remediation with patch

Sub-activity	Critical Vulnerability Remediation with patch
Scheduling	Punctual activity
Input	Emergency Action Report for the specific group/department
Description	A critical vulnerability with an enforceable patch must have its remediation completed within the maximum timeframe provided by the Rush. A critical vulnerability is considered to have a rating equal to or greater than 9 or, in general, an extremely high risk to the organization's Information Systems or the entity's image and external reputation.
Output	Remedied Vulnerability

Table 3.15: Normal Vulnerability Remediation with patch

Sub-activity	Normal Vulnerability Remediation with patch
Scheduling	Punctual activity
Input	Action Report for the specific group/department
Description	A normal vulnerability with an enforceable patch should see its remediation completed within the maximum timeframe provided by the Rush and, if possible, coordinate its remediation in conjunction with the current patching cycle. A normal vulnerability is considered to have a rating lower than or equal to 9 or, in general, low risk to the organization's Information Systems or the entity's image and external reputation. Still, consideration should be given to the possibility that the vulnerability could be exploited before the organization remedies it or that an exploit exists with a significantly reduced difficulty for the attacker.
Output	Remedied Vulnerability

The teams responsible for the affected asset(s) must follow the information provided in the Report stage. Instructions for installing a compatible patch that resolves the vulnerability must be followed in the case of patched vulnerabilities, as described in Tables 3.14 and 3.15. The same goes for misconfigurations that pose security problems, as described in Table 3.16. As for unpatched vulnerabilities, they must be monitored as well as the affected system(s) to prevent the problem from escalating, always controlling and preventing the worst-case scenario, as described in Table 3.17.

Table 3.16: Misconfiguration Vulnerability Remediation

Sub-activity	Misconfiguration Vulnerability Remediation
Scheduling	Punctual activity
Input	Action Report for the specific group/department or Emergency Action Report for the specific group/department
Description	A lousy configuration can lead to a larger window of vulnerabilities. In particular and previously identified situations, it is necessary to change the current settings. Examples of this are origin or weak passwords. Thus, this type of vulnerability remediation intends to act on the current settings and change them to remove the vulnerability from the affected asset(s).
Output	Remedied Vulnerability

Table 3.17: Vulnerability Management without patch

Sub-activity	Vulnerability Management without patch
Scheduling	Punctual activity
Input	Action Report for the specific group/department or Emergency Action Report for the specific group/department
Description	A vulnerability that does not have a patch to be applied is complex, as it requires constant monitoring of the affected asset(s) and, possibly, service failure or temporary unavailability of the same whenever it is plausible and justifiable to proceed in this way. The actions to be taken when monitoring are very focused on the behavior of the affected asset(s) and on the control of their access to them. However, it is not possible to establish a directive/norm on how to proceed, as vulnerabilities are multiform. Nevertheless, following the organization's best practices, recommendations and well-established and tested standards for this purpose is a way of proceeding in these types of vulnerabilities. This sub-activity is only considered complete when the manufacturer of the affected asset(s) implements and makes available a patch for this vulnerability.
Output	Remedied Vulnerability

3.1.5 Verification

Verification is the fifth stage of the vulnerability management process. It aims to ensure that the measures applied in the previous stage, Remediation, have been successfully implemented and that the organization's cybersecurity team validates these implementations.

Communication is of utmost importance in this stage, as it will drive the rapid resolution of vulnerabilities. Figure 3.5 presents a representation of the Verification stage. Table 3.18 specifies the inputs and outputs typically expected in this stage.

Table 3.18: Input and output of the Verification stage

Input	Remedied Vulnerability(ies).
Output	Remedied and Validated Vulnerability(ies).

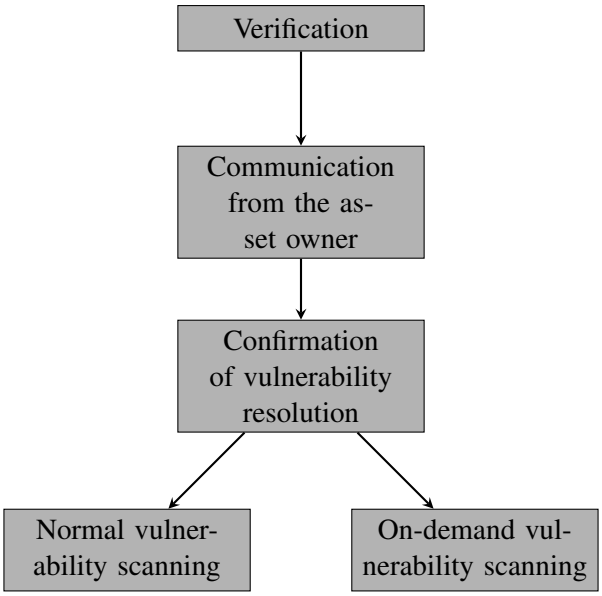


Figure 3.5: Representation of the Verification stage

Table 3.19: Communication from the asset owner

Sub-activity	Communication from the asset owner
Scheduling	Punctual activity
Input	Remedied Vulnerability
Description	In order to complete the remediation task, the asset owner must communicate to the entity responsible for the vulnerability management process that the vulnerability(s) are resolved, providing evidence of the change.
Output	Remediated Vulnerability Notification

Table 3.20: Confirmation of vulnerability resolution

Sub-activity	Confirmation of vulnerability resolution
Scheduling	Punctual activity
Input	Remediated Vulnerability Notification
Description	After sending the evidence(s) of the asset owner to the entity responsible for the vulnerability management process, the latter will validate whether it is valid to resolve the vulnerability(s). If so, this entity will confirm the resolution, and the vulnerability(s) will be considered solved.
Output	Vulnerability to Confirm

In the Verification, the cybersecurity team members, and those responsible for the vulnerability management process, will verify the remediation of the identified vulnerabilities, according to Figure 3.5. Table 3.19 presents the sub-activity "Communication from the asset owner" in detail. Table 3.20 describes what should be done to confirm a vulnerability's resolution. For that, the cybersecurity team can go two separate ways: normal vulnerability scanning, described in Table 3.21 or on-demand vulnerability scanning, described in Table 3.22.

Table 3.21: Normal vulnerability scanning

Sub-activity	Normal vulnerability scanning
Scheduling	Daily activity
Input	Vulnerability to Confirm
Description	This sub-activity includes all assets whose criticality for the business is medium or lower (like the ones treated in a normal rush), and the vulnerability does not present a high risk to the organization. Thus, the scan can be performed on its next scheduled occurrence (usually daily), and if it does not identify the vulnerability addressed, it is considered remedied and validated.
Output	Remedied and Validated Vulnerability

Table 3.22: On-demand vulnerability scanning

Sub-activity	On-demand vulnerability scanning
Scheduling	Punctual activity
Input	Vulnerability to Confirm
Description	This sub-activity must be requested by the owner of the asset(s), when applicable, using the internal processes to request a new scan. This sub-activity is only valid for assets whose criticality is very high or the vulnerability also has a high risk for the organization. Hence the need to quickly confirm that the vulnerability previously identified and resolved by the asset owner(s) is no longer identified by the vulnerability detection and active monitoring tools of the person responsible for the vulnerability management process.
Output	Remedied and Validated Vulnerability

3.1.6 Retrospective

The retrospective, represented in Figure 3.6, is the sixth and final stage of the vulnerability management process. It intends to guarantee information about the current state of risk around the organization's vulnerabilities, being highly relevant to all elements of the IT team, executives, or managers. Table 3.23 specifies the inputs and outputs typically expected in this stage.

The creation of tactical reports on identified and remedied vulnerabilities, the organization's progress in its vulnerability management, and the company's global risk provide executives with a summary of the current state of vulnerabilities.

Table 3.23: Input and output of the Retrospective stage

Input	Remedied and validated vulnerability(s)
Output	Documented Vulnerability(s). Procedural improvements. Final metrics report. Self-assessment of the vulnerability management cycle for possible improvements. The vulnerability's documentation, from its discovery to its mitigation, for further self-reflective analysis.

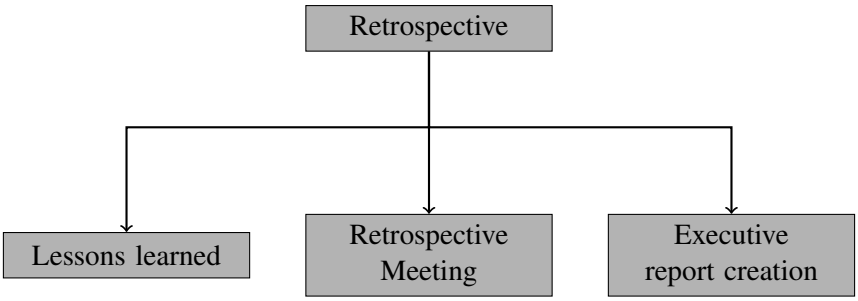


Figure 3.6: Representation of the Retrospective stage

The team will have to document the entire procedure, from the beginning to the end (described in Table 3.24). This aims to create a stage where it is in the organization's interest to carry out a self-reflective analysis of the challenges faced, as well as a self-assessment of the vulnerability management cycle, with a meeting as described in Table 3.25.

Table 3.24: Lessons learned

Sub-activity	Lessons learned
Scheduling	Punctual activity
Input	Remedied and Validated Vulnerability
Description	The entire process, from the identification of vulnerabilities to their mitigation, must be documented to create a knowledge base for future situations and ensure that the organization can respond to all incidents that it encounters, in terms of cybersecurity, in a faster and more agile way.
Output	Documented Vulnerability

Table 3.25: Retrospective Meeting

Sub-activity	Retrospective Meeting
Scheduling	Monthly activity
Description	The retrospective meeting is intended to be as informal as possible, as it only aims to incorporate the continuous improvement of this vulnerability management process. The meeting must be conducted by the entity responsible for this vulnerability management process, together with all the stakeholders. All stakeholders must answer the questions: "What should we have done/do more?", "What should we have done/do less?", "Should we start doing X?" or "Should we end up with Y?". Other questions with reflexive and self-assessment power for the vulnerability management process are accepted.
Output	Procedural improvements

Finally, it will be necessary to produce a report with a minimum monthly frequency, where all the information related to metrics will be exposed, to deliver it to the team of top or intermediate managers (C-Level). This sub-activity is described in Table 3.26.

Table 3.27 summarizes the contribution of the various vulnerability management processes presented in Chapter 2 to the proposed process.

