

**BABEȘ-BOLYAI UNIVERSITY CLUJ-NAPOCA**  
**FACULTY OF MATHEMATICS AND COMPUTER**  
**SCIENCE**  
**SPECIALIZATION COMPUTER SCIENCE IN**  
**ENGLISH**

## **DIPLOMA THESIS**

**Port Scanner with integrated Database**

**Supervisor**  
**Conf. Phd. Suciu Mihai**

*Author*  
*Ardelean Tudor*

2021

**UNIVERSITATEA BABEȘ-BOLYAI CLUJ-NAPOCA  
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ  
SPECIALIZAREA INFORMATICĂ ÎN LIMBA  
ENGLEZĂ**

## **LUCRARE DE LICENȚĂ**

**Scanner de porturi de rețea cu bază de  
date integrată**

**Conducător științific  
Conf. Dr. Suciu Mihai**

*Absolvent  
Ardelean Tudor*

2021



---

## ABSTRACT

---

As it is commonly believed, network architecture in our time is continuously improving and advancing in complexity so as we develop newer technologies and applications, we must also turn our attention towards security solutions that help us protect what we build. For that part we are in need of scanning solutions to can help us identify known vulnerabilities in those systems.

My goal is to improve my knowledge and understanding of this topic by developing a network port scanner that is capable of scanning and associating discovered open ports with it's known vulnerabilities using an integrated database to speed up the reconnaissance process. Doing so, a tester can immediately begin to start creating the attack vectors a real attacker could use.

The design of the application aims for a beginner friendly and easy to use tool that can also benefit a more experienced user. Moreover, a lot of similar tools lack the user-friendliness that the app aims to provide, so inexperienced users will find it easy to improve their skills by using it.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	Preview . . . . .	1
1.2	Motivation and Goals . . . . .	2
1.3	My contribution . . . . .	2
1.4	Thesis overview . . . . .	3
<b>2</b>	<b>Networking and Vulnerablity Essentials</b>	<b>4</b>
2.1	Networking . . . . .	4
2.1.1	Introduction to networking . . . . .	5
2.1.2	OSI Model . . . . .	5
2.1.3	TCP/IP Model . . . . .	8
2.1.4	Commonly used Ports . . . . .	13
2.2	Vulnerabilities . . . . .	14
2.2.1	Introduction to vulnerabilities . . . . .	14
2.2.2	Common Vulnerabilities . . . . .	15
2.2.3	Common cyber attacks . . . . .	17
2.2.4	Auditing . . . . .	19
<b>3</b>	<b>Related work</b>	<b>20</b>
3.1	Nmap . . . . .	20
3.2	Masscan . . . . .	21
<b>4</b>	<b>Application Structure and Design</b>	<b>22</b>
4.1	Used Technologies . . . . .	22
4.1.1	Programming Language . . . . .	22
4.1.2	Libraries . . . . .	22
4.1.3	Database . . . . .	23
4.1.4	Graphical User Interface . . . . .	23
4.2	Software architecture . . . . .	24
4.2.1	Description . . . . .	24
4.2.2	Application layers . . . . .	24

4.3	Interface design . . . . .	30
4.4	Testing . . . . .	34
4.5	User Manual . . . . .	36
<b>5</b>	<b>Conclusions and Future Work</b>	<b>37</b>
5.1	Conclusions . . . . .	37
5.2	Future Work . . . . .	38
	<b>Bibliography</b>	<b>39</b>

# Chapter 1

## Introduction

### 1.1 Preview

Nowadays, network architectures are getting more and more complex, which is a great thing for both cyber security awareness enthusiasts and product owners alike, but it doesn't mean that even if the challenge to compromise them is greater, they are impregnable.

Big tech companies and individuals alike fall prey to cybercrime and its perpetrators on a daily basis, thing that shows us that to this day, cyber security is still taken lightly by some or even ignored completely by others. In most cases, the weakest link in the chain is actually the uninformed employee that gets hit from inside the network, like in phishing scams, and that may lead to a complete takeover.

As the number of victims rise with each big or small hit, so does the need of investing in further research and solutions for protecting what is our most important online possession, our identity and, as mentioned, data theft is still a huge problem even to the wealthiest and sturdiest organisations.

Even if it might be impossible to completely eradicate data theft and cybercrime, it doesn't mean that we don't have to try and mitigate known problems and vulnerabilities so that we can create a safe and cyber security aware space around our personal information.

## 1.2 Motivation and Goals

I had a lot of passion for this field of study from the very beginning i started pursuing this bachelor's degree so i devoted a lot of time to gather the necessary knowledge to begin working at a decent company on a cyber security engineer position. My time there was well spent by learning and practicing all sorts of techniques used today by engineers and penetration testers. I even got the chance to witness a large scale cyber attack and help in fixing the damage it has done to people after realising how serious such an event can be and how it can affect such a large corporation.

I have a strong belief that there is a large demand of security solutions and products, as well as further research into the domain and for that, more people need to realise how big of a difference it can make to have a "bulletproof" security system. From my point of view, this can be achieved in two ways. The first is through breaches and data theft, the hard way, and the other is through consistent learning and staying up to date with best practices.

My goal is to raise the cyber security awareness by providing people with the means to start learning how to use a basic port scanner and how to identify and mend known vulnerabilities that can be found on a system. I really hope that, by doing so, i help them in studying this domain and i also hope to better myself by gathering more knowledge of this domain.

## 1.3 My contribution

I know there it takes a lot of work that needs to be done and a lot of time to be put in to implement a sophisticated port scanner, but, i plan to start my journey here and gradually work on creating one.

Knowing all that, i want to bring something new to the table, something that can help a new user identify vulnerabilities more easily. In the beginning, when i used the other port scanners, i found myself lost when the result came after the process of fingerprinting the services running on the specified ports because i didn't know if what i found was good or bad. I want to create something that can tell you if the version of the application running on the specific port has a problem. For example, if a version with known vulnerabilities is found, then the user should know it can and must be updated because that version is most likely vulnerable.



## 1.4 Thesis overview

I intend to structure my thesis into 5 different chapters in which i will talk about various things, from theoretical notions, to technical features and application design. They will be structured as followed:

- Introduction

This chapter offers an introductory presentation of the thesis.

- Networking and Vulnerability Essentials -

The networking and vulnerability chapter is a theoretical chapter that offers information on networks and vulnerabilities based on my studies from several books. It is divided in two main parts, each talking about my studies on the respective topic.

- Related work

In order to develop an application, one must look at other possible variants and, in this case, there are quite a few options but i chose to talk about two of the most important ones available on the market.

- Application structure and design

This technical chapter is outlining the work i have done on the application itself. It's main points are the technologies used and how they are linked together to create the final result. It also contains details about the user interface and testing.

- Conclusions and Future work

This chapter contains my final thoughts on the study and new ideas on how i can improve my knowledge and understanding of the domain and how i can improve the application.

# Chapter 2

## Networking and Vulnerability Essentials

In theory, a network is a collection of computers connected by digital interconnections that are sharing resources located on or provided by network nodes using a set of standard communication protocols. The interconnections between the specified nodes are made up of a number of telecommunication network technologies, including physically wired, optical, and wireless radio-frequency techniques that may be configured in various network topologies.

Although complex in nature, in the next sections i am going to briefly describe the studies i have done in order to get my knowledge pool expanded, knowledge about networking and the importance of keeping them secure.

### 2.1 Networking

“The internet is not something you just dump something on. It’s not a truck. It’s a series of tubes.”

