Universidade de Aveiro

U.A_V.M

Ana Filipe (93350), João Fernandes (93460), João Figueiredo (93250)

Olavo Estima (93249), Pedro Pereira (93196), Tiago Pedrosa (93389)

Projeto Engenharia Computadores Informática

Departamento de Eletrónica, Telecomunicações e Informática

March 24, 2022

Abstract

The Internet was arguably the biggest Invention of the last century. It created a way to get information to reach the entire world. Without a doubt, it was a huge leap forward in communication throughout the world. In the trying times we face today, with people locked in their homes for lengthy periods, there was an even higher than usual internet usage of websites dedicated to online purchases, social media, telecommuting...

With all this new online presence, there is a lot of important personal information like credit card information, account passwords, even the cameras on personal computers. Nowadays we do everything through screens that can leave big parts of our lives exposed. Therefore, it is necessary for our information to be kept safe. Nobody wants strangers to have access to our house address. That is why, from the moment that a *website* asks us for any personal information, we have to make sure no unwanted third party can have access to it .

All programs have vulnerabilities, some small, with little to no real consequences to the functionality of a *website*, and some catastrophic, that can lead to things like exposure of database information and manipulation containing thousands or millions of users' information, making the *website* inaccessible, remote control of personal computers and malware installation.

To combat possible threats to the Universidade de Aveiro (in our case), our aim is to create a platform to analyse all university domains, doing a port to port evaluation and grading the security of a domain from 1 to 10, 10 being the worst and extremely unsafe.

It shall be constantly analysed. Whenever a worrying vulnerability is found, it should notify the current domain user as well as qualified personnel in order to resolve the issue. With this project, we hope to achieve greater security to the servers and the whole online community in the university campus.

Resumo

A internet foi a maior invenção do século passado. Fez com que a informação pudesse chegar a todo o planeta através de um clique. Sem dúvida que foi um enorme avanço em termos de comunicação a nível mundial. Nestes tempos de pandemia, com as pessoas mais tempo fechadas em casa, houve uma maior exposição à internet para compras on-line, comunicação com amigos através das redes sociais, tele-trabalho...
Esse aumento de atividade levou a uma maior exposição de privacidade como por exemplo quando fazemos um pagamento pela internet e temos de colocar as credenciais do nosso cartão de crédito, passwords dos nossos perfis nas redes sociais e quando estamos com a câmara ligada para tele-trabalho ou para falar com amigos ou familiares. Fazemos praticamente tudo através da internet e muita da nossa vida está exposta nela.
Com tanta informação, é necessário haver um nível de segurança máximo. Ninguém quer que terceiros saibam onde moramos nem que tenham acesso aos nossos cartões bancários. É por isso que, no momento em que um *site* nos peça alguma informação pessoal, temos de ter a certeza de que mais ninguém terá acesso a essa informação.
Contudo, todos os programas têm as suas vulnerabilidades, algumas que não causam qualquer efeito no funcionamento dos *sites* e outras que podem ser catastróficos a nível de acesso a bases de dados que contêm informação de milhares ou até milhões de pessoas, fazer com que o *site* deixe de ser acessível, obter controlo remoto de computadores pessoais (por exemplo para a sua destruição) e até instalação de malware.
Para combater essas possíveis ameaças, no nosso caso ameaças contra a Universidade de Aveiro, queremos criar uma plataforma que irá análisar todos os domínios expostos fazendo uma avaliação porto a porto e dando como resultado um valor, entre 1 e 10, sendo 10 um nível crítico, isto é, nesse domínio foi encontrada uma vulnerabilidade que pode ser facilmente explorada por alguém que queira realizar um ataque.
Irá estar a analisar esses domínios constantemente e, sempre que for encontrada alguma falha preocupante, irá notificar alguém qualificado para resolver o problema. Com este projeto esperamos conseguir com que haja uma maior segurança no que trata aos servidores e toda a comunidade on-line da Universidade de Aveiro.

Acknowledgements

Contents

List of Figures

Abbreviations

CVSS - Common Vulnerability Scoring System
CVE - Common Vulnerabilities and Exposures
ASM - Attack Surface Monitoring
UA - University of Aveiro
OS - Operative System
IP - Internet Protocol
NSE - Nmap Scripting Engine
ACL - Access Control List
DNS - Domain Name System
MX - Mail Exchanger
HTML - HyperText Markup Language
CSS - Cascading Style Sheets
SQL - Structured Query Language
OSI - Open System Interconnection
IDP - Internally Displaced Person
SSL - Secure Sockets Layer
IT - Information Technology

Chapter 1

Introduction

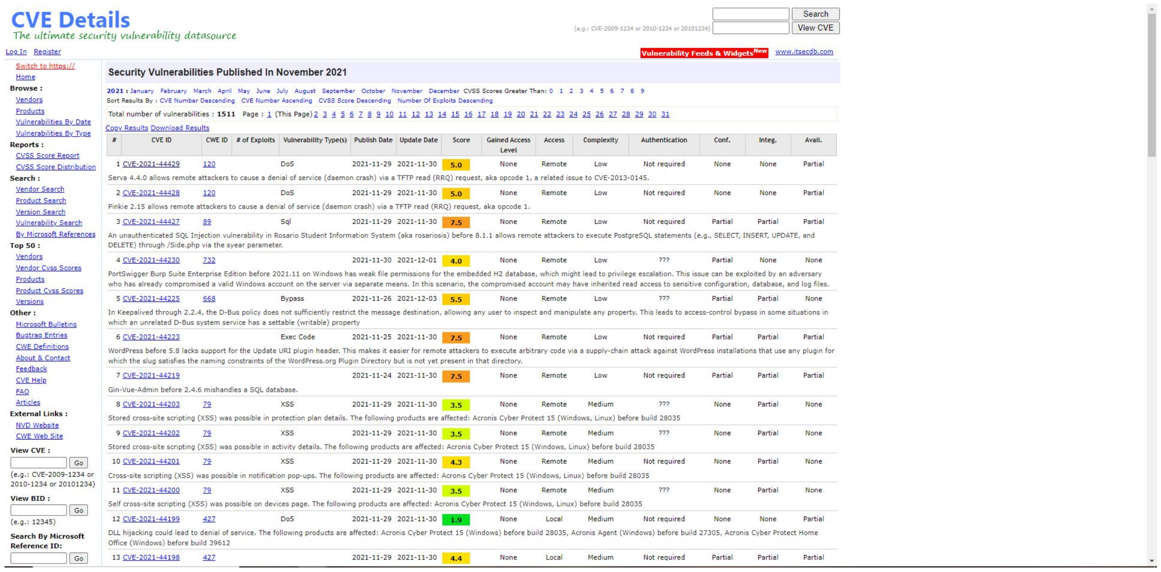
The **University of Aveiro** has a large number of domains on the internet. Some more important than others, but nonetheless, all connected to the university's servers.
This raises a security problem. Throughout the years, the university has been a target of countless cyber attacks, some are noticed and others we know nothing about. Some of these attacks are due to the large amount of information that the organization possesses, such as student personal information, companies that have projects being developed on campus and research that is being made all the time.
Its has been proposed to us to develop a platform that makes constant scanning for the vulnerabilities in all the campus network. We'll do it in a more user friendly way so that every person using it can have an easy time figuring out the problems detected and resolve them in a efficient way.
Our objective for this project is to provide a platform to give a more detailed view of the possible vulnerabilities that can be exploited by unwanted third parties and means to reduce their amount.
With our project we hope to have a more secure network in the university.