Table 3.26: Executive report creation

Sub-activity	Executive report creation
Scheduling	Monthly activity
Input	Information obtained in the collection of metrics of the vulnerability management process
Description	In order to complete the vulnerability management process, a final report should be generated with all the vulnerabilities detected and corrected, being only essential to portray the quantities of these, which were addressed, instead of explaining, in detail, which they are. For a better understanding, vulnerabilities can be classified according to a standard classification. This report should indicate as many metrics as possible to understand the performance and ability to resolve the vulnerabilities the organization is targeting and the risk metrics of vulnerabilities that could not be corrected according to the risk management process in place.
Output	Final metrics report

Table 3.27: Contributions of each revised process to SecRush

Stage	Exabeam	Ascend Technologies	Fordham University	AT&T
Discovery	N/A	As in Ascend Technologies, the Discovery stage is carried out continuously, where all devices connected to the network are identified, with no need for manual inventory.	N/A	As in AT&T, the Discovery stage is continuous, regular, and automatic, allowing the rapid updating of information.
Prioritization	The Exabeam process takes into account other metrics, in addition to CVSS. As the Exabeam, in this work, the prioritization stage does not rely on the CVSS. It follows a new scoring method herein proposed (the SecScore).	N/A	N/A	N/A
Report	N/A	As in the Ascend Technologies model, this work creates an intermediate step to document and account for vulnerabilities to be addressed.	N/A	N/A
Remediation	N/A	N/A	As in the Fordham University model, this work sets a minimum deadline to address vulnerabilities.	N/A
Verification	N/A	N/A	N/A	As the AT&T model, this work include tasks to verify that the controls applied to mitigate the vulnerability(s) are correctly applied.
Retrospective	Like the Exabeam model, this work includes the concept of "Lessons Learned" in case the team reencounters the same problem.	N/A	N/A	N/A

3.2 SecScore - Scoring Method

SecScore is based on CVSS. Nevertheless, it proposes improvements to circumvent some CVSS limitations [13, 36, 37, 38, 39, 45], especially in calculating its temporal and environmental components. The CVSS score is given in three components: base score, temporal score, and environmental score. They all use intermediate values and coefficients for their calculation, which are described in [23].

The CVSS's base score usually varies according to the Scope (S), a metric that captures whether a vulnerability in a vulnerable component affects resources in other components beyond its security context. Its formula can go three ways, depending on the given base metrics, according to Equation 3.1. If the resulting value from the Impact score is less than or equal to zero, then the base score is also zero. If the Scope is Unchanged, the scoring follows the second formula, and if the Scope is Changed, the scoring follows the third formula. The Impact sub-score follows the same principle as the Base equation, as it can go two separate ways, depending on the S value, as stated in Equation 3.2. The Exploitability score only has one equation that uses the remaining base metrics: Attack Vector (AV), Attack Complexity (AC), Privileges Required (PR), and User Interaction (UI), as displayed in Equation 3.3.

$$Base = \begin{cases} 0 & \text{if } Impact <= 0 \\ Round\left(Min\left[(Impact + Exploitability), 10\right]\right) & \text{if } S = \text{Unchanged} \\ Round\left(Min\left[1.08 \times (Impact + Exploitability), 10\right]\right) & \text{if } S = \text{Changed} \end{cases} \tag{3.1}$$

$$Impact = \begin{cases} 6.42 \times ISS & \text{if } S = \text{Unchanged} \\ 7.52 \times (ISS - 0.029) - 3.25 \times (ISS - 0.02)^{15} & \text{if } S = \text{Changed} \end{cases} \tag{3.2}$$

$$Exploitability = 8.22 \times AV \times AC \times PR \times UI \tag{3.3}$$

Note that the Impact Sub-Score (ISS) must be calculated first to calculate the Impact score. According to Equation 3.4, this sub-score will take all impact metrics: Confidentiality, Integrity, and Availability.

$$ISS = 1 - [(1 - Confidentiality) \times (1 - Integrity) \times (1 - Availability)] \tag{3.4}$$

For the sake of completeness, the base metrics equations are presented here, but the base component of SecScore does not get changed. As stated above, there will only be changes to the temporal and environmental components of CVSS.

The calculation of the temporal score is currently computed in the original CVSS as shown in the equation 3.5:

$$Temporal = Round(Base \times E \times RL \times RC), \tag{3.5}$$

wherein E is Exploit Code Maturity, RL is Remediation Level and RC is Report Confidence.

Most vulnerabilities do not have an exploit code, in line with [37]. However, for those that have it, its invariability over time, especially for the parameters contained in the temporal metrics (Exploit Code Maturity, Remediation Level, and Report Confidence), does not represent this change when it should be significant. There are three determining factors: the identified vulnerability, an indication of the existence

of code to exploit it, and no possession of the code for its exploitation. There may be cybercriminals in possession of that information with the ability to exploit it. The risk of this vulnerability should increase.

Currently, the CVSS classification equation concerning temporal metrics does not refer to any factor that varies over time but to characteristics that may change. Even if an exploit for the vulnerability is not yet public, it is reasonable to assume that an attacker is already developing it. Therefore, it is necessary to consider the possibility that the risk increases as time passes. However, the CVSS never reflects a real temporal factor, that is, the variation of time in the formula with the hypothetical increase in vulnerability risk. Since it is a metric based on time and the evolution of the vulnerability's state over time, it must be updated permanently, particularly in cases where vulnerabilities have an active exploit code with a High rating.

Based on this information, the original CVSS temporal equation (3.5) was modified in order to take into account the time elapsed since the vulnerability was discovered. Therefore, the temporal metrics formula for vulnerabilities whose exploit code is unknown or has a High classification is rewritten as in the equation 3.6.

$$Min\left(Round\left(Base\ \times\left(1+\frac{W}{R/7}\right)\ \times E\times RL\times RC\right),\ 10\right) \tag{3.6}$$

The change is the inclusion of the $\left(1+\frac{W}{R/7}\right)$ component that promotes the variability of the temporal classification over the time of the vulnerability's existence. This value considers the time it takes for a vulnerability whose exploit code maturity has the highest rating or is undefined to see an official publication. It will continually increase until it sees its mitigation applied. The value W defines the number of weeks elapsed since the CVE was published (first official vulnerability date). It would be compelling and closer to reality to define this parameter with a zero-day date. However, this value is difficult to access and is conditioned by the acquisition of vulnerability intelligence services. The variable R expresses the estimated number of days the vulnerability will see its communication by the software vendor and can be adjusted according to the risk appetite of the organization that intends to prioritize the vulnerabilities. The number 7 aims to normalize the final value in weeks, as this is the time resolution of this equation, and thus, it remains the same value over a week.

This R value must be tabulated and chosen according to the risk profile of the organization that applies it. Like GDP (Gross Domestic Product) or CPI (Consumer Price Index), possible values for the R value should be published by a central entity (e.g., NIST, which produces the NVD, or MITRE, which assigns the CVE numbers). These values, such as GDP or CPI, would be reviewed annually or quarterly based on current vulnerabilities relative to the period under review. In other words, the analysis produced here in this work could be performed every year by these central entities to ensure harmonization in calculating this new value, which would eventually become an index to be applied by organizations in calculating risk. It is essential to make this value variable, as the risk that vulnerabilities pose tends to change over time.

One of the ways to statistically estimate R is through a sample's quartiles. These statistics can provide R values considering the first, second, and third quartiles (25%, 50%, and 75% of the vulnerabilities). As in a risk matrix, there are only three levels of risk that organizations can accept, according to their risk appetite. So the R values obtained from these quartiles would correspond to these levels of risk. Further

on, in the Section 4.1, in Figure 4.2, it is possible to verify the results obtained for an analysis period of twelve months.

Environmental metrics allow customization of the CVSS score depending on the importance of the affected IT asset to a user's organization, measured in terms of implemented complementary or alternative security controls, confidentiality, integrity, and availability.

In the original CVSS, the calculation of environmental metrics is performed with the equation 3.7. If the Modified Scope (MS) is unchanged, it follows the first equation; otherwise, the second one is used. The "Round" denominates the mathematical rounding operation. Modified Impact (MI) and Modified Exploitability (ME) come from the original CVSS formulation, and their computation are described in Equations 3.8 and 3.9, respectively:

$$Environmental = \begin{cases} Round(Arr\left[Min\left([MI + ME]\,,\ 10\right)\right] \times \\ \qquad\qquad E\ \times RL\ \times RC) & \text{if } MS = \text{Unchanged} \\ Round(Arr\left[Min\left(1,08\ \times [MI + ME]\,,\ 10\right)\right] \times \\ \qquad\qquad E\ \times RL\ \times RC) & \text{if } MS = \text{Changed} \end{cases} \tag{3.7}$$

$$Modified\ Impact = \begin{cases} 6.42 \times MISS & \text{if } MS = \text{Unchanged} \\ 7.52 \times (MISS - 0.029) - 3.25 \times \\ \qquad (MISS \times 0.9731 - 0.02)^{13} & \text{if } MS = \text{Changed} \end{cases} \tag{3.8}$$

$$Modified\ Exploitability = 8.22 \times MAV \times MAC \times MPR \times MUI, \tag{3.9}$$

wherein MAV is Modified Attack Vector, MAC is Modified Attack Complexity, MPR is Modified Privileges Required, and MUI is Modified User Interaction.

Finally, Modified Impact Sub-Score (MISS) is calculated according to Equation 3.10. It combines the Confidentiality, Integrity, and Availability Requirements (CR, IR, AR) with the same modified properties - Modified Confidentiality, Integrity, and Availability (MC, MInt, MA).

$$MISS = Min\left(1 - \left[(1 - CR \times MC) \times (1 - IR \times MInt) \times (1 - AR \times MA)\right], 0.915\right) \tag{3.10}$$

Note that, regarding environmental metrics, the CVSS already has some information to assess this vulnerability in a given environmental context. One example of this information is the criticality of the security properties of each asset where the vulnerability exists. However, this is not enough, and it is necessary to prioritize its visibility to the outside (*i.e.*, distinguish machines that have services available to the public, such as a server, from a personal workstation).

Regarding the proposed environmental metrics formula, there are two new parameters. The parameter V considers the visibility of the vulnerable asset, and I represents the asset's importance to the organization. Both parameters vary in the continuous range between $]0, 1]$. The visibility of the asset is a value that intends to characterize the asset's exposure to the outside environment (not controlled by the organization), and it will be higher the greater its exposure. On the other hand, the importance to the organization characterizes the importance of the asset to the business. That is, the value will be higher

for the greater importance of keeping the asset in operation without disruption so as not to impact the organization.