-United States Senator, Ted Stevens(Alaska)

Although widely ridiculed because of the limited approach, the senator’s words against a proposed amendment opposing network neutrality offer us the most basic definition of them all and good starting point, that is, if we can connect the “tubes”.

### **2.1.1 Introduction to networking**

In the course of auditing a service, a security audit practitioner will perform different techniques to assert the situation, techniques like code-reviews that examine low and high level network traffic, a process that requires basic knowledge of networking so developing such a skill is trivial if one wants to advance in this domain.

The bulk of network-aware software makes use of the TCP/IP protocol stack's features via high-level interfaces like BSD sockets or frameworks like Distributed Component Object Model (DCOM), a software component that, actually, is now, obsolete. However, other applications must deal with network data at a lower level, in an materialistically unseen world of segments, frames, packets, fragments, and checksums.

Because networking is so common nowadays, it is sometimes taken for granted. Many applications rely on networking, such as email, the Internet, and instant messaging. Although each of these apps employs a different network protocol, they all put to good use the same basic network transport mechanisms that are well designed to do most of the hard work.

### **2.1.2 OSI Model**

"When two computers talk to each other, they need to speak the same language. The structure of this language is described in layers by the OSI model."(Page 196)[Eri08]

By definition, the Open Systems Interconnection model (OSI model) is a conceptual model that characterizes and standardizes a telecommunications or computing system's communication operations regardless of its underlying internal structure and technology. Its purpose is to ensure that different communication systems can communicate with each other using standard communication protocols.

The OSI model started being developed in the late 1970s to promote and diverse networking methods that were trying to accommodate the needs of large international networking. It was defined in a raw form in Washington, DC in February 1978 by Hubert Zimmerman, a pioneer of computer networking. The publication was made in 1980 being followed by Zimmerman's publication in April 1980: "OSI Reference Model — The ISO Model of Architecture for Open Systems Interconnection".

The representation of the OSI Model is presented in Figure 2.1.

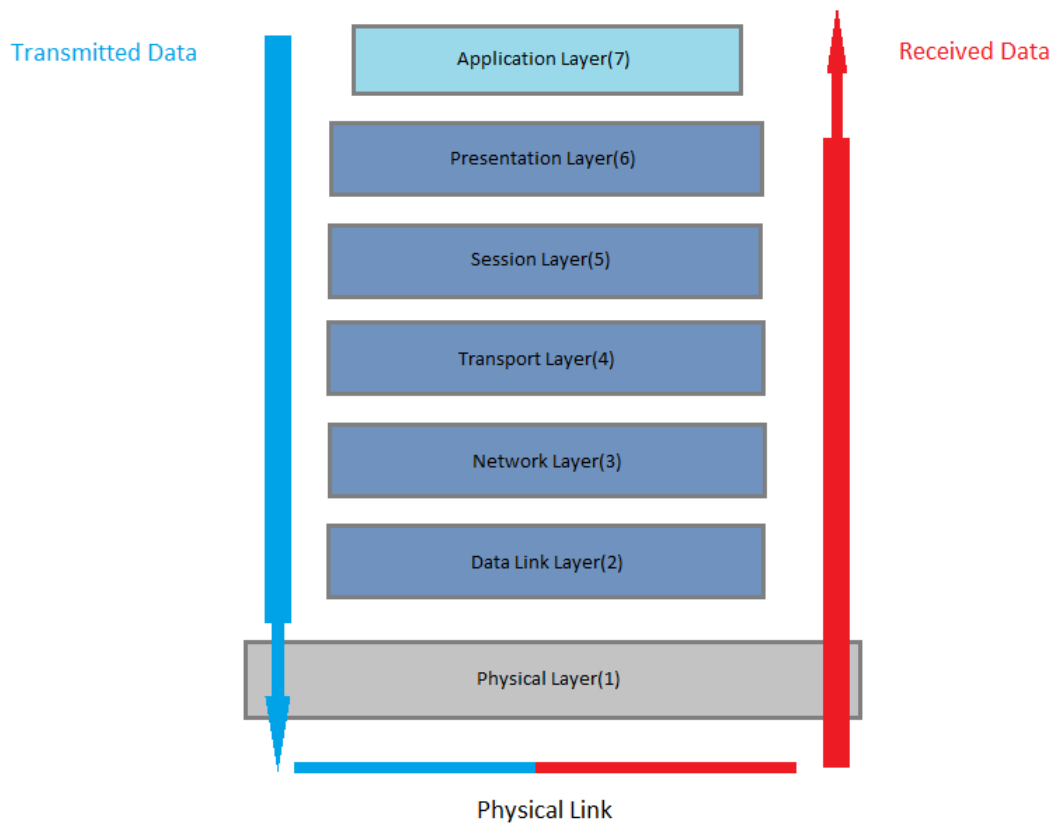


Figure 2.1: OSI Model representation drawing

The OSI model is divided into layers. This allows routing and firewall devices to focus on passing data at the lower layers while disregarding the data encapsulation levels needed by active applications, this ensures a faster communication speed for the packets.

The seven layers of the OSI model:

- Physical layer

The lowest layer, the physical layer is used to deal with the physical connections between two targets, it communicates raw bit streams and is also in charge of activating/deactivating and maintaining this stream communications.

- Data-link layer

The data-link layer is in charge of transferring the actual data between two points but the main difference between it and the physical layer is that this layer can handle high-level functions such as error correction and flow control, providing the ability to activate/deactivate and maintain data-link connections.

- Network layer

The network layer provides a middle ground. It works as a mediator between the lower layers and higher layers, providing routing and addressing.

- Transport layer

The transport layer provides transparent data transfer between systems by allowing reliable data communication, by doing that, it also provides the higher layers with reliability and insurance of effectiveness of data transmission.

- Session layer

The session layer is providing effective and continuous establishment and maintenance of connections(sessions) between network applications.

- Presentation layer

The presentation layer acts as a translator for the applications so that they can get the message and understand it, allowing usage of techniques like encryption/decryption/compression to be implemented and used.

- Application layer

The application layer is in charge of keeping the records of the details and requirements of the application.

How are these layered used?

When information is sent through these layers they follow a certain protocol and are communicated in small bits called packets. The packets are designed to accommodate these protocols as they are wrapped layers and layers, starting from the application layer. From there on, there is a wrap for each layer stacked onto each other. The mentioned process is called encapsulation and, as stated in Jon Erickson's *Hacking: The Art of Exploitation* [Eri08], it behaves just like an onion because each layer has a header and a body. While the header provides protocol information for the respective layer, the body has the entire package that it is left. Each OSI Model layer peels away another skin until the core reaches the lower layers.

As i read through Jon Erickson's book[Eri08], i was presented with a very good example that helped me materialise all the new information i got from the subject and i will try to recreate that experience.

The example talks about a real-world experience on how this layers work such as the physical layer could very much be and is in most cases the ethernet cable and the network card that take care of transmitting the data. The data-link layer is the ethernet itself as it provides low level communication between the LAN ports without having IP addresses yet. The network layer provides us with the concept of IP addresses as it is responsible of it and it also moves data from one address to the other. The transport layer enables the use of TCP, providing socket connections. From then on the router allows us to not use the session and presentation layer as we surf the web and get results through HTTP, the final layer.

### **2.1.3 TCP/IP Model**

TCP/IP allows us to specify how a particular computer should connect to the internet and how data should be sent between it and the internet. When a lot of computer networks are linked together, it aids in the creation of a virtual network. TCP/IP was created in the 1970s and accepted by ARPANET in 1983 as the standard protocol.