What is a Vulnerability?

A cybersecurity vulnerability is a flaw in a system that can leave it open to attack or any weakness within a computer system itself that leaves information exposed to outside threats. New vulnerabilities appear on a constant basis and systems require updates, patches and knowing what causes them and how to deal with them.

What is a CVE?

Common Vulnerabilities and Exposures, or **CVE**, is a list of records of all kinds of vulnerabilities related to information security that are publicly known. Within its records, it contains information of thousands of these accounts dating back decades, along with a brief description of the repercussions of such weaknesses, as well as information about Vulnerability Type(s), CVSS (Common Vulnerability Scoring System) score, gained access level, complexity, which systems it affects and more.



Security Vulnerabilities Published In November 2021

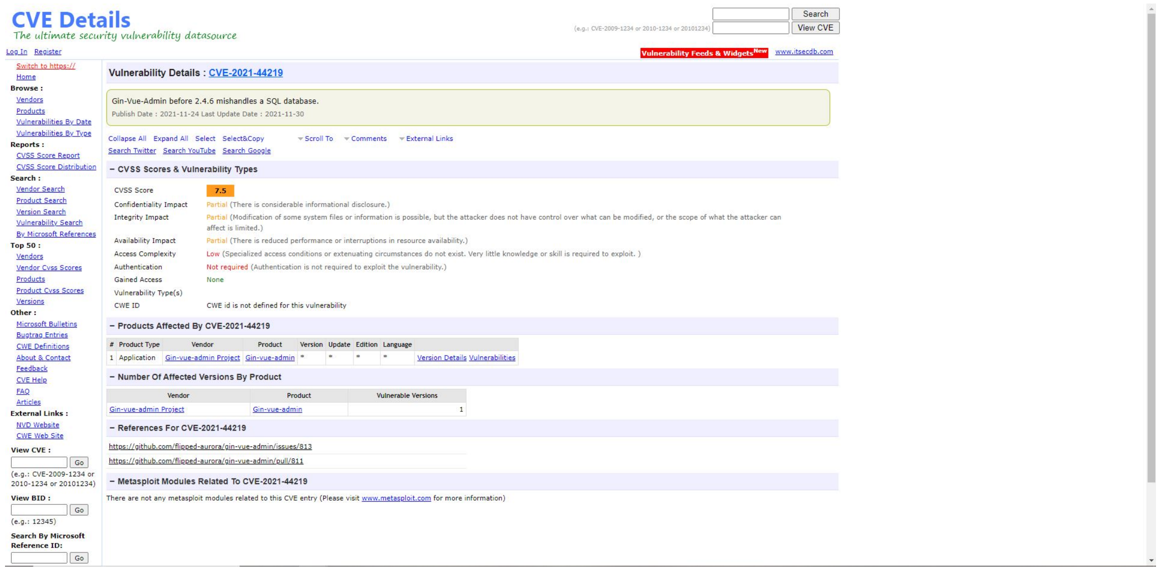
2021 : January February March April May June July August September October November December CVSS Scores Greater Than: 0 1 2 3 4 5 6 7 8 9
Sort Results By : CVE Number Descending CVE Number Ascending CVSS Score Descending Number Of Exploits Descending

Total number of vulnerabilities : **1511** Page : 1 (This Page) 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31

Copy Results Download Results

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2021-44429	120		DoS	2021-11-29	2021-11-30	5.0	None	Remote	Low	Not required	None	None	Partial
Serva 4.4.0 allows remote attackers to cause a denial of service (daemon crash) via a TFTP read (RRQ) request, aka opcode 1, a related issue to CVE-2013-0145.														
2	CVE-2021-44428	120		DoS	2021-11-29	2021-11-30	5.0	None	Remote	Low	Not required	None	None	Partial
Pinkie 2.15 allows remote attackers to cause a denial of service (daemon crash) via a TFTP read (RRQ) request, aka opcode 1.														
3	CVE-2021-44427	89		Sql	2021-11-29	2021-11-30	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
An unauthenticated SQL Injection vulnerability in Rosario Student Information System (aka rosariosis) before 8.1.1 allows remote attackers to execute PostgreSQL statements (e.g., SELECT, INSERT, UPDATE, and DELETE) through /Side.php via the syear parameter.														
4	CVE-2021-44230	732			2021-11-30	2021-12-01	4.0	None	Remote	Low	???	Partial	None	None
PortSwigger Burp Suite Enterprise Edition before 2021.11 on Windows has weak file permissions for the embedded H2 database, which might lead to privilege escalation. This issue can be exploited by an adversary who has already compromised a valid Windows account on the server via separate means. In this scenario, the compromised account may have inherited read access to sensitive configuration, database, and log files.														
5	CVE-2021-44225	668		Bypass	2021-11-26	2021-12-03	5.5	None	Remote	Low	???	Partial	Partial	None
In Keepalived through 2.2.4, the D-Bus policy does not sufficiently restrict the message destination, allowing any user to inspect and manipulate any property. This leads to access-control bypass in some situations in which an unrelated D-Bus system service has a settable (writable) property														
6	CVE-2021-44223			Exec Code	2021-11-25	2021-11-30	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
WordPress before 5.8 lacks support for the Update URI plugin header. This makes it easier for remote attackers to execute arbitrary code via a supply-chain attack against WordPress installations that use any plugin for which the slug satisfies the naming constraints of the WordPress.org Plugin Directory but is not yet present in that directory.														
7	CVE-2021-44219				2021-11-24	2021-11-30	7.5	None	Remote	Low	Not required	Partial	Partial	Partial
Gin-Vue-Admin before 2.4.6 mishandles a SQL database.														
8	CVE-2021-44203	79		XSS	2021-11-29	2021-11-30	3.5	None	Remote	Medium	???	None	Partial	None
Stored cross-site scripting (XSS) was possible in protection plan details. The following products are affected: Acronis Cyber Protect 15 (Windows, Linux) before build 28035														
9	CVE-2021-44202	79		XSS	2021-11-29	2021-11-30	3.5	None	Remote	Medium	???	None	Partial	None
Stored cross-site scripting (XSS) was possible in activity details. The following products are affected: Acronis Cyber Protect 15 (Windows, Linux) before build 28035														
10	CVE-2021-44201	79		XSS	2021-11-29	2021-11-30	4.3	None	Remote	Medium	Not required	None	Partial	None
Cross-site scripting (XSS) was possible in notification pop-ups. The following products are affected: Acronis Cyber Protect 15 (Windows, Linux) before build 28035														
11	CVE-2021-44200	79		XSS	2021-11-29	2021-11-30	3.5	None	Remote	Medium	???	None	Partial	None
Self cross-site scripting (XSS) was possible on devices page. The following products are affected: Acronis Cyber Protect 15 (Windows, Linux) before build 28035														
12	CVE-2021-44199	427		DoS	2021-11-29	2021-11-30	1.9	None	Local	Medium	Not required	None	None	Partial
DLL hijacking could lead to denial of service. The following products are affected: Acronis Cyber Protect 15 (Windows) before build 28035, Acronis Agent (Windows) before build 27305, Acronis Cyber Protect Home Office (Windows) before build 39612														
13	CVE-2021-44198	427			2021-11-29	2021-11-30	4.4	None	Local	Medium	Not required	Partial	Partial	Partial