A maximum V value implies that this asset will have great visibility to the outside and can be accessed by any user. However, a V value tending to 0 already indicates that it can only be accessed by restricted users in the company's network. A value of I equal to 1 implies that the importance of this asset to the company's business is maximum. It cannot have disruptions that impact the normal functioning of the organization. In contrast, the same parameter, with a value close to 0, indicates lower importance and that it will not affect any particular area of the organization for a long time or that there are alternative solutions already foreseen by the organization to solve the problem.

In SecScore, all vulnerabilities will be classified, in their environmental aspect, with the equation 3.11. Vulnerabilities whose Modified Scope (MS) remains unchanged will use the first equation. Otherwise, they will use the second one. In both equations, it is crucial to keep the expression $V \times\ (1 + I)$, as these parameters can be any value from the continuous range $]0, 1]$. Since it can take decimal values, the calculated value would necessarily be smaller, so the unit in parentheses is vital to balance the result to be determined:

$$Environmental = \begin{cases} Min(V \times (1 + I) \times Round\left([Min\left(\right.\right. & \text{if } MS = \text{Unchanged} \\ \left.\left.[MI + ME], 10 \times E \times RL \times RC, 10\right); \right. & \\ Min(V \times\ (1 + I) \times Round\left([Min\left(1, 08\right.\right. & \text{if } MS = \text{Changed} \\ \left.\left. \times [MI + ME],\ 10 \times E\ \times RL\ \times RC,\ 10\right); \right. & \end{cases} \tag{3.11}$$

where $V, I \in]0,\ 1]$.

In the formula, visibility multiplies by importance, leading to the belief that one overlaps the other. However, the interpretation is that visibility weighs more in the severity of the vulnerability classification since external exposure is more dangerous and likely to lead to an attack.

At first, adding more parameters for calculating a security score could make it counterproductive, as it would be another value to consider when assessing vulnerabilities. However, the impact of this addition is minimal since these values are factual and observable, or else, in the case of R, they may be provided in advance by external organizations or statistically estimated by the organization. For example, the W in equation 3.6 is the number of weeks since the publication of the CVE code, which is observable by recording the publication date. R is always a stable value unless there is a need to diversify the risk appetite for different assets. Finally, the organization usually defines the value I through an initial risk analysis, where the asset's criticality for its mission is determined.

3.3 Vulnerability Management System

The vulnerability management system is implemented in Python with Django, aiming at building the most likely usage scenarios to test the process and the proposed scoring equation for prioritizing vulnerabilities. This system has the following requirements:

- Congregate all vulnerabilities from all detection sources on the same system;

- Collect and organize vulnerability information according to a standardized structure for the organization;
- Manage the vulnerabilities of the organization's information systems with asset owners directly through the platform;
- Be able to monitor the vulnerabilities treatment status, dividing them into untreated vulnerabilities, vulnerabilities in treatment, and mitigated vulnerabilities;
- Identify the machines affected by a particular vulnerability;
- Be aware of all the base, temporal (and environmental) variables on the vulnerability description page, as well as the respective SecScore classification;

The project has two data structures: Assets and Vulnerabilities. The asset table describes the composition and identification of each asset and is composed of the following parameters:

- DNS Name;
- User;
- OS;
- Version;
- Build;
- Last IP;
- Last External IP;
- RBAC Group ID;
- RBAC Group Name;
- Asset Owner;
- Last Updated.

The table of vulnerabilities identifies all the parameters related to the vulnerability classification, allowing its analysis and prioritization. This table is composed of the following parameters:

- CVE;
- Description;
- Attack Vector;
- Attack Complexity;
- Privileges Required;

- User Interaction;
- Scope;
- Confidentiality;
- Integrity;
- Availability;
- Exploit Code Maturity;
- Remediation Level;
- Report Confidence;
- URL;
- State;
- Public Exploit;
- Exploit Verified;
- Exploit URL;
- Reported On;
- Asset;
- Last Update.

The interface of the developed Vulnerability Management System is composed by four main pages:

- **Home page:** description of the organization's current status in vulnerability management, showing the organization's total number of vulnerabilities and its composition of monitored assets;
- **Vulnerabilities:** shows vulnerabilities that must be addressed in the organization's systems. By default, the vulnerabilities are prioritized by SecScore;
- **In Progress:** shows vulnerabilities that transitioned from "Untreated" to "Under Treatment" status;
- **Archive:** shows vulnerabilities that have seen their mitigation measures implemented.

Chapter 4

Results

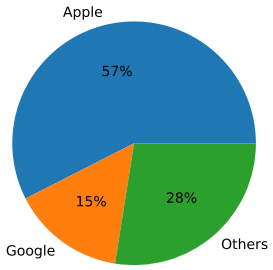
This chapter details the results of the experiments carried out to validate the proposal presented in Chapter 3. It is divided into three subsections: in Section 4.1, the results of the tests carried out to validate the SecScore proposal are presented; followed by Section 4.2, where the results of the tests of the SecRush process are presented and, finally, in Section 4.3, a brief presentation of the vulnerability management system created specifically for this work.

4.1 SecScore

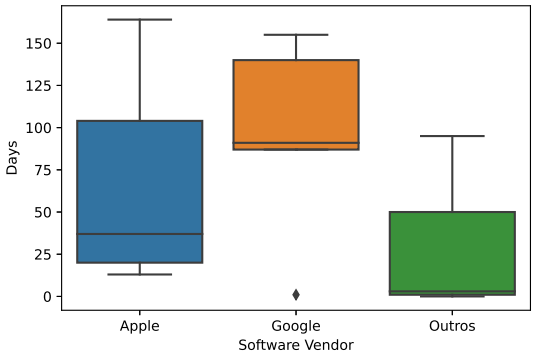
The analysis of the SecScore classification method was carried out using a dataset obtained in VulDB [16]. There were selected vulnerabilities classified as High in the temporal parameter "Exploit Code Maturity", whose notice of the vulnerable software producer was published between April 2021 and March 2022. The dataset used is available in Appendix B. In this period, 40 vulnerabilities with this Exploit Code Maturity classification were published in the database used as a reference in this work (VulDB).

As shown in Figure 4.1a, among the vulnerabilities analyzed, most were Apple vulnerabilities, followed by Google. All the others represent between 3% to 5% individually, *i.e.* about 1 to 2 vulnerabilities from each vendor in the analyzed period.

In this section, the number of days always represents the time elapsed between the publication of the



(a) Vulnerabilities distribution



(b) Boxplot between the publication of the CVE and the confirmation of the vulnerability

Figure 4.1: Data grouped by software vendor.

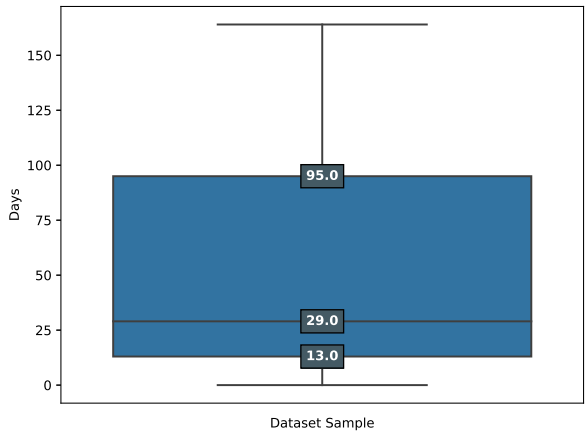


Figure 4.2: Boxplot between the publication of the CVE and the confirmation of the vulnerability by the software vendor.

CVE and the confirmation of the vulnerability. Figure 4.1b shows the mean number of days between the publication of the CVE and the confirmation of the vulnerability for each vendor[1]. Considering all vulnerabilities obtained from the database according to the aforementioned criteria, the mean, mode, and standard deviation are 55 days, 20 days, and 55 days. Note that it is not trivial to extract much information from these measures of central tendency since the mean is equal to the standard deviation. This is primarily caused by the dispersion of data and the disparity in the treatment of vulnerabilities by the different software producers. Therefore, it is impossible to define an R directly from, for example, the average number of days of the entire sample.

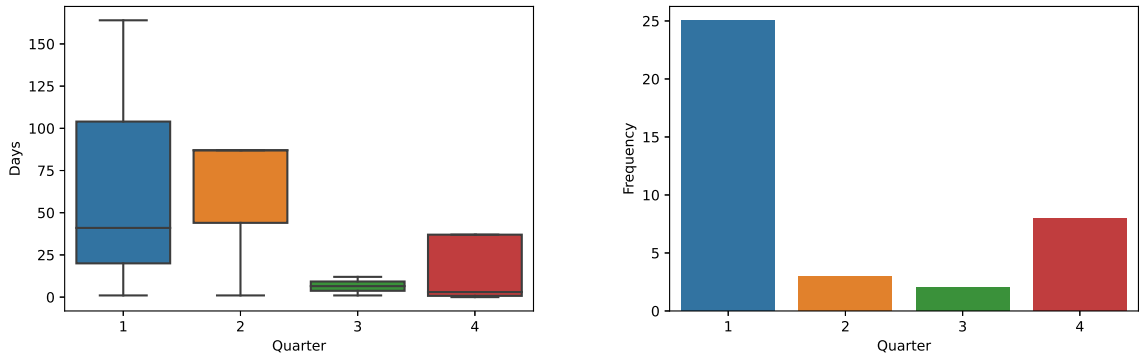
From the exact information of the number of days between the publication of the CVE and the confirmation of the vulnerability, a diagram of extremes and quartiles was produced for the entire sample, as shown in Figure 4.2. In this figure, it is already possible to extract some information, namely that 25% of vulnerabilities are discovered and recognized by software-producing companies within 13 days, and 50% of these same vulnerabilities suffer the same action within 29 days. From the median to the highest value is a large difference in the number of days. This analysis makes it possible to conclude that half of the vulnerabilities in the analyzed sample should be mitigated in less than a month, with 25% of them in less than two weeks. Figure 4.1b shows the same information of Figure 4.2 segmented by each software producer.

To better understand the dataset, it was also analyzed by quarters, as shown in Figure 4.3a. The first quarter is the period from April to June 2021, with values of 21, 41, and 104 days for the first, second, and third quartiles. In the second quarter, the values were 44, 87, and 87 days for the first, second, and third quartiles; in the third quarter, the values were 3.8, 6.5, and 9.2 days for the first, second, and third quartiles; and finally, in the fourth quarter, the values of 0.8, 3, and 37 days for the first, second, and third quartiles, respectively.

The vulnerabilities' frequency in each of the quarters was also analyzed in Figure 4.3b. The first quarter of the sample contains the most vulnerabilities - 23, followed by the last quarter of the analysis with 8. The second has only three, and the third has two quantities of little relevance for the individual analysis.