#### **Internet Protocol**

The Internet Protocol(IP) is, by definition, the main communications protocol used today on the wide web. It allows routing and chaining together the links that establish the internet. It is tasked with delivering packets from source to destination using headers and encapsulation techniques.

It's behaviour is described very good in "The Art of Software Security Assessment" by Justin Schuh Mark Dowd and John McDonald [MD06], the host performs this processing immediately after receiving a packet and decides how to handle it, whether that means passing it to a higher-level protocol handler in the network stack (such as TCP or UDP), signaling an error because the packet cannot be processed, or blocking the packet because it fails to meet the criteria of a firewall or other similar data inspection tools.

The primary IP protocol is Internet Protocol Version 4 (IPv4), which is the first major version. It was succeeded by Internet Protocol Version 6 (IPv6), which has been widely used on the public Internet since around 2006, but, as it's getting older and older, it had began to develop some problems. IPV4 headers are illustrated in Figure 2.2.

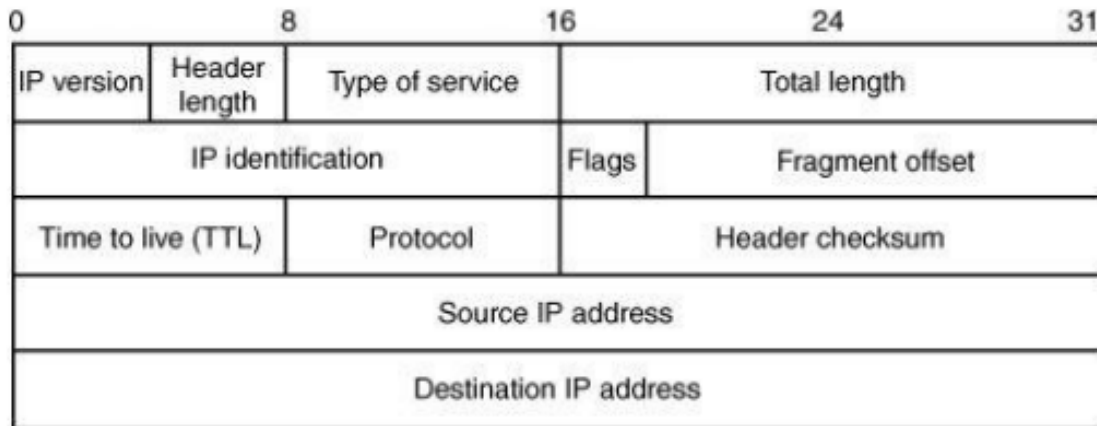


Figure 2.2: IPV4 Header diagram [MD06]

## TCP

The Transmission Control Protocol (TCP) is a transport-layer protocol built on top of the Internet Protocol. Based on the idea of connections, it's a system for assuring dependable and precise data transport from one host to another. A connection is a bidirectional communication channel between two hosts that allows one program to communicate with the other. Exchanging specific TCP packets makes establishing and closing connections possible.

As stated in "The Art of Software Security Assessment" [MD06] TCP data crossing the connection is seen by the endpoints as streams: ordered sequences of contiguous 8-bit bytes of data. The TCP stack is in charge of dividing this data down into packet-sized chunks called segments. It's also in charge of ensuring that the data is correctly sent. As it is a stream based connection protocol, the data is treated as an uninterrupted stream which is monitored internally using sequence numbers to represent parts of the data that is being transmitted.

The connection is established by a mutual agreement between the client and the server called the three way handshake. It follows thorough steps to perform connection operations like open, close and reset.

The three basic operations run like this:

- Opening a connection
  - The server is in a state called Listen, which means it has a port that waits for new connections.
  - The client sends a SYN packet to the listening port wanting to establish a connection.
  - The server receives the packet and enters the SYN RCVD state.
  - The server sends a SYN ACK packet because it acknowledges the clients SYN so it needs to provide one of it's own.
  - The client receives the packet and sends again a packet, this time an ACK packet to show that it acknowledges the connection too.
  - The last step is the server receiving the ACK packet and establishing the connection as the handshake has been made.
- Closing a connection
  - The client sends a FIN ACK packet signaling that he wants to close the connection then waits for the server.
  - The server receives the packet and acknowledges the request.
  - The client receives the acknowledgement.
  - The server now closes the connection and sends a FIN packet.
  - The client receives the packet and acknowledges it.
  - The server receives the acknowledgement.
  - Finally, the client closes the connection.

- Resetting a connection

The connection reset can happen when an error occurs on either sides, during any process of interaction between the client and the server and it is done by the server sending a packet with the RST flag set.

It usually happens if:

- The server doesn't have it's port listening(open) and the client is trying to connect to it.
- The connection packet arrives without a SYN flag so the handshake cannot be processed and acknowledged by the server so it tries to reset it.



The TCP Header components can be found in Figure 2.3

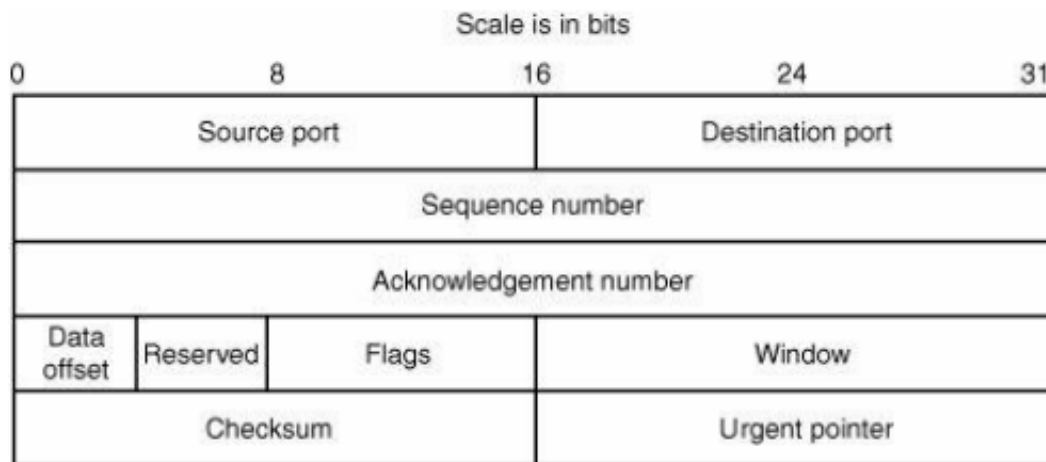


Figure 2.3: TCP Header diagram [MD06]

Here are some basic details about each of them:

- Source Port
  - it indicates the TCP source port for the connection, where the data is sent to.
- Destination Port
  - it indicates the TCP destination port for the connection, where the data arrives.
- Sequence Number
  - it indicates where the data should go when to be rebuilt correctly from the incoming stream.
- Acknowledgement number
  - this is the sequence number expected from the client so if the data is lost, it can be pinpointed and it can be resent.
- Data Offset
  - this indicates the size of the TCP header.
- Reserved
  - empty field.

- Flags
  - this is where the flags are set so the communication between the client and the server is clear and productive.
- Window
  - this indicates how much data can be sent down the stream at a time.
- Checksum
  - the checksum is a verification parameter that gets computed from several fields combined and the ip addresses of both peers.
- Urgent pointer
  - this is used as a pointer to urgent data.

As good as the protocol is, attackers can find ways to exploit it by performing attacks like a SYN ACK attack [Sea19].

## UDP

UDP (User Datagram Protocol) is, by definition, a communications protocol that is largely used on the internet to build low-latency, loss-tolerant connections between peers. It speeds up transmissions by allowing data to be transferred before the receiving party has agreed, a method that is considered very unsafe from a security point of view and has a limited amount of applications.