Figure 1.1: *Example of the list of security vulnerabilities published in November 2021*



Vulnerability Details : CVE-2021-44219

Gin-Vue-Admin before 2.4.6 mishandles a SQL database.
Publish Date : 2021-11-24 Last Update Date : 2021-11-30

− CVSS Scores & Vulnerability Types

CVSS Score	7.5
Confidentiality Impact	Partial (There is considerable informational disclosure.)
Integrity Impact	Partial (Modification of some system files or information is possible, but the attacker does not have control over what can be modified, or the scope of what the attacker can affect is limited.)
Availability Impact	Partial (There is reduced performance or interruptions in resource availability.)
Access Complexity	Low (Specialized access conditions or extenuating circumstances do not exist. Very little knowledge or skill is required to exploit.)
Authentication	Not required (Authentication is not required to exploit the vulnerability.)
Gained Access	None
Vulnerability Type(s)	
CWE ID	CWE id is not defined for this vulnerability

− Products Affected By CVE-2021-44219

#	Product Type	Vendor	Product	Version	Update	Edition	Language	
1	Application	Gin-vue-admin Project	Gin-vue-admin	*	*	*	*	Version Details Vulnerabilities

− Number Of Affected Versions By Product

Vendor	Product	Vulnerable Versions
Gin-vue-admin Project	Gin-vue-admin	1

− References For CVE-2021-44219

https://github.com/flipped-aurora/gin-vue-admin/issues/813
https://github.com/flipped-aurora/gin-vue-admin/pull/811

− Metasploit Modules Related To CVE-2021-44219

There are not any metasploit modules related to this CVE entry (Please visit www.metasploit.com for more information)

Figure 1.2: *Information about a specific vulnerability*

Chapter 2

The State of the Art

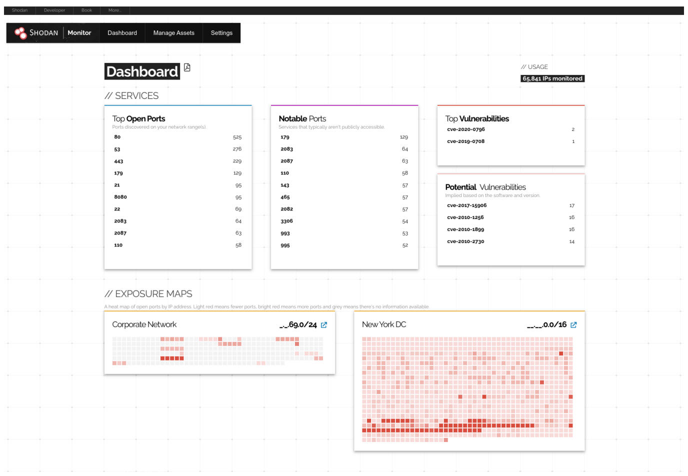
2.1 Related Work

2.1.1 Shodan

Shodan [1] is a search engine for Internet-connected devices. You can, for example see what is currently connected to the internet within your network space. It helps to keep track of all your devices across the internet. It detects data leaks, phishing compromised databases and more as they say in their websites. It is scalable ass you can monitor one or more IPs at the same time. The platform itself is very user friendly as it shows all the information needed in a very simple and compact way. It gives you gives you the hability to see if a problem has a fix or if there is a way arround the problem for the time being.

The **Shodan Monitor** within the Shodan website is the part that we will be focusing more. It is very similiar to the goal of our project which is to monitor the entire network of the **University of Aveiro**.

Shodan Monitor is used by a lot of people who have different network sizes they want to be monitored. You can keep track of all the domains with real time scans or timed scans. It notifies you if something goes wrong in your network. It makes periodic scans of the IPs but you can also request for immediate scanning of a certain domain.



A heat map of open ports by IP address. Light red means fewer ports, bright red means more ports and grey means there's no information available.

Top **Open Ports**
Ports discovered on your network range(s).

80	525
53	276
443	229
179	129
21	95
8080	95
22	69
2083	64
2087	63
110	58

Notable Ports
Services that typically aren't publicly accessible.

179	129
2083	64
2087	63
110	58
143	57
465	57
2082	57
3306	54
993	53
995	52

Top **Vulnerabilities**

cve-2020-0796	2
cve-2019-0708	1

Potential Vulnerabilities
Implied based on the software and version.

cve-2017-15906	17
cve-2010-1256	16
cve-2010-1899	16
cve-2010-2730	14

Figure 2.1: *Shodan Monitor Dashboard*

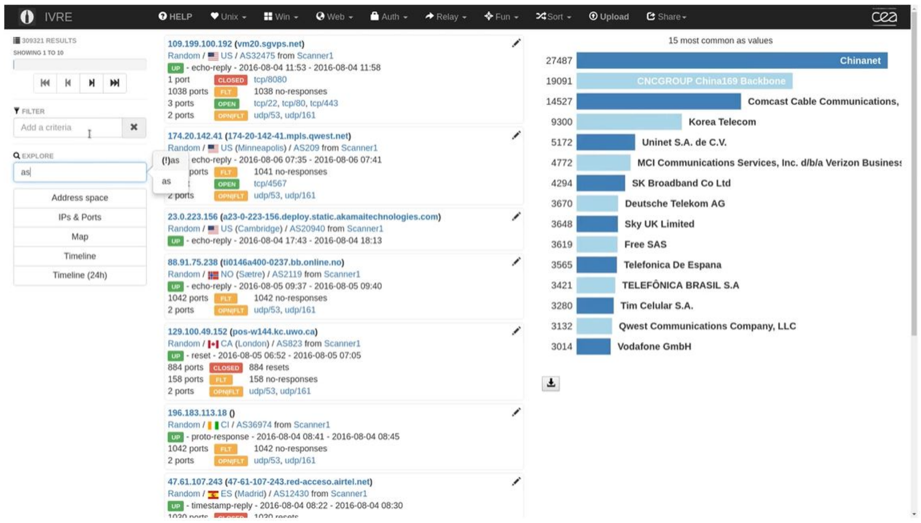
2.1.2 Binary Edge

Binary Edge [2] is another platform similar to **Shodan**. It has a plethora of features about analysis of domains, IP, DNS and more. But for our project we are going to focus in the domain monitoring aspect.
This company, in 29 May 2020, launched a project called ASM (short for Attack Surface Monitoring). It lets you scan your network and keep you updated if something goes wrong. It can handle big organizations and provides a clean, user friendly dashboard with all the crucial information and some statistics to show how the network is progressing. It makes an association between DNS and IP addresses to show a more complete list of assets. Binary Edge identifies vulnerabilities detected and scales them from low to critical. It can warn you about an SSL certificate that has expired, check if domains are being used for phishing and it can also monitor torrent downloads. As we all know they often come infected with malware.