Based on the analyzed data, the R value could be 13, 29, or 95 days (corresponding to the first,

[1]Each vendor issues a warning, often accompanied by a form of mitigation.



(a) Boxplot

(b) Vulnerability frequency

Figure 4.3: Sample data grouped by quarter.

second, and third quartiles), according to the company's risk appetite. However, the temporal analysis done here by quarter, over a year, shows that the statistics on vulnerabilities can change over time. It is necessary to follow up and update R when necessary.

4.1.1 Analysis on the temporal modification

Tests were performed on a typical vulnerability prioritization scenario from April 12 to April 19, 2021, to simulate a typical week of mitigation application and validate the proposed modification.

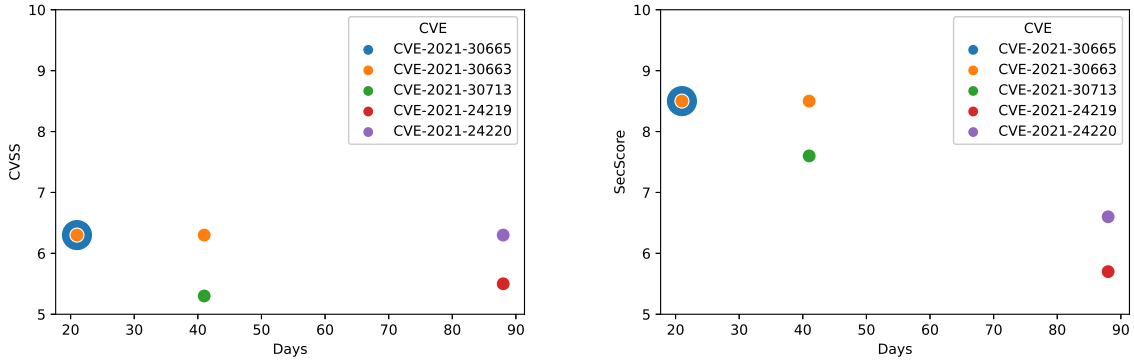
For this purpose, six vulnerabilities were chosen for the assessment. The condition was that the CVE of the same had already been published. However, the disclosure of the security mitigation by the company producing the software was still to be announced. Table 4.1 shows the vulnerabilities, with the number of days it took for the responsible company to issue a security warning, the base CVSS classification, the SecScore classification, and its variation compared to CVSS. It should be noted that each row of this table denotes a vulnerability with its corresponding security mitigation disclosure by the software company. Hence why there is a repetition for "CVE-2021-30663" as, in this case, the advisory from Apple – the software manufacturer with this CVE – was released at different dates for iOS, macOS and Safari (21 days) and tvOS (41 days).

The information in the Table 4.1 was organized graphically by CVSS in the Figure 4.4a and by SecScore in the Figure 4.4b. In both figures, the x-axis designates the days that the software manufacturer took to disclose the security mitigation and the y-axis designates the classification of CVSS and SecScore, respectively. In the CVSS graph, it is possible to observe that the vulnerabilities are distributed throughout the rectangle of the figure, with no great relationship between them. When it is necessary to prioritize vulnerabilities, it becomes difficult to decide. In the SecScore graph, most vulnerabilities tend to decrease along the x-axis (days). It is important to note that this information is calculated for a particular date (April 12) and that these vulnerabilities today would not have the same computed value. Graphically, the interest is that the vulnerabilities form a line as similar as possible to a line with a negative slope.

In Figure 4.5, where the horizontal axis designates the week number of 2021, it is clear to see how the new prioritization formula affects vulnerabilities over time, allowing an evaluation on how the SecScore evolves since the vulnerability has its CVE published. With the original CVSS computation, we would

Table 4.1: Vulnerabilities chosen for testing

CVE	Days until disclosure	CVSS	SecScore	Variation
CVE-2021-30665	21	6,3	8,5	-34,9%
CVE-2021-30663	21	6,3	8,5	-34,9%
CVE-2021-30663	41	6,3	8,5	-34,9%
CVE-2021-30713	41	5,3	7,6	-43,4%
CVE-2021-24219	88	5,5	5,7	-3,5%
CVE-2021-24220	88	6,3	6,6	-4,6%



(a) CVSS priorization

(b) SecScore priorization

Figure 4.4: One-week demo of vulnerability prioritization.

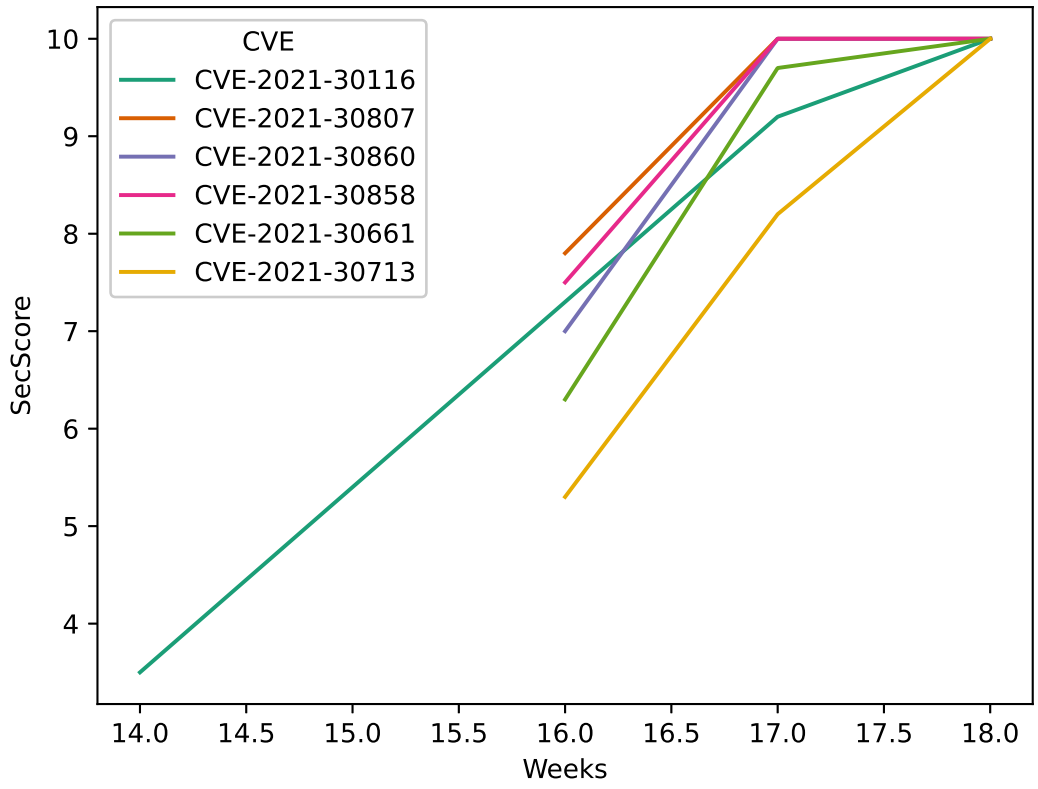


Figure 4.5: Temporal variation of a subset of vulnerabilities from 2021 with SecScore.

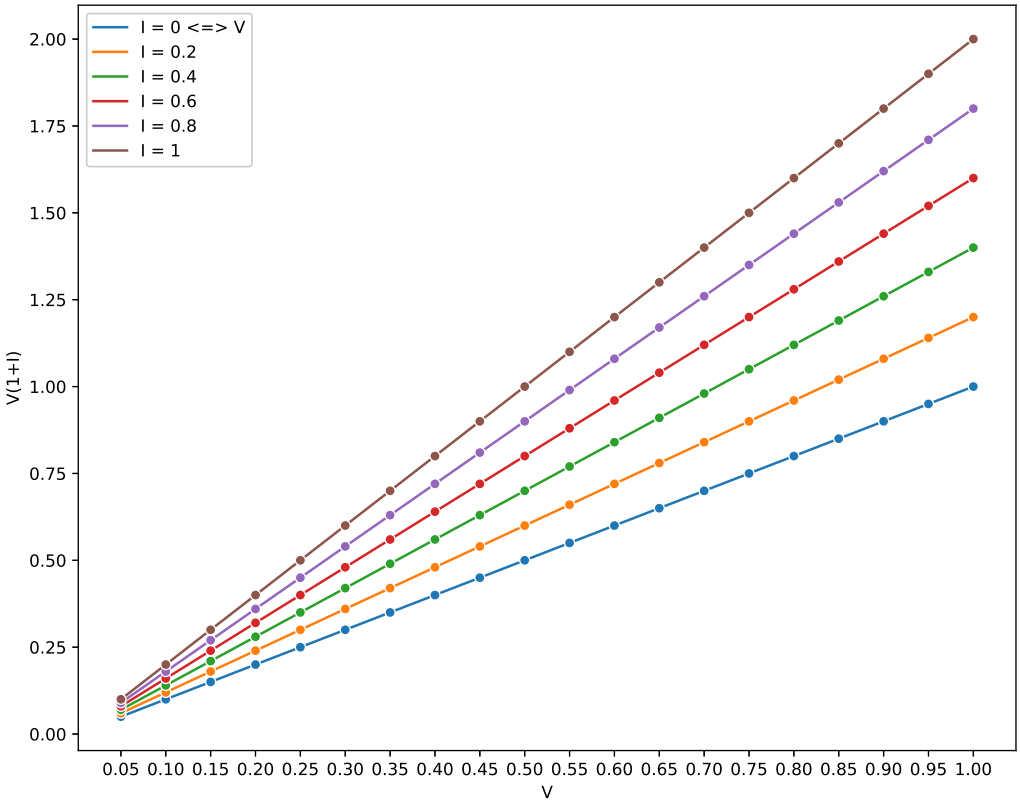


Figure 4.6: Analysis of $V(1 + I)$ for 20 levels of V and 5 levels of I.

only obtain the initial value (which in the graph is represented by the initial point). With SecScore, in turn, we get a value that increases towards ten at different speeds. Take CVE-2021-30116 as an example. This vulnerability was the first to appear; however, CVE-2021-30661 or CVE-2021-30860 started with lower values than the first mentioned vulnerability but quickly surpassed it as time passed. In a strategy in which an organization is faced with vulnerability mitigation organized in weeks, it would likely try to mitigate the first vulnerability to appear (in the 14th week of 2021). However, this graph shows that this is not always the correct form of prioritization, requiring this variation in the much-needed time.