To put more stress on the matter, exemplified in the “The Art of Software Security Assessment”[MD06], the authors indicate that stateless firewalls have a difficulty with UDP connections. A connection initiation is represented by a specific packet in TCP: the SYN packet. However, there is no such packet in UDP. When administrators try to pass the DNS protocol over the firewall, this problem generally arises.

As quick as it may be, it indeed comes with a multitude of problems and vulnerabilities like Layering issues, Spoofing attacks and the lack of reliability which comes from the fact that, if a packet is lost, there is no way of knowing it happened or what is missing, unlike its counter variant, TCP.

On another note of comparing the two protocols, the UDP connection is much simpler. The header is composed of the source and destination ports, length, checksum and the data itself. If the TCP is like a phone call, then UDP is like sending a text message on an old phone.

Nowadays it finds its use in numerous key applications like the DNS, SNMP, RIP, DHCP and even some VPN services.

### 2.1.4 Commonly used Ports

Ports allow our systems to communicate with different services and there are some that are standardized by the Internet Assigned Numbers Authority or widely used by the users.

Here is a list with the most common ones that need to be known as they can be easily fingerprinted:

Port	Name	Description
5	rje	Remote job entry
7	echo	Echo Service
9	discard	Null service for connection testing
11	systat	System status service
13	daytime	Date and Time provider
17	qotd	Quote of the day
18	msh	Message send protocol
20	ftp-data	File transfer protocol data port
21	ftp	ftp port, sometimes used as fsp(file service protocol)
22	ssh	Secure Shell
23	telnet	Telnet Service
25	smtp	Simple mail transfer protocol
37	time	Time protocol
42	nameserver	Internet Name Service
69	tftp	Trivial file transfer protocol
80	http	HyperText Transfer Protocol
161	snmp	Simple Network Management Protocol
194	irc	Internet relay chat
389	ldap	Lightweight directory access protocol
443	https	Secure HyperText Transfer Protocol

From personal experience, it is worth knowing about them and having them close by as they may come in handy when you are trying to perform an audit.

## 2.2 Vulnerabilities

“Understanding hacking techniques is often difficult, since it requires both breadth and depth of knowledge.”

-John Erickson, *Hacking: The Art of Exploitation*[Eri08]

As John Erickson described in his book, most people equate hacking with violating the law, and they believe that everyone who hacks is a criminal. Granted, some people utilize hacking skills to disobey the law, but that isn't what hacking is all about, as much bad as it can do, it can also help us understand how to protect ourselves better.

### 2.2.1 Introduction to vulnerabilities

Since the discovery of the computer, internet and how they all can be linked together, hackers or people that were interested in that side of the coin appeared and they have been actively affecting the course of all things related to software and hardware advancements. As much damage one think they have done, they are also problem solvers and through harsh but maybe necessary methods to be one step ahead of the developers, they also keep them on guard as new defensive technologies continue to be researched and developed to counter the cyber threat.

By looking into our past and reading studies about the matter, i found out that times change but some questions remain. For example, in “*Hacking: The Art of Exploitation*”, John Erickson rises a question, how does one distinguish a good hacker from a bad one. For that, we must acknowledge the existence of good hackers, the people that help us stay protected and secure, protected from the real threat that is lurking.

Back in the day, “Hacker” was the term for the good guys. Now, we even have an existing hierarchy amongst them. Some are penetration testers, some are participating in red/blue team activities and others are freelancers that do their own thing, matter that shows us how, even in this field, evolution began and prospered.

### **2.2.2 Common Vulnerabilities**

A common starting point in learning about the potential and scale of cyber attacks and how we can prevent them is to start researching the most common vulnerabilities from both the perspectives of an attacker and a developer alike.

The Open Web Application Security Project® ([Fou21]), a nonprofit foundation that works to improve the security of software through community-led open-source software projects developed a Top10 most common and dangerous web application security risks and i believe it is on point as i have studied it myself and found the information to be very helpful.

Here are the items they put on that list briefly explained:

- **Injection**

There are a lot of injection flaws that get exploited on a daily basis and the most common are the database(SQL,NoSQL), OS and the ldap. These flaws get exploited by injecting malicious data or data through all kinds of means into the main application, disguised as a line of code or a command. The main system can get tricked into doing what it shouldn't do and spilling out information or compromising the system itself. This can be fixed with prevention by using common methods like data sanitisation and in general a more secure way of coding.

- **Broken Authentication**

This generally happens when a safe authentication method is not implemented by the developers and it can allow the attackers to gain access to accounts they do not possess and, by that, they gain access to personal information which is a very serious matter. This can be fixed by implementing the standards for a safe authentication and by not allowing the users to voluntarily put themselves in the danger of exposure.

- **Sensitive Data Exposure**

This issue is characterized by outsiders having access to application internal data which can lead to more serious matters like fingerprinting the system and accessing data that should be encrypted but it's not leading to sensitive data being leaked. This can be prevented by using standard encryption and data protection protocols.

- XML External Entities

Vulnerable XML processors can be exploited by uploading malicious XML into the system and, by doing that, exploiting vulnerable code which can lead to a compromised system. This can be fixed by avoiding serialization of sensitive data and keeping the XML processors up to date.

- Broken Access Control

Not setting boundaries to an application's users may lead to serious exploits like privilege escalation. This usually happens by bypassing the control checks using data manipulation and modifying requests. It can be resolved through setting up defenses against things like that, like denying any access to something a user does not have ownership to.

- Security Misconfiguration

This is the most common problem amongst web applications and it usually occurs when the system's configuration is incomplete, the default configuration was not changed or using well known configurations. This can be fixed by seeing that a system is configured all the way through with no errors and keeping it patched and updated.

- XSS - Cross-Site Scripting

This vulnerability allows users to change a web application's structure by executing scripts through HTML and Javascript leading to attackers executing malicious code directly in the victims' browser. This can also be resolved by input sanitisation and using frameworks that automatically escape XSS by design.

- Insecure Deserialization

This are logic attacks against an application that can lead to remote code execution and data tampering and it can affect a wide range of technologies, from databases to HTTP cookies. We can prevent this by using integrity checks and monitoring the data that comes through the network.

- Using components with known vulnerabilities

Using components with known vulnerabilities is a vulnerability itself because of the developers' refusal to acknowledge that their system might be compromised. This can be achieved by using outdated frameworks, libraries or other software modules. Of course it can be fixed by making sure the latest patch is running or we are willing to accept the risk.

- Insufficient Logging and Monitoring

Missing logs and evidence of attacks that happened of the system can only make the attackers persist and maintain their position. We must know what got exploited so we can fix it and deny the attackers the opportunity of going even deeper.

- Personal Selection: Cross-Site Request Forgery(CRSF)

This attack is an exploit that forces an end user to execute unwanted code on a web application that can lead to, for example, transferring data or funds without the victim's approval. This can be prevented by implementing multi-step verification on transactions and data request.

There are a lot of ways of exploiting this vulnerabilities and i found that The Web Application Hacker's Handbook[DS11] offers the exact steps on how to reproduce them.

### 2.2.3 Common cyber attacks

As we talked about the common vulnerabilities, it is only natural to talk on a larger scale too. Large scale cyber attacks are happening very frequently without the large mass of people even knowing so here are some examples of such kinds of threats.

#### Malware

Malicious code is very dangerous, but what happens when whole applications are designed to do just that, compromise or break, not just a few lines of code injected into a web application. Malware is the shortened term for a variety of harmful software types such as viruses, ransomware, and spyware. Malware, short for malicious software, is a type of code created by cyber criminals with the intent of causing substantial harm to data and systems or gaining unauthorized access to a network. The Shellcoder's Handbook [CA07] offers a very in depth analysis of the power of what a harmful software can do.