2.1.3 IVRE

IVRE [3] (also known as **DRUNK** or Dynamic Recon of UNKnown networks) is an open source framework for network recon written in Python. It relies on powerful open-source tools like Nmap to gather intelligence from a network, actively or passively.

IVRE can prove itself useful in several different scenarios including the creation of our own **Shodan**-like service which, in our case, can be more specified towards a single network.



309321 RESULTS
SHOWING 1 TO 10

```
109.199.100.192 (vm20.sgvps.net)
Random / US / AS32475 from Scanner1
UP - echo-reply - 2016-08-04 11:53 - 2016-08-04 11:58
1 port       CLOSED    tcp/8080
1038 ports   FLT       1038 no-responses
3 ports      OPEN      tcp/22, tcp/80, tcp/443
2 ports      OPN|FLT   udp/53, udp/161

174.20.142.41 (174-20-142-41.mpls.qwest.net)
Random / US (Minneapolis) / AS209 from Scanner1
 echo-reply - 2016-08-06 07:35 - 2016-08-06 07:41
   ports     FLT       1041 no-responses
             OPEN      tcp/4567
2 ports      OPN|FLT   udp/53, udp/161

23.0.223.156 (a23-0-223-156.deploy.static.akamaitechnologies.com)
Random / US (Cambridge) / AS20940 from Scanner1
UP - echo-reply - 2016-08-04 17:43 - 2016-08-04 18:13

88.91.75.238 (ti0146a400-0237.bb.online.no)
Random / NO (Sætre) / AS2119 from Scanner1
UP - echo-reply - 2016-08-05 09:37 - 2016-08-05 09:40
1042 ports   FLT       1042 no-responses
2 ports      OPN|FLT   udp/53, udp/161

129.100.49.152 (pos-w144.kc.uwo.ca)
Random / CA (London) / AS823 from Scanner1
UP - reset - 2016-08-05 06:52 - 2016-08-05 07:05
884 ports    CLOSED    884 resets
158 ports    FLT       158 no-responses
2 ports      OPN|FLT   udp/53, udp/161

196.183.113.18 ()
Random / CI / AS36974 from Scanner1
UP - proto-response - 2016-08-04 08:41 - 2016-08-04 08:45
1042 ports   FLT       1042 no-responses
2 ports      OPN|FLT   udp/53, udp/161

47.61.107.243 (47-61-107-243.red-acceso.airtel.net)
Random / ES (Madrid) / AS12430 from Scanner1
UP - timestamp-reply - 2016-08-04 08:22 - 2016-08-04 08:30
```

15 most common as values

27487	Chinanet
19091	CNCGROUP China169 Backbone
14527	Comcast Cable Communications,
9300	Korea Telecom
5172	Uninet S.A. de C.V.
4772	MCI Communications Services, Inc. d/b/a Verizon Busines
4294	SK Broadband Co Ltd
3670	Deutsche Telekom AG
3648	Sky UK Limited
3619	Free SAS
3565	Telefonica De Espana
3421	TELEFÔNICA BRASIL S.A
3280	Tim Celular S.A.
3132	Qwest Communications Company, LLC
3014	Vodafone GmbH

Figure 2.2: *IVRE Dashboard*

2.1.4 Spyse

Spyse [4] is a search and analysis platform that reveals data interrelationships and provides relevant security details on entities of the entire Internet. It carries out a great deal of work starting from scanning ports and determining of technologies, ending with the gathering of certificates' details and business information on companies represented on the network.
Spyse specializes in large-scale scans and big data processing fields and has a vast number of technologies it detects from programming languages (e.g. Ruby, PHP, Java, ...) to advertising (e.g. Yahoo Advertising, AdRoll, ...)

2.2 CVSS

The **CVSS** [5] is the Common Vulnerability Scoring System which gives us a score, normally from 1 to 10 (10 being a system with no security at all). However there are two version that actively in use, CVSS v2 and CVSS v3. For both we have three major aspects to the calculation of the CVSS, **Base Score**, **Environmental Score** and **Temporal Score**.

2.2.1 Base Score

For this score we take in consideration four factors:

- **Attack Vector (AV) -** It reflects how the exploitation can be achieved. There are four possible ways:
  - **Network (N) -** the vulnerable component is bound to the network stack and the attacker's path is through OSI [6] layer 3;
  - **Adjacent -** the vulnerable component is bound to the network stack but the attacker has to be in same shared physical or logical network (e.g. Bluetooth or local IP subnet) but it cannot be performed across an OSI layer 3 boundary (e.g. a router);
  - **Local (L) -** the attacker can only make attacks via read/write/execute capabilities. He often relies on the users of that services to run the malware;
  - **Physical (P) -** the attacker has to have physical interaction with the system. An example of such an attack is a cold boot attack which allows an attacker to access to disk encryption keys after gaining physical access to the system.
- **Attack Complexity (AC) -** describes the conditions that have to exist so the attacker may exploit the vulnerability. The conditions are beyond the attacker's control.
  - **Low (L) -** all the condition are meet so the attacks to the same vulnerability are always successful;
  - **High (H) -** the conditions may or may not be meet for the attack to be successful. The attacker must do a more in-depth research about the vulnerability, prepare the target environment to improve exploit reliability or inject herself into the logical network path between the target and the resource requested by the victim in order to read and/or modify network communications.
- **Privileges Required (PR) -** describes the level of privileges required to successfully exploit a vulnerability.
  - **None (N) -** the attacker can make an attack without requiring any access to settings or files;
  - **Low (L) -** the attacker as a normal user like privileges which means he can as to change some settings that are normally changeable. The attack can cause an impact only to non-sensitive resources;

  - **High (H) -** the attacker is provided with administrative like privileges and has the control over the vulnerable component that could affect component-wide settings and files.

- **User Interaction (UI) -** describes the requirement for a user, other than the attacker, to participate in the successful compromise of the vulnerable component. It determines if the attacker can work alone or has to have the participation of a user.
  - **Non (N) -** can be exploited without the interaction of a user;
  - **Required (R) -** it requires the interaction of a user to perform a successful attack.

- **Scope (S) -** it refers to the collection of privileges defined by a computing authority when granting access to computing resources. The scope changes when a software component governed by one authorization scope affects the resources governed by another scope.
  - **Unchanged (U) -** the exploited vulnerability only affects the resources governed by the same authority;
  - **Changed (C) -** the exploited vulnerability affects resources beyond the authorization in that component.