4.1.2 Discussion on the environmental modification

As previously presented in Section 3.2, V represents the visibility of the vulnerable asset and I represents the asset's importance to the organization. In the example shown in Figure 4.6, there are twenty levels of V and five levels of I defined. The value of $V(1+I)$ is plotted for each level of I. It is possible to verify that the value of $V(1 + I)$ gets bigger the greater the value of V. However, as expected, as the value of I increases, the greater the parameter to be added in the final calculation of the environmental metrics. It should be noted that with this parameter, the value can depreciate when $V(1+I)$ is less than 1 and inflate if the opposite occurs, being possible to reach the minimum value close to 0 and a maximum value of 2.

In line with what was said in [37, 38, 39, 45], an issue that could be asked when presenting this

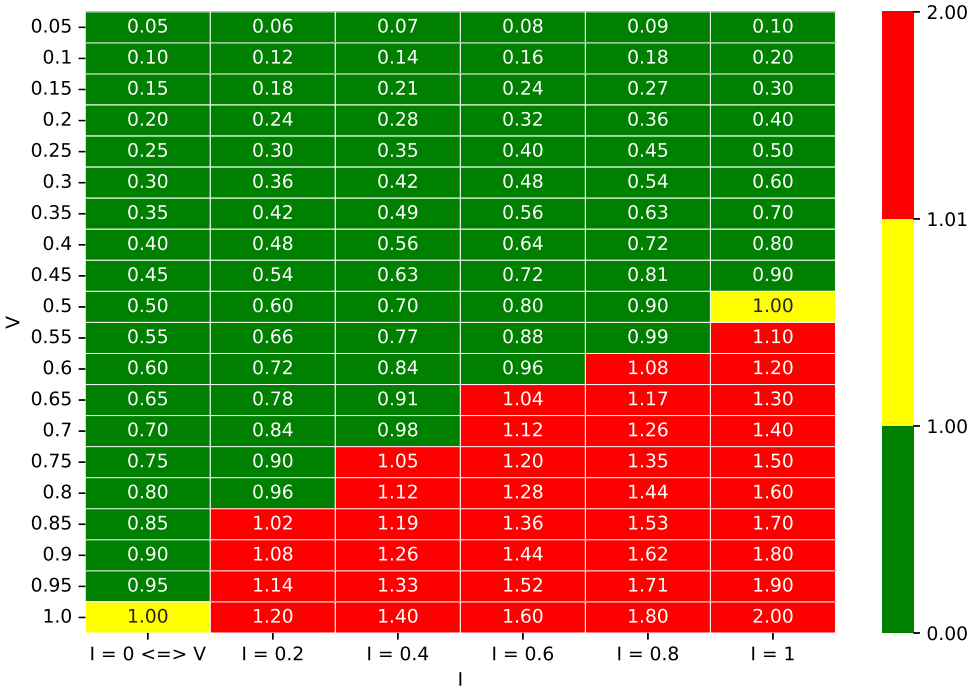


Figure 4.7: Heatmap for the results of $V(1 + I)$ for 20 levels of V and 5 levels of I.

parameter would be the tendency to obtain an inflated value compared to the CVSS classification since the proposed formula presents this possibility. However, this is not always the case. With this addition to the environmental metrics formula, there is an attenuation of the values previously computed by the temporal metrics, as seen on Figure 4.7. This attenuation depends on the need and criticality of the vulnerability and the asset where the vulnerability is present.

4.2 Process

The process described in Section 4.2 was tested in its entirety to collect information on its suitability in the organization, the identified vulnerabilities, its extent and described procedures, and the performance that this agile methodology promotes through its potential to increase.

In the tests developed, the organization's cybersecurity and IT Administration teams were included since the sources of the chosen vulnerabilities were monitored. They identified the vulnerabilities of the employees' workstations. The IT Administration Team is in charge of the management and maintenance of these assets.

Bearing in mind that the process developed had six phases, all of them were listed in the sequence shown in Figure 3.1.

The Discovery was carried out in an automated and continuous way, as expected, by the internal tool. The vulnerability detection systems were already active, and they collected the information in an automated way for the assets targeted in these tests. Hence, no additional step was necessary to what was already being conducted internally.

Table 4.2: Identified vulnerabilities that posed the greatest risk

Software	Identified CVE's	Vulnerable assets	Total of vulnerabilities
Cisco Jabber	6	1907	11442
Cisco Webex Meetings	12	1751	21012
Adobe Reader DC	203	32	6496
PuTTY	11	19	209
Flash	53	45	2385
Total			41544

Then, in Prioritization, the vulnerability management system, created for the purpose, aggregated all the information collected by the vulnerability detection tool and prioritized it according to the SecScore classification. In this phase, the information collected by the detection tool and the information in the NVD databases were added to the information collected in VulDB, particularly the temporal metrics of each vulnerability. For the parameter R, essential for the calculation of SecRush, the value of 13 days was named. Until this stage, the entire process has remained automated, as foreseen in Section 4.2.

The third phase, the Report, was the first that saw human intervention. According to Table 4.2, the cybersecurity team prepared a brief report with the most critical vulnerabilities, synthesizing and aggregating this information in five different software, each one having their own CVE's and identified vulnerable assets, capping in a total of 41544 identified vulnerabilities. The software mentioned were the ones that, to date, had the greatest potential for an intrusion threat, not only because they had a higher SecScore rating but because they had one or more active forms of exploitation. In order to quickly resolve the vulnerabilities detected, a two-week rush was prepared for the IT Administration team, as these are the owners of the assets targeted in this vulnerability detection. As foreseen in the process, this rush was accompanied by an action report identifying these vulnerabilities and the recommended mitigation method.

Afterward, the IT Administration team proceeded with Remediation. Of the four types of remediation foreseen in the process, all vulnerabilities fall under the same category – Normal Vulnerability Remediation with a patch for the current rush. All had an official way of remediation, updating to a newer version and not being critical, given the criteria that define the need for a critical vulnerability or an emergency rush.

In the fifth phase, Verification, the team carrying out the mitigation of vulnerable assets communicated the conclusion of the mitigations to the cybersecurity team, confirming it. Confirmation was performed at the end of the rush, two weeks after it started, through the normal vulnerability scan provided for in the process. This scan was done daily, and it was possible to see a constant decrease in the number of vulnerabilities previously detected in the organization's assets.

Finally, in the Retrospective, the group held a retrospective meeting. The meeting got some suggestions on improving the process and adapting it to the workflow of both teams. This adjustment was needed so each team is synchronized with the process and to improve the mitigation's response effectiveness. No content was built to save in "Lessons Learned" because there was no anomalous situation that the team had encountered when solving this rush. The creation of the executive report was also not carried out, not only because there was only a rush and there was no previous data with the same units to compare, but there was also no executive need to carry it out.

Iterating the process by creating this rush proved successful as it managed, within two weeks, to resolve 41544 vulnerabilities on the organization's workstations. Including agile project management concepts in this process is essential to achieving these results. These results happen for several reasons: 1/3 of the process was automated to perform these tests (Discovery and Prioritization). However, half of the process can be automated since the construction of action reports in rush preparation is built with content previously collected from other sources, as mentioned above. The focus shifts to the rapid mitigation of vulnerabilities, as opposed to creating documentation; that is, the cybersecurity work takes a dominant position so that it can be executed in the shortest possible time and thus cover more vulnerabilities to be mitigated in the same amount of time. From the beginning of the process, the roles were well defined and grouped by the different areas/departments of the organization, that is, there is a clear responsibility and distribution of tasks for what a rush will contain; the collaboration between the cybersecurity team and the other teams responsible for vulnerable assets facilitates the process in order to make it faster, as the cybersecurity team is aware of the vulnerabilities and risks that this implies in IT terms and the impact that these will bring to the business, while the other teams responsible for the assets are more knowledgeable about the features they have and can, with greater ease, carry out a mitigation action in the shortest possible time; this collaboration ends up increasing the dependence between the two teams, forcing them to work towards a common goal, which will bring about an increase in productivity; finally, as in an agile methodology, the process is scalable – what happened with just two teams could increase to be able to incorporate the various teams responsible for the organization's IT assets and thus lead vulnerability management on a much larger scale.

4.3 Vulnerability Management System

In order to carry out the process in Section 4.2, with the calculation formula in Section 3.2, a vulnerability management system was created, as mentioned in Section 3.3. The solution was carried out through the creation of a Web interface that presents a structure with four main pages:

- **Home page:** description of the organization's current status in vulnerability management, showing the organization's total number of vulnerabilities and its composition of monitored assets;

Figure 4.8: Main dashboard

- **Vulnerabilities:** all vulnerabilities present in the organization's systems that must be addressed. Vulnerabilities are prioritized by default by SecScore. This page presents a table that shows some

of the data indicating the vulnerabilities, such as the CVE, the description, and the SecScore;

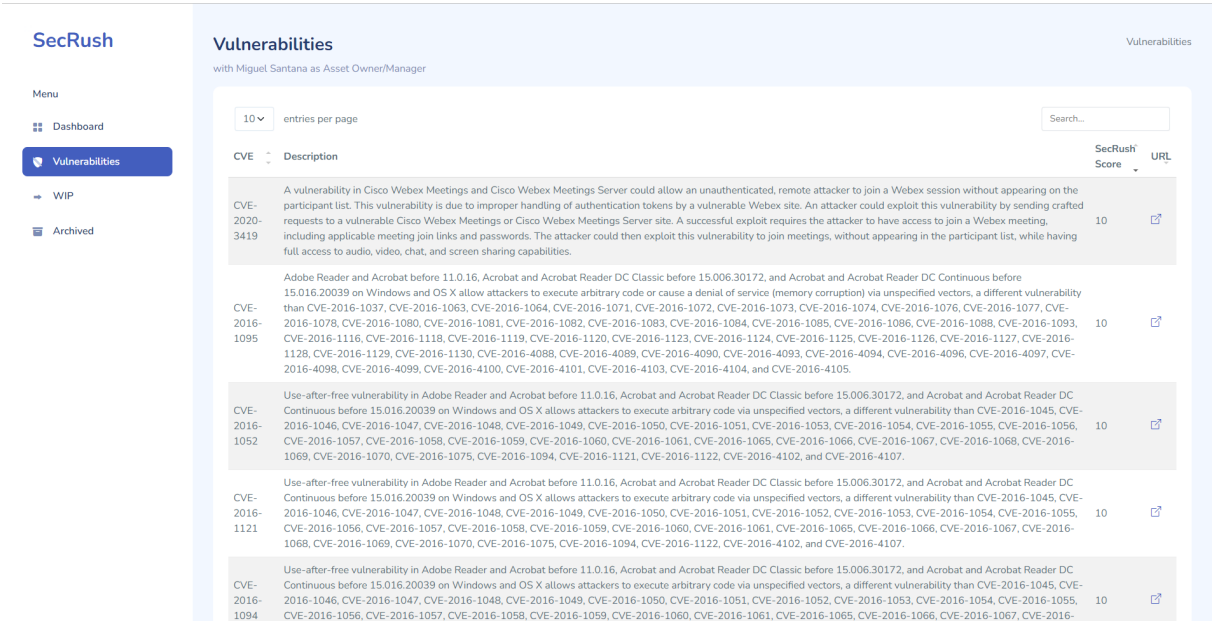


Figure 4.9: Active vulnerabilities page

- **In Progress:** All vulnerabilities that have transitioned from “Untreated” to “Under Treatment” status. As on the “Vulnerabilities” page, the same data are displayed;