There are a lot of types of malware that get spread through networks and here are some of them explained:

- Worms

Worms are designed to spread from system to system and do damage. They can do a various of things from stealing data, overloading your system by replicating itself to modifying or deleting your personal data.

- Viruses

Viruses are, unlike worms, stealthier and in need of an already damaged system so they can spread through the network. It can use your own persona to infect others.

- Ransomware

- This type of malware is designed for one purpose only. To infect your system and encrypt all of your files demanding payment for their return. For example, WannaCry was a ransomware that infected over 200 000 computers, spreading across 150 countries in the world. It dealt heavy economical damage and to this day it remains one of the largest cyber attacks.

- Spyware

Like it's name suggests, this program is used to spy on it's victims by secretly recording your activity and sending it over to the person who designed it.

- Phishing

Phishing bots use the tactic of social engineering by tricking people into getting themselves infected through incredible offers transmitted via e-mail or social platforms. Some just demand your data but there are some more persuasive ones that can trick even trick people into losing money.

## **Denial of Service**

During a denial of service(DOS) assault, traffic is flooded into systems, servers, or networks in order to deplete resources and bandwidth. As a result, genuine requests are unable to be fulfilled by the system. This attack may also be launched using many hacked devices that have been tampered and turned into BotNets with the instructions to all do the same thing, attack. This type of escalation is called a distributed denial of service(DDOS). It is best described in the Computer Security: principles and practice[WS17].

## **Spoofing Attacks**

Spoofing attacks can be a very effective way to get through firewalls, yet they aren't well discussed in the security literature. Spoofing is the method of making



a packet appear to come from a computer other than the one from which it originated. Typically, attackers construct packets from scratch, indicating their preferred source and destination, and then send them out on the network to be routed. With the firewall compromised, there is nothing that can stop an attacker to infect your system.

### **2.2.4 Auditing**

A security audit is a methodical assessment of an information system's security by determining how well it adheres to a set of standards. The security of the system's physical setup, environment, software, information handling procedures, and user behavior are usually assessed during a complete audit.

Security audits are important. How important one may ask, well, in "The Art of Software Security Assessment" [MD06] we are presented with the authors' opinion on the importance of audits. It is stated that the unfortunate reality is that software suppliers are providing minimal assurances of quality in some cases and the fact that the end user agreement stands by it just proves the point. In the pursuit of adopting quality assurance methods for pleasing the clients, methods which concentrate on marketable considerations like features, availability, and overall stability, the security factor is often overlooked and the result may prove critical.

As stated, an audit covers a large range of system areas like networking, security controls, information processing, encryption so that is why the need for a vast pool of knowledge in these domains is required. The process is also known in the industry as penetration testing [Kim18].

An audit is performed with different kinds of tools and the need for such tools is growing hence my desire for trying to create a port scanner.

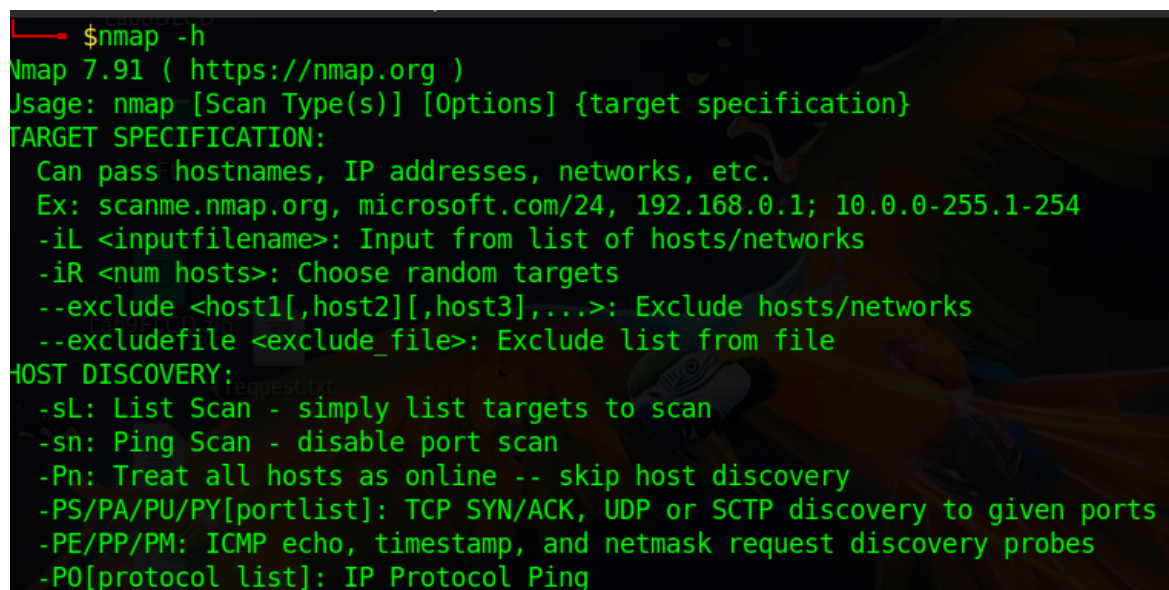
# Chapter 3

## Related work

There are a lot of port scanners currently available on the internet, some more proficient and complex than others. I chose to mention two of the most notable ones as i have some degree of experience working with them, Nmap and Masscan.

### 3.1 Nmap

Nmap, which is short for "Network mapper" is a free and open source utility tool for port scanning and security auditing. There are a lot of professionals, like network administrators and security auditors, that use nmap to perform tasks that are vital in their line of work. I myself used Nmap and found it to be very useful as long as you knew the commands thoroughly. [Lyo09]

A terminal window with a dark background and green text. The command '\$nmap -h' has been executed, displaying the help information for Nmap 7.91. The output is organized into sections: 'Usage', 'TARGET SPECIFICATION', and 'HOST DISCOVERY'. The 'TARGET SPECIFICATION' section lists various options like -iL, -iR, --exclude, and --excludefile. The 'HOST DISCOVERY' section lists options like -sL, -sn, -Pn, -PS/PA/PU/PY, -PE/PP/PM, and -PO.

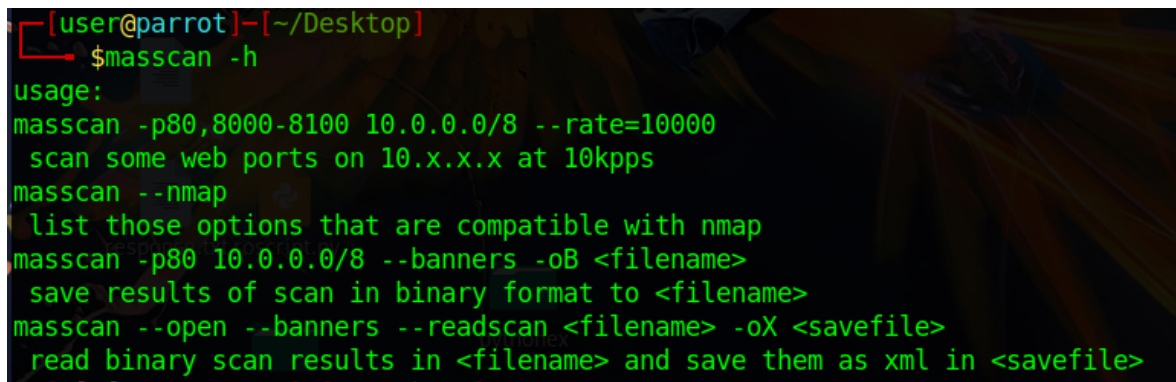
```
$nmap -h
Nmap 7.91 ( https://nmap.org )
Usage: nmap [Scan Type(s)] [Options] {target specification}
TARGET SPECIFICATION:
  Can pass hostnames, IP addresses, networks, etc.
  Ex: scanme.nmap.org, microsoft.com/24, 192.168.0.1; 10.0.0-255.1-254
  -iL <inputfilename>: Input from list of hosts/networks
  -iR <num hosts>: Choose random targets
  --exclude <host1[,host2][,host3],...>: Exclude hosts/networks
  --excludefile <exclude_file>: Exclude list from file
HOST DISCOVERY:
  -sL: List Scan - simply list targets to scan
  -sn: Ping Scan - disable port scan
  -Pn: Treat all hosts as online -- skip host discovery
  -PS/PA/PU/PY[portlist]: TCP SYN/ACK, UDP or SCTP discovery to given ports
  -PE/PP/PM: ICMP echo, timestamp, and netmask request discovery probes
  -PO[portlist]: IP Protocol Ping
```