- **Confidentiality Impact (C) -** measures the confidentiality of information accessed when exploiting the vulnerability of a component.
  - **None (N) -** there is no loss of confidentiality;
  - **Low (L) -** there is access to some information of that component but the attacker doesn't control what information is obtained or the amount of loss is constrained. The information does not cause a loss to the impacted component;
  - **High (H) -** there is a total loss of confidentiality, resulting in the loss of high confidential information like passwords or private encryption keys.

- **Integrity Impact (I) -** it refers to the trustworthiness and veracity of information.
  - **None (N) -** there is no loss integrity in the component;
  - **Low (L) -** data can be modified in that affected component but the attacker does not have control of the consequence of the modification. The changed data does not have a direct impact on the component.
  - **High (H) -** the attacker can modify all files contained in the impacted component. Malicious modification would have a direct impact over the affected component.

- **Availability Impact (A) -** it refers to the loss of the availability of the impacted components such as information, files, web, database.
  - **None (N) -** there is no impact to the availability of the impacted component;
  - **Low (L) -** the attacker can perform interruptions in the availability of the component but doesn't have the ability to completely deny service;
  - **High (H) -** the attacker can fully deny access to the resources impacted by the attack. The loss can be sustained, the denial of service continues while the attack is active, or persistence, the services continues denied even after the attack as ended.

2.2.2 Temporal Score

Measures the current state of the exploit techniques or code availability, the existence of patches or the confidence that one has in the description of a vulnerability.

- **Exploit Code Maturity (E) -** measures the likelihood of the vulnerability being attacked. It's based on the current exploit techniques, exploit code availability or active exploitation.
  - **Not Defined (X) -** this metric is skipped;
  - **High (H) -** the exploit code works in every situation or its being delivered autonomously by a worm or a virus. When the network is scanned it will show exploitation attempts. The exploit is widely available;
  - **Functional (F) -** the functional exploit code is available and it works in most situations;
  - **Proof-of-Concept (P) -** the functional code is available but it doesn't work in all situation. It might require a skilled attacker to modify the code if needed;
  - **Unproven (U) -** the exploit is only theoretical. No exploit code is available.
- **Remediation Level (RL) -** measures which type of patches exist to fix the exploit.
  - **Not Defined (X) -** this metric is skipped;
  - **Unavailable (U) -** there is no solution available to the vulnerability;
  - **Workaround (W) -** there is a solution to the problem but its unofficial. The users might create a patch of their own or mitigate the vulnerability;
  - **Temporary Fix (T) -** there is an official but temporary fix. Its similar to the **Workaround** but its issued by the vendor;
  - **Official Fix (O) -** there is an official final fix available.
- **Report Confidence (RC) -** measures the degree of confidence in the existence of the vulnerability and the credibility of the known technical details.
  - **Not Defined (X) -** this metric is skipped;
  - **Confirmed (C) -** there are detailed reports or a functional reproduction of the problem and the vendor acknowledges the exploit;
  - **Reasonable (R) -** there are detailed reports of the vulnerability but the researchers do not fully confirm the problem;
  - **Unknown (U) -** there are reports of the causes of a vulnerability but the true nature of the vulnerability is unknown.

2.2.3 Environmental Score

This score is customized depending on the importance of the IT asset to an organization. It affects Confidentiality, Integrity and Availability.

- **Security Requirements (CR, IR, AR) -** the analyst can assign a greater value to Availability relative to Confidentiality and Integrity.
  - **Not Defined (X) -** this metric is skipped;
  - **Low (L) -** the loss only makes **limited damage** to the organization or the people associated to it;
  - **Medium (M) -** the loss makes **serious damage** to the organization or the people associated to it;
  - **High (H) -** the loss only makes **catastrophic damage** to the organization or the people associated to it;

2.2.4 Rating

Making all the calculation with the metrics above we'll have a result between 0 and 10 showing us the severity of the vulnerability.

- **None -** the CVSS score is **0.0**;
- **Low -** the CVSS score is **0.1 - 3.9**;
- **Medium -** the CVSS score is **4.0 - 6.9**;
- **High -** the CVSS score is **7.0 - 8.9**;
- **Critical -** the CVSS score is **9.0 - 10.0**;

2.2.5 Example

The vector below is obtained by the CVE-2020-15778

AV:L/AC:L/PR:N/UI:R/S:U/C:H/I:H/A:H

With this vector we know:

- The Attack Vector is low;
- The Attack Complexity is low;
- The Privileges Required is none;
- The User Interaction is required;
- The Scope is Unchanged;
- The Confidentiality Impact is high;
- The Integrity Impact is high;
- The Availability Impact is high;

2.2.6 Overall Score Decision Tree

The **Overall Score** is based on 3 main types of scores: **Base Score**, **Environmental Score** and **Temporal Score**. The final score (Overall Score) is obtained by a simple decision tree:

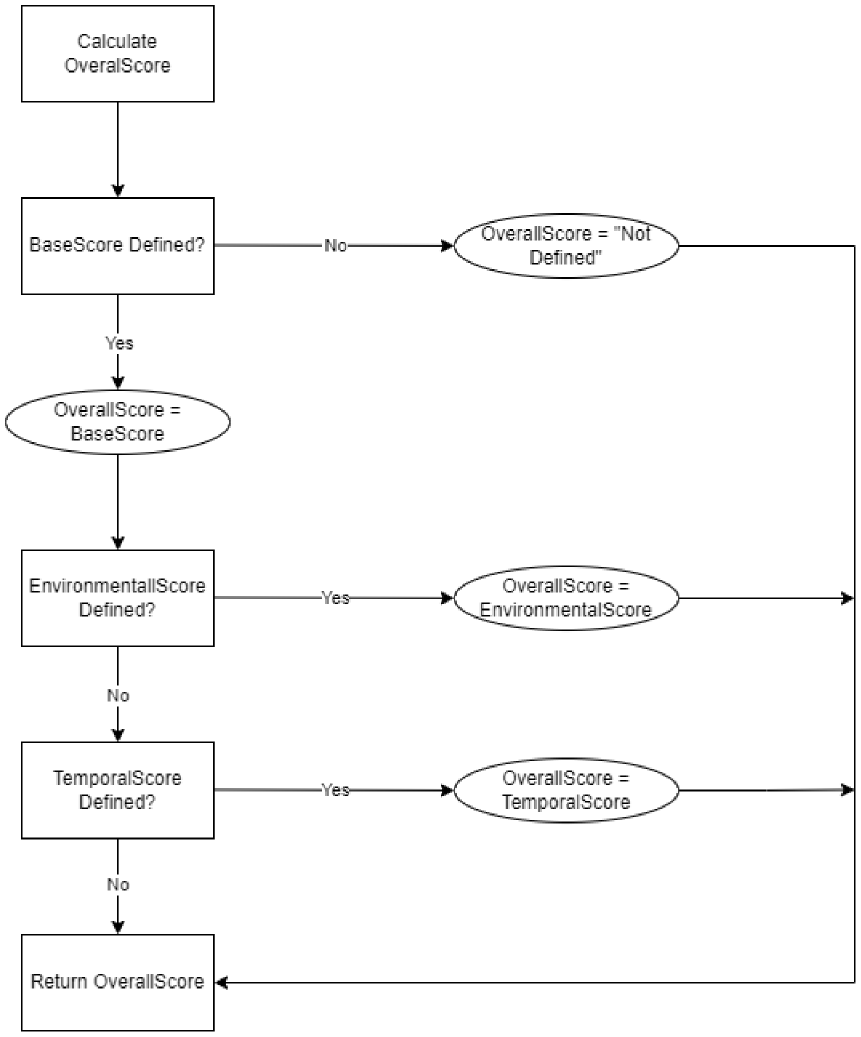


Figure 2.3: *CVSS v2 decision tree*

2.2.7 CVSS v3 Equations

The calculation of the CVSS v3:

- **Base Score**
  Its a function of the Impact and Exploitability sub score
  - Base Impact Sub Score
    $$ISC_{base} = 1 - [(1Impact_{Conf}) * (1 - Impact_{Integ}) * (1 - Impact_{Avail})]$$
    If Scope Unchanged:
    $$ISC = 6.42 * ISC_{base}$$
    If Scope Changed:
    $$ISC = 7.52 * [ISC_{base} - 0.029] - 3.25 * [ISC_{base} - 0.02]^{15}$$
  - Exploitability sub Score
    $$8.22 * AttackVector * AttackComplexity * PrivilegeRequired * UserInteraction$$
  - The main function is (if the score is less than 0 its considered as 0):
    Scope Unchanged:
    $$Round\,Up(Minimum[1.08 * (Impact + Exploitability), 10])$$
    Scope Changed:
    $$Round\,Up(Minimum[(Impact + Exploitability), 10])$$
- **Temporal Score**
  $$Round\,up(BaseScore \times ExploitCodeMaturity \times RemediationLevel \times ReportConfidence)$$
- **Environmental Score**
  - Modified Impact Sub Score:
    $$ISC_{Modified} = Minimum[[1 - (1 - M.I_{Conf} \times CR) \times (1 - M.I_{Integ} \times IR) \times (1 - M.I_{Avail} \times AR)], 0.915]$$
    If Modified Scope Unchanged:
    $$ISC = 6.42 * ISC_{base}$$

If Modified Scope Changed:

$$ISC = 7.52 * [ISC_{base} - 0.029] - 3.25 * [ISC_{base} - 0.02]^{15}$$

– The main function is (if the score is less than 0 its considered as 0):
Modified Scope Unchanged:

$$Round\ up(Round\ up(Minimum[(M.\,Impact + M.\,Exploitability), 10]) \times ExploitCodeMaturity \times RemediationLevel \times ReportConfidence)$$

Modified Scope Changed:

$$Round\ up(Round\ up(Minimum[1.08 \times (M.\,Impact + M.\,Exploitability), 10]) \times ExploitCodeMaturity \times RemediationLevel \times ReportConfidence)$$

– Modified Exploitability Sub Score

$$8.22 \times M.\,AttackVector \times M.\,AttackComplexity \times M.\,PrivilegeRequired \times M.\,UserInteraction$$

2.2.8 Comparison Between Products

All these platforms have the same goals: to provide the costumers with a platform to monitor all their networks and to warn and give them feedback so they can take action in a more informed way.
We are only going to use these platforms as examples of implementations in order to decide the best way to build our own. The difference between our platform and others is that ours will not be an open platform. It will be a more strict platform because it will be working within the University's firewall. The average person will only have access to some statistics about the domains' performance.

We will use a narrower risk score (1 to 5) and it will be calculated (based on the CVEs found) the same way it is calculated in Shodan and Spyse. Unlike all the platforms above mentioned we will have a notification system that provides a possible solution to the vulnerabilities found.

In our platform we will have a continuous scan of all domains but it is possible to request an immediate scan to a certain domain.

2.3 Technology

2.3.1 Python

Python is a versatile interpreted high-level programming language. Its emphasizes code readability using significant indentation. Its constructs and object-oriented approach help programmers write clear, logical code for small and large-scale projects. By supporting multiple programming paradigms, including structured (particularly, procedural), object-oriented and functional programming it became an obvious choice. The group chose **Python** because of its versatility, easy learning curve, ample collection of modules and the group's familiarity with the language. It will be mainly used for processing all the information given by the analysis. The processing of such data will make use of the cryptography modules, the flask module and other modules available in python.

2.3.2 SQL

SQL, otherwise known as Structured Query Language, is a language frequently used for building and managing databases. Due to it's relational nature it adapts fairly well to the project's needs. We will be using SQL, as the group is quite familiar with the language and the data organization trough tables is very easy to understand and illustrate, hence it becomes easier to implement. We also chose the SQLite module due to it's easy use and the fact that the database is stored in one or more local files, the database will be encrypted using the cryptography module referred above.

2.3.3 HTML/CSS/JavaScript

HTML/CSS/JavaScript will be used for our Website interface. The platform's front-end will be developed using these languages. We are mainly going to use templates to do this faster as the only goal is to make the platform look more professional. The HTML will be implemented using flask so it can be dynamic.

2.4 Tools

2.4.1 Nmap

Nmap [7] [8] started as a tool for scanning and testing network connectivity between different hosts and services such as firewalls, router ACLs and other factors that can impact a network based connection.
More modern versions of **Nmap** include a built-in scripting language **NSE** that gives it the capability of doing network vulnerability tests. It became a fast lightweight vulnerability scanner.
We will use this tool to analyse and give a CVSS to each IP of the IP pool given by the DNS dumpster.

2.4.2 Vulscan

Vulscan is a script made to run in the Nmap program. It has implemented into it a series of .csv files,that work like databases, which have different maners of displaying the vulnarabilities found. We went to the CVE database website and downloaded a file which is going to be our database for the vulscan script. It displays some information we need such as the CVE vectors which will be used to find more details about the vulnerabilities found in the domains in question.

2.4.3 DNS Dumpster Server

The DNS Dumpster is a tool used for getting the subdomains of a certain domain. This tool uses open source intelligence resources to query for related domain data. The search relies on data from the crawls of the Alexa top 1 millions, Search Engines, Common Crawl, Certificate Transparency, Max Mind, Team Cymru and Shodan. It´s a actionable resource for both attackers and defenders.
We'll use this tool to obtain all the subdomains and web hosts.

2.4.4 Docker

We'll use a container with Docker [9] [10] [11] to make our program run properly. It will make it easier to implement the final program into our platform. A container is a way to package application with all the necessary dependencies and configurations so the application runs quickly and reliably from one computer to another. A docker container image is a lightweight, executable package of software that includes everything needed to run an application. This tool is available for both Linux and Windows-based applications. Containerized software will always run the same, regardless of the infrastructure.

2.5 Current Research Directions

By researching different website and platform architectures we saw a rising worry about the safety of the content and the private information of users but sometimes it is easier said than done. We should always be more careful about certain services. We should always do more profound research about a websites that we want to put our private information in. Sometimes the software is outdated and there are a lot of flaws that can lead to unknowingly expose our personal information.