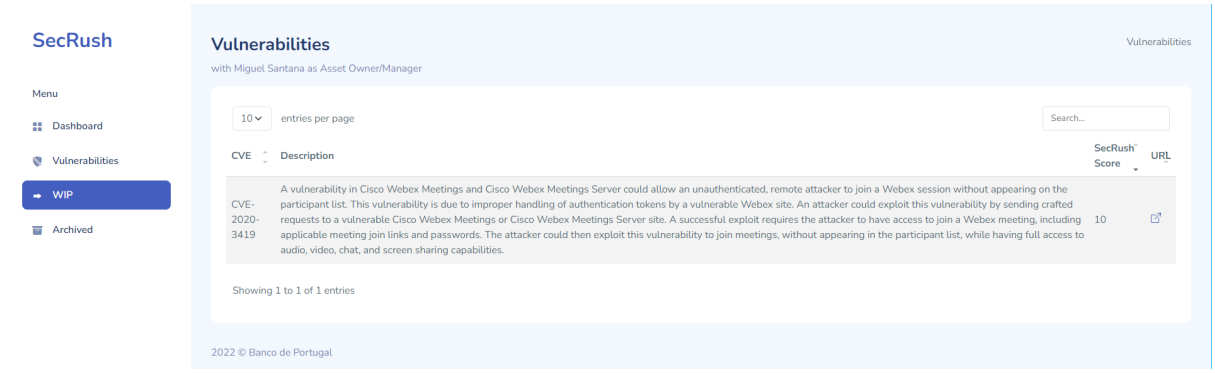


Figure 4.10: In Progress Vulnerabilities page

- **Archive:** all vulnerabilities that have already seen their mitigation measures implemented. Similar to the “Vulnerabilities” and “In Progress” pages, the same data characteristic of vulnerabilities are also presented.

Figure 4.11: Archived vulnerabilities page

- **Vulnerability:** characterizes a particular vulnerability associated with a CVE and identified in a known asset. All the information presented is the same as the "CVE" page and with the same options for the user.

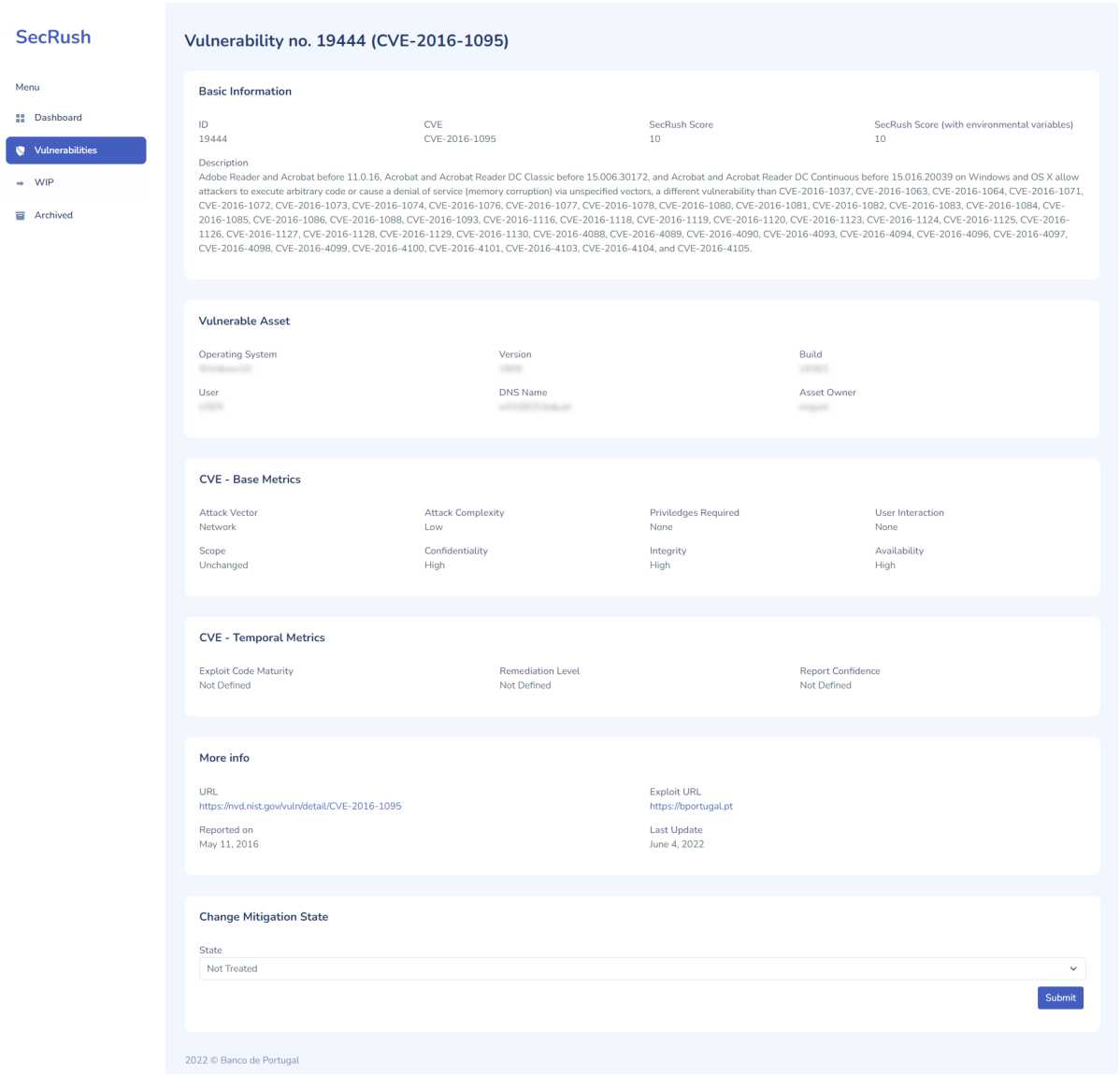


Figure 4.12: Vulnerability page

- **CVE:** characterization of a vulnerability publicly identified and classified by external entities. It presents information about vulnerability identification, description, SecScore ratings, vulnerable assets associated with this CVE, baseline metrics, temporal metrics, information about creating and updating information, and links to external content in more detail. At the end of the page, the user is given the option to change the vulnerability status;

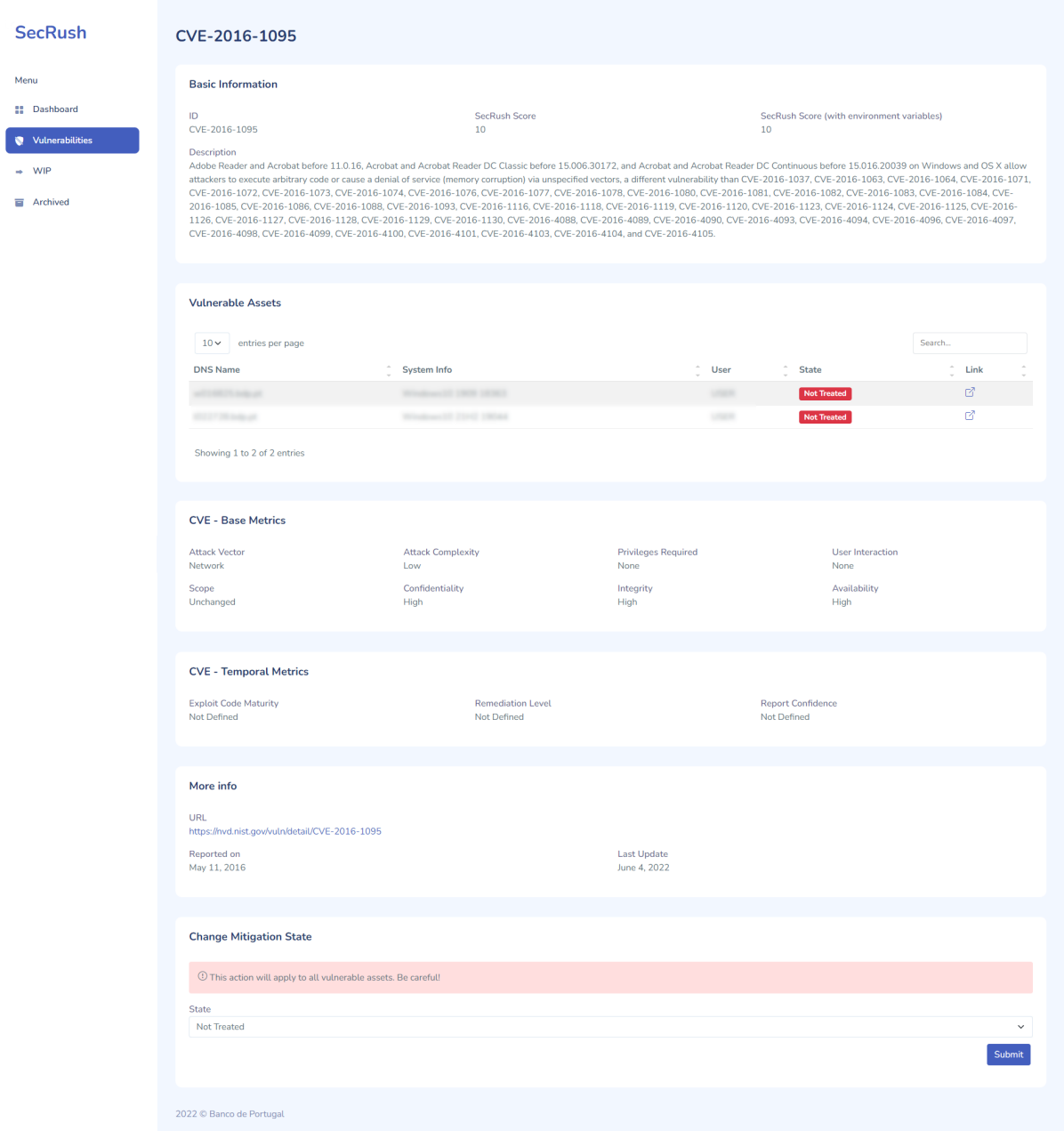


Figure 4.13: CVE page

Chapter 5

Conclusions

In this thesis, SecRush was presented, a new generation risk-based vulnerability management framework that aims to standardize vulnerability management through well-defined processes and procedures, independent of the tools used for its execution. This framework includes a new vulnerability management process and a new classification method that aims to improve the CVSS scoring system when prioritizing vulnerabilities. The process has six phases and monitors the vulnerability from when it is detected in the organization's IT infrastructure until the implemented mitigation is confirmed.