Figure 3.1: nmap specification example

As mentioned above, it is a very powerful tool but one must do some research before knowing how to use it properly. Part of nmap's options can be observed in Figure 3.1. All things considered, they also designed a GUI application that uses Nmap, called Zenmap, which fixes a lot of its user-friendliness problems.

## 3.2 Masscan

Masscan is another powerful scanner that claims to be even faster than nmap. It is said that it can scan the whole internet in 6 minutes at an overwhelming speed of 10 million packets per second, according to its creator, Robert Graham. Masscan's options can be observed in Figure 3.2.

A terminal window with a dark background and green text. The prompt is [user@parrot] - [~/Desktop]. The command \$masscan -h is entered. The output shows usage instructions for masscan, including options for port ranges, rate, banners, and saving results in binary or XML format.

```
[user@parrot] - [~/Desktop]
$masscan -h
usage:
masscan -p80,8000-8100 10.0.0.0/8 --rate=10000
  scan some web ports on 10.x.x.x at 10kpps
masscan --nmap
  list those options that are compatible with nmap
masscan -p80 10.0.0.0/8 --banners -oB <filename>
  save results of scan in binary format to <filename>
masscan --open --banners --readscan <filename> -oX <savefile>
  read binary scan results in <filename> and save them as xml in <savefile>
```

Figure 3.2: masscan specification example

I found myself using it on a couple of occasions when Nmap couldn't get the job done in the desired amount of time or when it couldn't find something that was there for sure but it kept failing to detect it.

# Chapter 4

## Application Structure and Design

The development of a software application must be a tedious process and i tried to make it that way by documenting how i constructed my version of a port scanner, so, with all being said, the steps are described in the next subsections.

### 4.1 Used Technologies

When i first started thinking about the development process i decided on what i have to do to achieve my idea of a basic level port scanner, and so, by that, i chose the technologies that would facilitate my needs the best.

#### 4.1.1 Programming Language

For the programming language part i knew that i needed something that can assure me the implementation is very cooperative with the libraries i intended to use and so i chose Python3 as it offers the users a lot of options in that regard. Python is a very powerful language that can aid the user in learning new technologies by providing the simplicity needed to make the learning process simpler and quite easy. It is also able to run just about everywhere with minimum set-up efforts.

#### 4.1.2 Libraries

##### PyQt5

PyQt was developed by Riverbank Computed Limited and is a Python package developed as part of the Qt project that allows you to utilize the Qt GUI framework.

Qt is developed in the C++ programming language but you can construct programs significantly faster using it from Python without compromising much of the performance of C++.

## **Socket**

Socket is a python module that provides access to the BSD socket interface, a programming interface that allows programmers to create and manipulate internet connections. It uses inter-process communication and it is implemented as a library.

## **Psycopg2**

Psycopg2 is the most popular PostgreSQL database adapter that can be used in python. It is implemented in C as a libpq wrapper so it makes it very efficient and most python types are accepted as they are and adapted to PostgreSQL data types automatically.

### **4.1.3 Database**

I chose to use PostgreSQL because it is a highly stable database management system that has been constantly improved since it's creation by the community. It is a free and open source relational database management system that i grew fond of during developing other projects.

### **4.1.4 Graphical User Interface**

For it's ability to design stylish graphical user interfaces with ease, i chose Qt Designer. It is able to provide the user with a couple of advantages like saving a lot of time by not writing the GUI code by hand and keeping the code and implementation files in a clean and organised manner. It also links well with python through the PyQt collection of modules.

## 4.2 Software architecture

This section describes both the details of implementing the application and the architecture of it as well.

### 4.2.1 Description

A port scan is a method used by security auditors to find open doors or weak spots in a network. A port scan attack assists the perpetrators in locating open ports and determining whether they are receiving or transmitting data. It may also tell if a business employs active security measures such as firewalls.

When users send a message to a port, the answer they get indicates whether or not the port is in use and whether or not there are any possible vulnerabilities that might be exploited.

A port scanner is a tool that facilitates the process of a port scan. By knowing all that i designed and implemented a basic port scanner with the option of providing information on discovered ports.

### 4.2.2 Application layers

For developing a cleaner and reusable code i chose to implement a layered architecture specific to the design of the application.

Layered architecture abstracts the overall perspective of the system while giving enough information to comprehend the functions of individual levels and their interactions. Each modular layer in a layered architecture is reliant on the layers underneath it but entirely independent of the ones above it. The class diagram can be seen in Figure 4.4, and the use case diagram in 4.5 for a better understanding of the relation between the user and the application.

#### Scanner layer

The Scanner layer contains the Scanner class. It is designed to perform a TCP or an UDP port scan based on a provided hostname and a range of ports.

The TCP Scan method is implemented by trying to perform a socket connection based on the AF\_INET family of addresses that is used to designate the addresses that the socket can communicate to and the SOCK\_STREAM, a constant that indicates that the connection will be performed as a stream of data. If the connection can be established it means that the port is open and can communicate with the Scanner and, if otherwise, the port is closed and a connection cannot be performed. After the connection attempt, the socket is closed and the result is passed on. The algorithm is represented in Figure 4.1

```
s = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
s.settimeout(1)
try:
    con = s.connect((destination_ip, destination_port))
    return "[TCP]Host: " + str(destination_ip) + ", Port " + str(destination_port) + " is open!"
except:
    return "[TCP]Host: " + str(destination_ip) + ", Port " + str(destination_port) + " is closed!"
finally:
    s.close()
```

Figure 4.1: TCP Scan code snippet

The UDP Scan method is implemented by trying to perform a socket connection based on the same family of addresses that is used in the TCP Scan and the SOCK\_DGRAM, a constant that indicates that the connection will be performed as a datagram, the UDP way. In the same manner, if the connection can be established, the port is open and the port is closed otherwise and the connection cannot be performed. The algorithm is represented in Figure 4.2

```
s = socket.socket(socket.AF_INET, socket.SOCK_DGRAM)
s.settimeout(1)
try:
    con = s.connect((destination_ip, destination_port))
    return "[UDP]Host: " + str(destination_ip) + ", Port " + str(destination_port) + " is open!"
except:
    return "[UDP]Host: " + str(destination_ip) + ", Port " + str(destination_port) + " is closed!"
finally:
    s.close()
```

Figure 4.2: UDP Scan code snippet

The Scanner class also contains two(TCP and UDP) rangescan methods that use their respective scan method to generate results for all the ports in the specified range.