We came to see instances of system software that runs without protection constraints. A better running time of the application, misunderstood design and inadequate hardware support are some of the many reasons this occurs.

A recent study suggested that the average cost of a data breach in Canada is of about $6.5 million per incident. With all the available tools at a hacker's disposal nowadays, its easier than ever to get into a database with valuable information. We have to be in constant alert and analysis to provide the safest tools to ensure data safety.

2.6 Concluding Remarks

Ever since the rise of computers and the internet, it is more important to start seeing and understanding all kinds of flaw exploitation. The field of cybersecurity has been in a constant battle of bettering safety protocols and making it harder and harder for hackers to get access to sensitive information. But as security improves, so do the tools that hackers use. There is always a flaw in the code or some software that is not updated. Any problem related to the architecture of a program can be a reason to note that a certain service must be updated. Because there are a lot of services to be analysed and everyday more are created, it is necessary to have a tool that can warn us of those vulnerabilities and lead us to have a better understanding of the necessary protection domains need so that we don't end up with major leaks and the integrity of a service compromised.

Chapter 3

System Requirements & Architecture

3.1 System Requirements

3.1.1 Actors

- **Admin -** entity with most authority on the system, has access to the entire system and features. Has the permission to see the list of all machines, users and vulnerabilities detected. Can add and remove users from a machine and accept and decline solicitations from the users.
- **User -** it only has access to its own machines, where it can verify statics and vulnerabilities found. He can receive notification messages about new vulnerabilities from the assigned machines. He can also request for a subscription to a different machine that can either be accept or decline by an admin. The rest of the machines can only check global statistics.
- **Guest -** this is going to be a random person accessing the platform. Because of the unknown origins of the individual we have to reduce the information shown so the security and the privacy of the machines won't be violated. For that the Guest can only check the global statistics about the system.

3.1.2 Use Cases

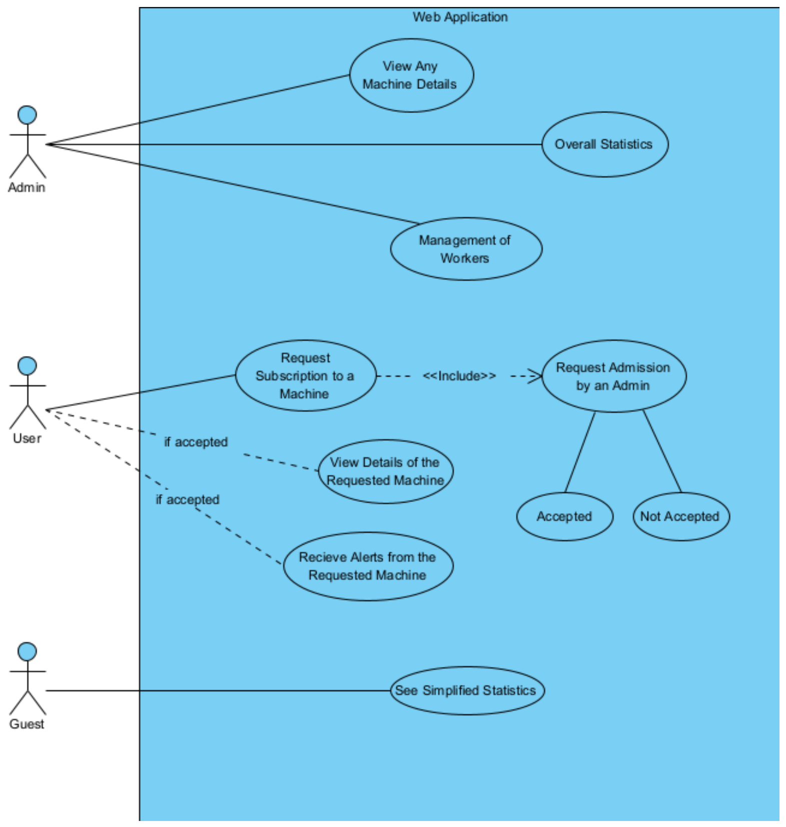

Web Application
Admin
View Any Machine Details
Overall Statistics
Management of Workers
User
Request Subscription to a Machine
<<Include>>
Request Admission by an Admin
if accepted
View Details of the Requested Machine
if accepted
Recieve Alerts from the Requested Machine
Accepted
Not Accepted
Guest
See Simplified Statistics

Figure 3.1: *Use Cases*

The figure above represents the use case model of the platform and it has three main user types. As we can see, the **Admin** as access to the machines and their statistics and can also assign new **Users** to a machine they requested which will give them access and power to see the statistics and "protect" that machine. Finally the **Guest**, because, its from unknown origins can only see how the university is doing only by some very simplified statistics like the number of vulnerabilities in all domains, not giving them any information that can be exploited.

- **View Any Machine Details -** The person has access to all machines in the network that are being analysed by the platform. Has the ability to see all the information available from all the machines.
- **Overall Statistics -** Can see all the information gathered from all the machines or only the machines chosen by the person who has this kind of access. Will be notified every time a vulnerability is spotted in one of the machines in the network.
- **Management of Workers -** The power to admit and see how the users are progressing with their machines. Receives solicitations from the users so they can be assigned to a

machines and can also block any user who's not doing a proper maintenance of their machine.

- **Request Subscription to a Machine -** As a user you will have to ask permission from an admin to be assigned to a certain machine. He'll have to specify why he wants to have access to that machine so the admins can do a proper evaluation and give the user access.
- **Request Admission by an Admin -** The requested subscription will be put under evaluation to determine whether the user is accepted or not.
- **View Details of the Requested Machine -** If the admin accepts the request to a machine, the user will have full access to the machine details and the machine statistics over time. He will be able to operate on that machine. Has admin like permissions to that specific machine.
- **Receive Notifications of the Requested Machine -** Admin like notifications will be received only from the machine assigned to the user.
- **See Simplified General Statistics -** Has access to a simplified dashboard that only shows the number of vulnerabilities discovered in all machines and also the vulnerabilities that have already been dealt with.

3.1.3 Non-functional Requirements

- **Portability -** The platform is intended to scan all the domains of UA despite the OS of the machine running it. So it should be able to run in every OS (windows, linux, ...). The users will access to the information via website through network connection. Saying so, the platform should be accessible by different web browsers, operating the same way in each of them.
  Priority: Low
- **Usability -** The management interface must be intuitive following the latest User Experience and User Interaction principles in order to provide a good experience to every single user. Even though this platform is designed to people with a technological background, everything that occurs should be of easy comprehension.
  Priority: Medium
- **Reliability -** The fact that this system works (or may work) with sensitive data gives it higher responsibility to treat that information. It is mandatory to have error-handling treated carefully in order to prevent malfunctions in the system.
  Priority: High
- **Privacy -** The system must not store any sensitive or personal data about the hosts, even if it is retrieved by any of the scrapping tools during the scans.
  Priority: High
- **Security -** Despite the possibility to obtain the information collected by anyone who uses a scrapping tool in the same network this system operates, only people with authorized subscription, ownership or administration permissions should be able to see this

data.
Priority: High

3.1.4 Functional Requirements

- **DashBoard -** The main point of interaction between the user and the system is the web platform. Therefore, the web platform's dashboard should allow for system management and it should display any information related to vulnerability monitoring that the user has access to. It is very important that the web platform is accessible 24 hours a day. Another key aspect of the web platform is that it uses UA IDP access for user authentication.
  Priority: High