The thesis presents an alternative way to calculate the CVSS. It is understandable, across all the stakeholders involved, that there is a need to change the direction of how the CVSS is built, thanks to its criticisms. The classification method herein proposed offers a gain in prioritizing vulnerabilities since it addresses some of the CVSS constraints. Prioritization abandons the static analysis of CVSS, starting to guarantee the fluctuation of its characteristics over time. Especially with the prioritization of more severe forms of vulnerabilities, such as those with well-developed forms of exploitation. The assignment of variable parameters opens the possibility of calculating new coefficients taking into account what will be the natural evolution of the discovery and mitigation of vulnerabilities in the future. Thus, the new classification formulas change the temporal and environmental metrics, introducing new variables for the prioritization of vulnerabilities: the R, a risk factor defined statistically and in line with the company's risk appetite; the I, a coefficient that describes the importance of the asset for the organization; and V, the visibility of an asset to the organization's external network, that is, in other words, the level of exposure.

This framework brings some significant improvements over what already exists in terms of vulnerability management. In addition to the aforementioned changes, the framework is inspired by agile project management methodologies, namely SCRUM. This agile inspiration makes the central work – mitigation – much more privileged, and all the additional work for the vulnerability management process can be typified rather than eliminated, as in SCRUM. The framework is also expected to be as automated as possible, ensuring a continuous line of action for detecting and mitigating vulnerabilities. An example of this is the first three phases of the vulnerability management process proposed here: Discovery, Prioritization, and Report can (and should) be fully automated, ensuring much more efficient workflows and in line with the inherent agility of the process.

The results obtained from the experimental period of vulnerability management verified what was estimated in the construction of the model. The process conquered a large number of mitigated vul-

nerabilities, namely 41544 vulnerabilities mitigated in the time frame of two weeks. The vulnerability prioritization also abandoned the organization's old technique. The new and better prioritization measures the risk it promotes to the organization.

This work did not present known limitations during the project's development, either in its ideation, implementation, or results extraction. However, it would be highly fruitful to apply the test scenarios to more than one organization, preferably from different sectors of activity, to assess the adequacy of this plan and a new formula for managing and prioritizing vulnerabilities.

5.1 Future Work

Given the limited number of human resources to manage vulnerabilities in an organization, in addition to this plan, it would be helpful to create a function to calculate the time needed to apply to the mitigation of vulnerabilities. This function would make it possible to define and plan the efforts needed to mitigate particular vulnerabilities – such as those that appear in a rush – and to adapt better parallelization.

In addition to this, it would be essential to assess the vulnerabilities prioritized with SecScore, namely to assess the reliability in detecting the results obtained with this new calculation formula and respective mechanisms for reducing possible outliers.

Often, vulnerabilities in IT systems are associated with active forms of exploitation that allow the attacker to intrude through ransomware. In order to better prioritize vulnerabilities, organizations that want to define a monetary value associated with the hypothetical loss would be helpful to associate the financial risk of the attack, which could be evaluated based on the ransomware asking value (known or trend of previous values).

Finally, creating a machine learning model for the best prioritization of vulnerabilities and even an improvement to the current SecScore. Here, care must be taken not to make the same errors as [19] and [18]. It is imperative to make clear the procedure for obtaining data, training the model, and its reproducibility by organizations. It is the only way to make the model easy to reproduce by all organizations and thus create a viable alternative to CVSS.

Bibliography

[1] Luca Allodi. *The Heavy Tails of Vulnerability Exploitation*, pages 133–148. Springer International Publishing Switzerland, 2015.

[2] ArcherySec. Archerysec. https://github.com/archerysec/archerysec, 2021.

[3] Debricked AS. User settings — documentation — debricked. https://debricked.com/docs/administration/user-settings.html#debai, 2021.

[4] Team Ascend. The five stages of vulnerability management. https://blog.teamascend.com/stages-of-vulnerability-management, 2019.

[5] Balbix. A deeper look into risk-based vulnerability management. Technical report, Balbix, Inc., 2018.

[6] Nick Cavalancia. Vulnerability management explained. https://cybersecurity.att.com/blogs/security-essentials/vulnerability-management-explained, 2020.

[7] PCI Security Standards Council. PCI Security Standards. https://www.pcisecuritystandards.org/index.php, 2022.

[8] Mark Dowd, John Mcdonald, and Justin Schuh. *The art of software security assessment : identifying and preventing software vulnerabilities*. Addison-Wesley, 2007.

[9] Center for Disease Control and Prevention. Vulnerability management life cycle. https://www.cdc.gov/cancer/npcr/tools/security/vmlc.htm, 2021.

[10] Recorded Future. Patch what matters with vulnerability management. Technical report, Recorded Future, Inc.

[11] Ayala Goldstein. Vulnerability management – what you need to know. https://www.whitesourcesoftware.com/resources/blog/vulnerability-management/, 2020.

[12] Jonathan Greig. With 18,378 vulnerabilities reported in 2021, NIST records fifth straight year of record numbers. https://www.zdnet.com/article/with-18376-vulnerabilities-found-in-2021-nist-reports-fifth-straight-year-of-record-numbers/, 2021.

[13] Henry Howland. CVSS: Ubiquitous and Broken. *Digital Threats: Research and Practice*, 2021.

[14] Sam Humphries. 4 stages of vulnerability management: A process for risk mitigation. https://www.exabeam.com/information-security/vulnerability-management/, 2020.

[15] Digital Defense Inc. Vulnerability Management: What is It? Process, Best Practices. https://www.digitaldefense.com/blog/vulnerability-management-program-basics/.

[16] Pyxyp Inc. Vuldb - vulnerability database. https://vuldb.com/, 2022.

[17] TechTarget Inc. How to achieve risk-based vulnerability management. Technical report, AT&T, 2020.

[18] Jay Jacobs, Sasha Romanosky, Idris Adjerid, and Wade Baker. Improving vulnerability remediation through better exploit prediction. *Journal of Cybersecurity*, 6, 1 2020.

[19] Jay Jacobs, Sasha Romanosky, Benjamin Edwards, Idris Adjerid, and Michael Roytman. Exploit Prediction Scoring System (EPSS). *Digital Threats: Research and Practice*, 2:1–17, 7 2021.

[20] Information Technology Laboratory. CVSS Severity Distribution Over Time. https://nvd.nist.gov/general/visualizations/vulnerability-visualizations/cvss-severity-distribution-over-time, 2021.

[21] R. A. Miura-Ko and N. Bambos. Securerank: A risk-based vulnerability management scheme for computing infrastructures. In *2007 IEEE International Conference on Communications*, pages 1455–1460. IEEE, 6 2007.

[22] Forum of Incident Response and Security Teams. Common Vulnerability Scoring System SIG. https://www.first.org/cvss/.

[23] Forum of Incident Response and Security Teams. Common Vulnerability Scoring System v3.1: Specification Document. https://www.first.org/cvss/v3.1/specification-document, 2019.

[24] Forum of Incident Response and Security Teams. Exploit Prediction Scoring System (EPSS). https://www.first.org/epss/, 2022.

[25] University of Miami. Vulnerability management. https://security.it.miami.edu/services/vulnerability-management/index.html.

[26] National Institute of Standards and Technology. CVEs and the NVD Process. https://nvd.nist.gov/general/cve-process.

[27] National Institute of Standards and Technology. CVE FAQs - What is the CVE Lifecycle? https://nvd.nist.gov/general/FAQ-Sections/CVE-FAQs#faqLink1, 2022.

[28] National Institute of Standards and Technology. National Vulnerability Database. https://nvd.nist.gov/, 2022.

[29] Bill Olson. Five steps to building a successful vulnerability management program. Technical report, Tenable Network Security, Inc., 2017.

[30] Outpost24. Reduce time to remediation with predictive risk-based vulnerability management. Technical report, Outpost24.

[31] John Pescatore. From SANS: How to Show Business Benefit by Moving to Risk-Based Vulnerability Management. Technical report, Tenable, Inc., 2020.

[32] Rapid7. Vulnerability management process: Scanning, prioritizing, and remediating. https://www.rapid7.com/fundamentals/vulnerability-management-and-scanning/.

[33] Rapid7. 4 key pillars of modern vulnerability risk management. Technical report, 2021.

[34] Jukka Ruohonen. A look at the time delays in CVSS vulnerability scoring. *Applied Computing and Informatics*, 15:129–135, 7 2019.

[35] Miguel Santana, Alan Oliveira de Sá, and Pedro Silva. SecScore - Priorização de Vulnerabilidades de Nova Geração. *INForum: Simpósio de Informática*, pages 357–368, 2022.

[36] Jonathan Spring, Eric Hatleback, Allen D. Householder, Art Manion, and Deana Shick. Towards Improving CVSS. Technical report, Carnegie Mellon University, 2018.

[37] Tenable. How to implement risk-based vulnerability management. Technical report, Tenable, Inc., 2020.

[38] Tenable. Risk-based vulnerability management: Focus on the vulnerabilities that pose the greatest risk. Technical report, Tenable, Inc., 2020.

[39] Tenable. Your answer to the vulnerability overload problem: Risk-based vulnerability management. Technical report, Tenable, Inc., 2020.

[40] Tenable. Overcoming challenges created by disparate vulnerability management tools. Technical report, Tenable, Inc., 2021.

[41] Fordham University. Vulnerability management procedure. https://www.fordham.edu/info/29071/vulnerability_management_procedure, 2021.

[42] Dinesh Chandra Verma. *Principles of Computer Systems and Network Management*. Springer US, 2009.

[43] Paulo Esteves Veríssimo, Nuno Ferreira Neves, and Miguel Pupo Correia. Intrusion-tolerant architectures: Concepts and design. In Rogério de Lemos, Cristina Gacek, and Alexander Romanovsky, editors, *Architecting Dependable Systems*, pages 3–36. Springer Berlin Heidelberg, 2003.