## Database layer

The database layer represents the data abstraction layer class that processes data from the app and from the database and bridges the communication between the two of them.

With the aid of `psycopg2(4.1.2)`, a postgresSQL database adapter designed for python, the `DatabaseAdapter` class performs the connection to the PostgreSQL database and creates a cursor when it is initialized by using hardcoded database details and credentials for testing purposes. The cursor allows python code to execute database commands in the context of the database session wrapped by the connection.

For security reasons, mainly to protect the database against SQL injection attacks, the methods use prepared statements, a parameterized method of executing SQL commands. It also provides a higher level of efficiency by being able to perform multiple operations way quicker than a normal string based method.

The database diagram can be found in Figure 4.3.

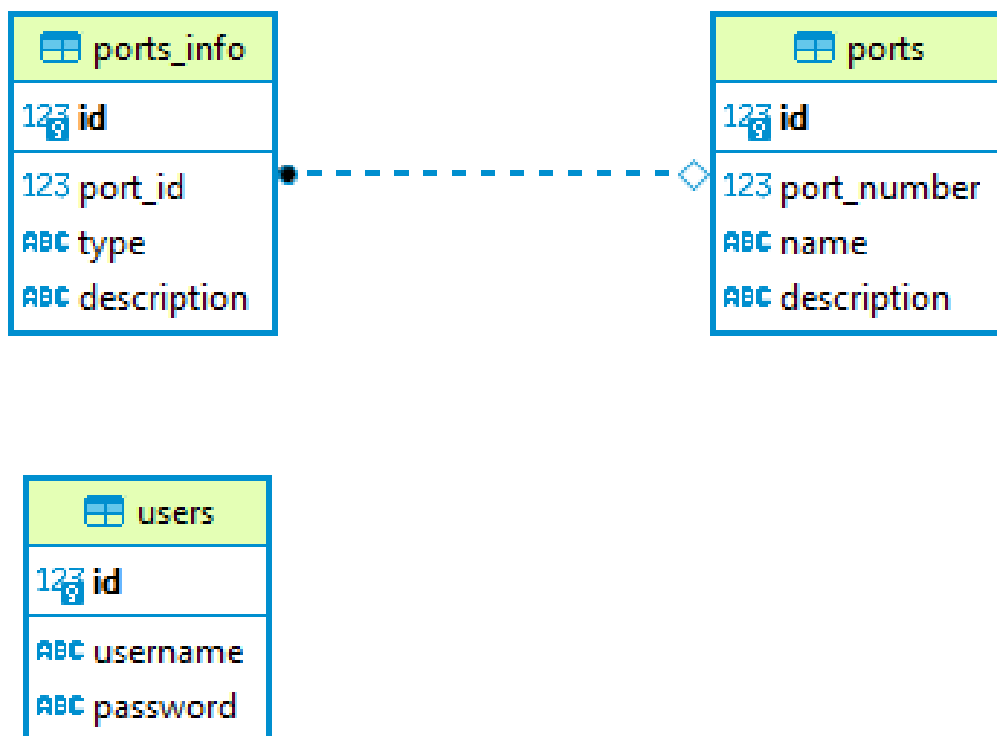


Figure 4.3: Database diagram



The application needs the following functions to provide data from the database:

- **getAllPortInfo Function**

This method receives as a parameter a port number and the cursor will execute a query on the database that searches for all the corresponding information belonging to that port.

- **registerCheck Function**

The function that checks if the database already contains the provided username and it returns True if it does. Otherwise a DatabaseException needs to be raised as the cursor is unable to provide a result.

- **registerUser Method**

The register method that introduces the username and password combination into the database.

- **logIn Function**

The cursor executes a SELECT query on the database based on the provided user and password combination. It returns the boolean value of the match between the provided combination and the obtained combination.

## **Validation layer**

I created the validation layer for the sole purpose of effective data validation and error handling. The data that comes from the service needs to be sanitized and validated before reaching the scanner or the database, otherwise the application may behave in an unknown destructive manner. It contains the ServiceValidator and the custom exceptions i created for the program.

The ServiceValidator class tests the data that comes from the service before sending it to the Scanner.

First of all, it checks if the hostname is valid by checking it's structure. A hostname needs to be a valid IP address or a valid DNS. If it's an IP, it checks whether it has the right format and if the bit fields that compose it are valid. The structure of an IP address can be described by four decimal numbers separated by dots with each number representing an eight bit value. If it is a DNS, it must contain a valid domain name from a known domain list.

Secondly, it checks whether the port range provided is valid. The values must be integers and the lower bound cannot be of greater value than the upper bound.

The custom exceptions i created are all inherited from their super class, Exception and are presented in the list below.

- DatabaseException - Exceptions regarding the database.
- ServiceException - Exceptions regarding the service.
- ValidatorException - Exceptions regarding the validator.

## Service layer

The Service layer contains the Service class that is responsible with handling the connection between the interface and the lower level classes. It contains an instance for each of the DatabaseAdapter, ServiceValidator and PortScanner classes.

The details of the procedures used in the Service class:

- scanTCP  
Function used for validating the TCP Scan that is going to be sent to the scanner and handling the result.
- scanUDP  
Function used for validating the UDP Scan that is going to be sent to the scanner and handling the result.
- discardClosedPorts  
Function that takes a list of found ports that keeps the open ones and removes the closed ones, used when the user wants to see only the open ports on a host.
- getInformationForPorts  
Function that gets the information for a specific port from the database and returns the crafted response.
- checkCredentials  
Function that calls the logIn from the database and handles the result.
- checkRegister  
Function that calls the username check from the database and handles the result.
- register  
Function that calls the register feature from the database and handles the result.