- **Vulnerability Analysis -** The information displayed on the web platform's dashboard is obtained through vulnerability scans. All scanned domains will also have a CVSS score attributed to it. This score intelligently ranks the risk that the domain is in.
  Priority: High

- **Scrapping -** Our system will make use of some external plugable tools. DNSdumpster is a domain research tool that will be used to obtain the IP pool of the campus. Among the tools used is also Nmap, which will be used for vulnerability scanning. Docker will also be used to allow for software containerization.
  Priority: High

- **Alerts -** The notification system will be implemented to allow the user to be updated through e-mail. This should help the user stay more aware of the state of the machine he's assigned to.
  Priority: High

3.2 System Architecture

3.2.1 Domain model

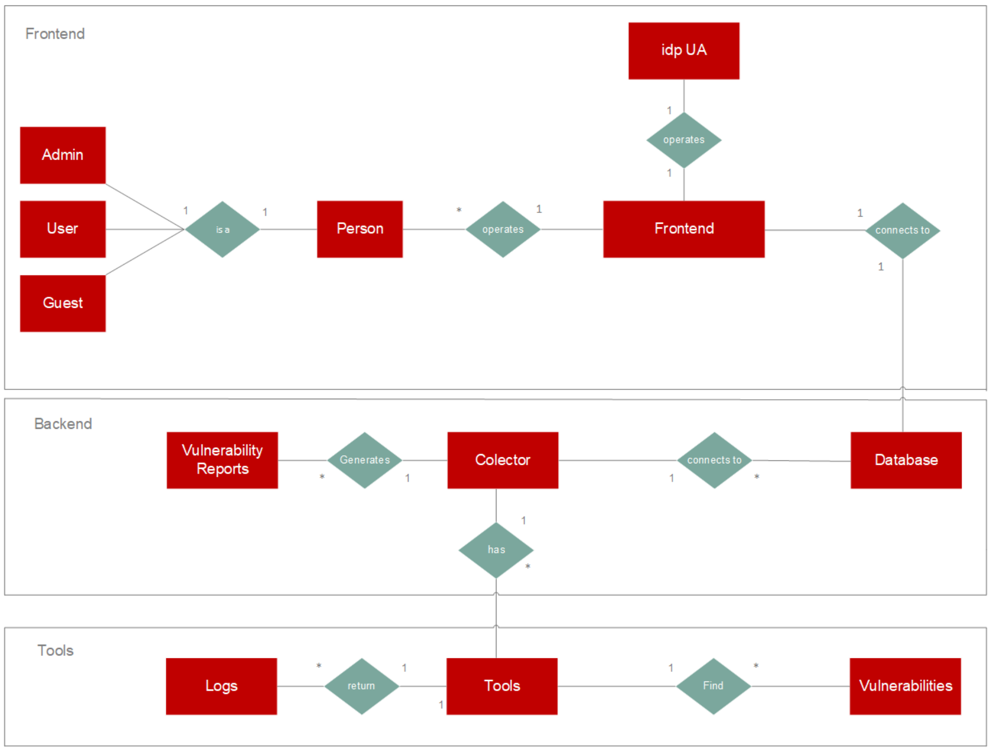


Figure 3.2: *Domain Model*

The domain model presented on Figure 3.2 describes the various entities, roles and relationships of the system used for the application.
To begin with, there is the Person entity, which can be three different roles: Admin, User and Guest (see Figure 3.1) that can operate the Frontend. The Database will store CVEs, login information and University domains. The Collector will use tools like nmap discussed before to detect CVEs in domains and update the database with its findings.

3.2.2 Database model

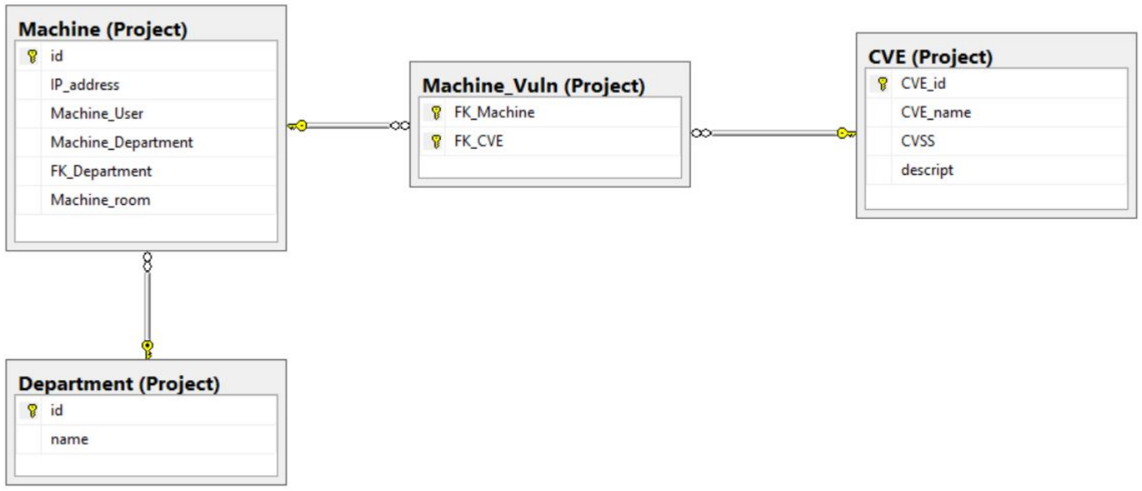


Figure 3.3: *Database Model*

In the Database, we need a way so store known CVEs, the Univesity machines we want to scan and an easy way to connect the two. For that, a many-to-many relationship between those entities (seen in Table "Machine Vuln", Figure 3.3) offers a way to store and access all CVEs detected in a machine as well as all machines that have found to have a specific CVE.

A department table was also created in order to find where a machine is located on the campus grounds. A department can and most likely will have more than one machine, therefore creating a one to many relation.

Bibliography

[1] Shodan. https://www.shodan.io.

[2] Binary edge. https://www.binaryedge.io/.

[3] Ivre. https://ivre.rocks/.

[4] Spyse. https://spyse.com.

[5] Common vulnerability scoring system v3.0: Specification document. https://www.first.org/cvss/v3.0/specification-document, 2020.

[6] What is the osi model? https://www.forcepoint.com/cyber-edu/osi-model.

[7] Nmap.org. https://nmap.org.

[8] Running a quick nmap scan to inventory my network. https://www.redhat.com/sysadmin/quick-nmap-inventory, 2020.

[9] Docker. https://www.docker.com.

[10] What is docker? https://www.oracle.com/pt/cloud-native/container-registry/what-is-docker.

[11] Docker. https://www.redhat.com/pt-br/topics/containers/what-is-docker, 2018.