[44] Vulcan. Justify your vulnerability management program. Technical report, Vulcan Cyber Ltd.

[45] Vulcan. Vulnerability management 2020: Navigating to new heights. Technical report, Vulcan Cyber Ltd., 2019.

[46] Michal Walkowski, Maciej Krakowiak, Marcin Jaroszewski, Jacek Oko, and Slawomir Sujecki. Automatic CVSS-based Vulnerability Prioritization and Response with Context Information. *International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–6, 2021.

[47] Michał Walkowski. VMC (OWASP Vulnerability Management Center). `https://github.com/DSecureMe/vmc`, 9 2022.

[48] we45. Orchestron community. `https://github.com/we45/orchestron-community`, 2020.

[49] Michael E Whitman and Herbert J Mattord. *Management of information security*. Cengage Learning, 2014.

Appendix A

CVSS Metrics

A.1 Base Metrics

The Base Metrics of CVSS are all metrics that serve as a basis for the preliminary vulnerability assessment. In Table A.1, these metrics are described in full detail.

A.2 Temporal Metrics

Temporal metrics measure the current state of exploit techniques or code availability, the existence of patches or workarounds, or the certainty of the vulnerability description. In Table A.2, these metrics are described in full detail.

A.3 Environmental Metrics

Environmental metrics allow customization of the CVSS score depending on the importance of the affected IT asset to a user's organization, measured in terms of implemented complementary or alternative security controls, confidentiality, integrity, and availability. In Table A.3, these metrics are described in full detail.

Table A.1: CVSS - Base Metrics

Exploitability Metrics	The exploitability metrics reflect values related to the affected components and properties of a vulnerability that would carry out a successful attack.	
	Attack Vector (AV)	The context in which exploiting the vulnerability is possible - the value will be greater the more an attacker can reach (logically and physically) to exploit the vulnerable component.
	Attack Complexity (AC)	Conditions beyond the attacker's control that must exist to exploit the vulnerability.
	Privileges Required (PR)	Level of privileges an attacker must have before successfully exploiting the vulnerability – the fewer privileges required, the higher the score.
	User Interaction (UI)	Captures the need for a human user, other than the attacker, to participate in the successful compromise of the vulnerable component
Scope	Captures whether a vulnerability in a vulnerable component affects resources in other components beyond its security context.	
Impact Metrics	Captures the effects of a successfully exploited vulnerability on the component that suffers the worst outcome that is most directly and predictably associated with the attack.	
	Confidentiality (C)	Measures the impact on the confidentiality of information resources managed by a software component thanks to a successfully exploited vulnerability – the rating is higher when the loss to the affected component is greater.
	Integrity (I)	Measures the integrity impact of a successfully exploited vulnerability – the rating is higher when the consequence for the affected component is greater.
	Availability (A)	Measures the impact on the availability of the affected component resulting from a successfully exploited vulnerability. As availability refers to the accessibility of information resources, attacks that consume network bandwidth, processor cycles, or disk space impact the availability of the affected component – the rating is higher when the consequence for the impacted component is greater.

Table A.2: CVSS - Temporal Metrics

Exploit Code Maturity (E)	Measures the likelihood that the vulnerability will be attacked and is typically based on the current state of exploit techniques, availability of exploit code, or active in-the-wild exploitation.
Remediation Level (RL)	It is an important factor for prioritization - the less official (from the vendor) and permanent a fix, the higher the vulnerability's score.
Report Confidence (RC)	Measures the degree of confidence that the vulnerability exists and the credibility of known technical details - the more a vulnerability is validated by the vendor or other trusted sources, the higher the score.

Table A.3: CVSS - Environmental Metrics

Security Requirements (CR, IR, AR)	Allows the analyst to customize the CVSS score depending on the importance of the affected IT asset to a user's organization, measured in terms of confidentiality, integrity, or availability - each security requirement has three possible values: Low, Medium, or High.
Modified Base Metrics	Allows the analyst to override individual baseline metrics based on specific user environment characteristics. These include Modified Attack Vector (MAV), Modified Attack Complexity (MAC), Modified Privileges Required (MPR), Modified User Interaction (MUI), Modified Scope (MS), Modified Confidentiality (MC), Modified Integrity (MI) or Modified Availability (MA).

Appendix B

SecScore Dataset

This dataset was used to calculate the R value from the temporal metrics of SecScore. Table B.1 is a summarized version of the dataset with only the essential data. The full dataset is available on this thesis entry in ULisboa's repository.

Table B.1: SecScore Dataset

Created in VulDB	**Vulnerable Product**	**Exploit Code Maturity (E)**	**Remediation Level (RL)**	**CVE**	**Reserved CVE (date)**	**Advisory disclosed (date)**
07/03/2022	Mozilla Firefox/Firefox ESR/Thunderbird XSLT	High	Official Fix	CVE-2022-26485	04/03/2022	07/03/2022
07/03/2022	Mozilla Firefox/Firefox ESR/Thunderbird WebGPU IPC Framework	High	Official Fix	CVE-2022-26486	04/03/2022	07/03/2022
11/02/2022	Apple iOS/iPadOS WebKit	High	Official Fix	CVE-2022-22620	05/01/2022	11/02/2022
11/02/2022	Apple macOS WebKit	High	Official Fix	CVE-2022-22620	05/01/2022	11/02/2022
11/02/2022	Apple Safari WebKit	High	Official Fix	CVE-2022-22620	05/01/2022	11/02/2022
15/02/2022	Google Chrome Animation	High	Official Fix	CVE-2022-0609	14/05/2022	15/02/2022
18/02/2022	RigoBlock Dragos set-MultipleAllowances	High	Not Defined	CVE-2022-25335	18/02/2022	18/02/2022

Created in VulDB	Vulnerable Product	Exploit Code Maturity (E)	Remediation Level (RL)	CVE	Reserved CVE (date)	Advisory disclosed (date)
14/01/2022	NUUO NVR-Mini2 TAR Archive handle_import_user.php	High	Official Fix	CVE-2022-23227	14/01/2022	14/01/2022
19/10/2021	Apple watchOS IOMobileFrameBuffer	High	Official Fix	CVE-2021-30807	13/04/2021	26/07/2021
19/10/2021	Apple macOS IOMobileFrameBuffer	High	Official Fix	CVE-2021-30807	13/04/2021	26/07/2021
23/10/2021	SuiteCRM Log File Name Setting	High	Official Fix	CVE-2021-42840	22/10/2021	23/10/2021
23/10/2021	BQE BillQuick Web Suite xp_cmdshell	High	Official Fix	CVE-2021-42258	11/10/2021	23/10/2021
29/10/2021	Google Chrome	High	Official Fix	CVE-2021-38000	03/08/2021	29/10/2021
29/10/2021	Google Chrome V8	High	Official Fix	CVE-2021-38003	03/08/2021	29/10/2021
07/09/2021	Zoho ManageEngine ADSelfService Plus REST API	High	Official Fix	CVE-2021-40539	06/09/2021	07/09/2021
14/09/2021	Apple iOS/iPadOS CoreGraphics FORCEDENTRY	High	Official Fix	CVE-2021-30860	13/04/2021	13/09/2021
15/09/2021	Google Chrome V8	High	Official Fix	CVE-2021-30632	13/04/2021	15/09/2021

Created in VulDB	Vulnerable Product	Exploit Code Maturity (E)	Remediation Level (RL)	CVE	Reserved CVE (date)	Advisory disclosed (date)
15/09/2021	Google Chrome Indexed DB API	High	Official Fix	CVE-2021-30633	13/04/2021	15/09/2021
21/09/2021	Apple macOS WebKit	High	Official Fix	CVE-2021-30858	13/04/2021	13/09/2021
24/09/2021	Apple iOS XNU Kernel	High	Official Fix	CVE-2021-30869	13/04/2021	24/09/2021
25/09/2021	Apple macOS XNU Kernel	High	Official Fix	CVE-2021-30869	13/04/2021	24/09/2021
06/07/2021	Kaseya Virtual System Administrator	High	Official Fix	CVE-2021-30116	02/04/2021	06/07/2021
17/07/2021	Google Chrome V8	High	Official Fix	CVE-2021-30563	13/04/2021	17/07/2021
27/07/2021	Apple iOS/iPadOS IOMobileFrameBuffer	High	Official Fix	CVE-2021-30807	13/04/2021	26/07/2021
30/06/2021	Western Digital WD My Book Live/WD My Book Live Duo Administrator API	High	Workaround	CVE-2021-35941	29/06/2021	30/06/2021
06/05/2021	Apple iOS WebKit	High	Official Fix	CVE-2021-30666	13/04/2021	03/05/2021
06/05/2021	Apple iOS/iPadOS WebKit	High	Official Fix	CVE-2021-30665	13/04/2021	03/05/2021
06/05/2021	Apple iOS WebKit Storage	High	Official Fix	CVE-2021-30661	13/04/2021	03/05/2021
06/05/2021	Apple iOS/iPadOS WebKit	High	Official Fix	CVE-2021-30663	13/04/2021	03/05/2021

Created in VulDB	Vulnerable Product	Exploit Code Maturity (E)	Remediation Level (RL)	CVE	Reserved CVE (date)	Advisory disclosed (date)
06/05/2021	Apple watchOS WebKit	High	Official Fix	CVE-2021-30665	13/04/2021	03/05/2021
06/05/2021	Apple macOS WebKit	High	Official Fix	CVE-2021-30665	13/04/2021	03/05/2021
06/05/2021	Apple macOS WebKit	High	Official Fix	CVE-2021-30663	13/04/2021	03/05/2021
06/05/2021	Apple Safari WebKit	High	Official Fix	CVE-2021-30665	13/04/2021	04/05/2021
06/05/2021	Apple Safari WebKit	High	Official Fix	CVE-2021-30663	13/04/2021	04/05/2021
25/05/2021	Apple tvOS WebKit	High	Official Fix	CVE-2021-30663	13/04/2021	24/05/2021
25/05/2021	Apple macOS TCC	High	Official Fix	CVE-2021-30713	13/04/2021	24/05/2021
12/04/2021	Thrive Optimize Plugin REST API	High	Official Fix	CVE-2021-24219	14/01/2021	12/04/2021
12/04/2021	Thrive Legacy Rise Theme REST API Endpoint	High	Official Fix	CVE-2021-24220	14/01/2021	12/04/2021
27/04/2021	Apple macOS WebKit Storage	High	Official Fix	CVE-2021-30661	13/04/2021	26/04/2021
27/04/2021	Apple tvOS WebKit Storage	High	Official Fix	CVE-2021-30661	13/04/2021	26/04/2021