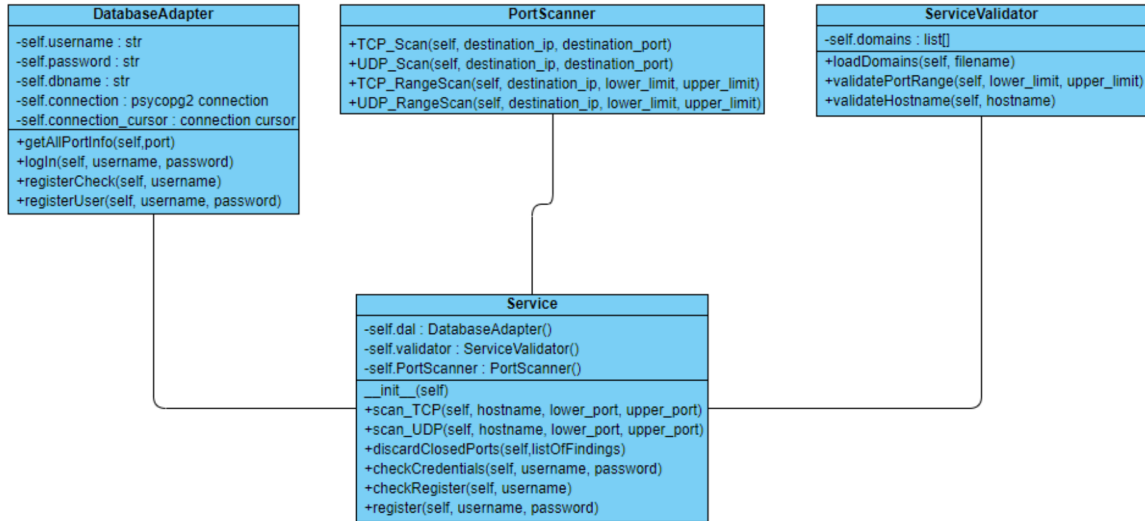


Figure 4.4: Class diagram

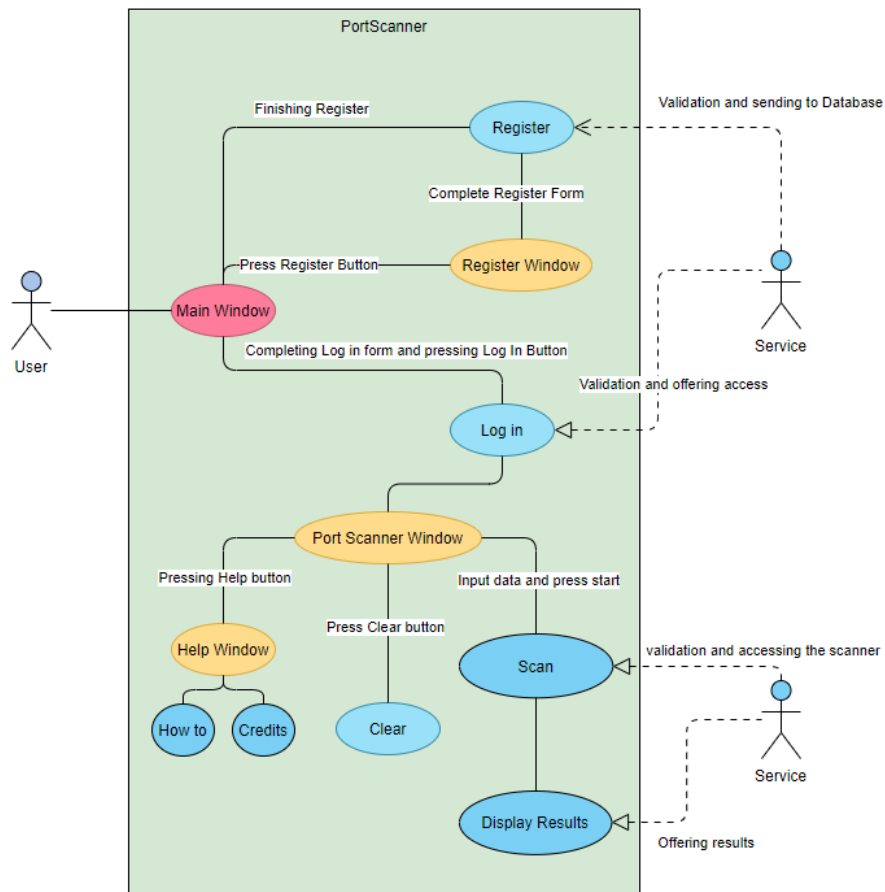


Figure 4.5: Use case diagram

## 4.3 Interface design

The design of the graphical user interface was done with the help of Qt Designer and PyQt5 library. It led to the creation of four screens, detailed below, that are encountered by the user when accessing all of its features.

### Log in Screen

The log in screen provides the user with a form in which he can input the details of an already existing account and two buttons.

The first button is the log in button which signals the service the user wants to access the log in feature. After the validation and verification of the service, if the respective account exists, the user will be logged in and sent to the Port Scanner screen.

The second button is the register button which sends the user to the Register screen.

The view of that screen can be observed in Figure 4.6.

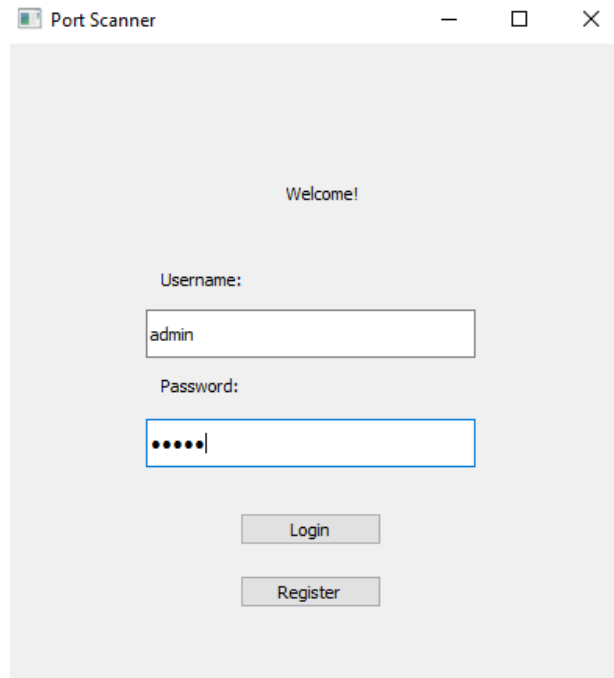


Figure 4.6: Log In Screen

## Register Screen

The register screen is consisted of a form where the user can input the data he wants to create a new account with and the register button that signals the service that he wants to attempt to proceed. It is illustrated in Figure 4.7.

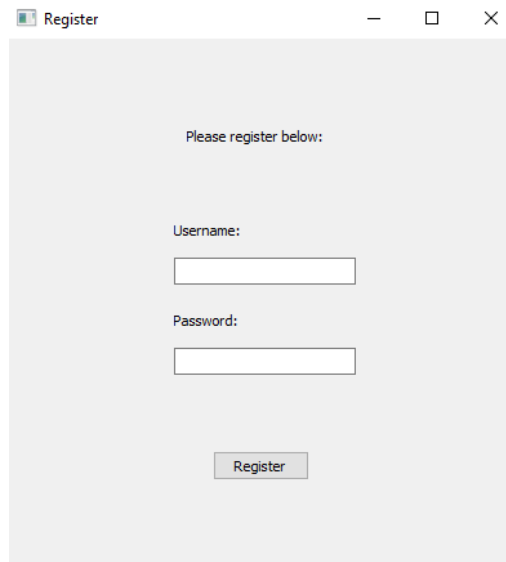


Figure 4.7: Register Screen

## Exceptions

Exceptions are handled internally by the service and are transmitted to the GUI in the form of MessageBoxes that display the description of them, as portrayed in Figure 4.8

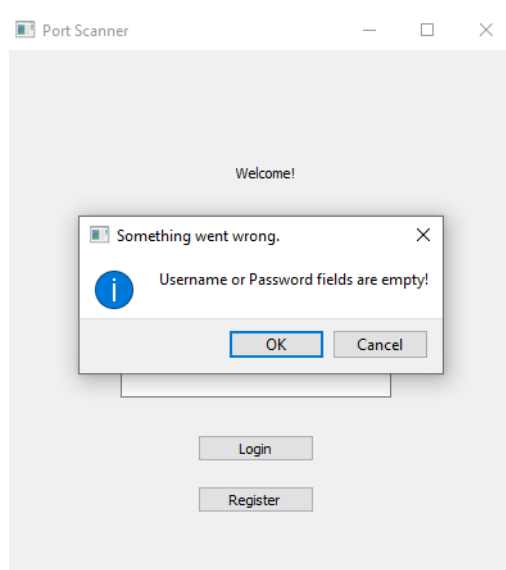


Figure 4.8: Exception Screen

## Port Scanner Screen

The port scanner screen offers the main application screen and with that a list of options and features as it can be observed in Figure 4.9.

- Help button - opens the Help screen
- Clear button - clears the result list and browser box
- Input Form for hostname and port range - set the details for the upcoming scan
- Scan type checkboxes - sets the type of scan that is going to be performed
- Start button - signals the service to begin the scanning validation and process
- Show only open ports checkbox - option to discard the closed ports
- Result list - List where the resulted ports will be saved, a port can be selected to display it's corresponding information in the browser box
- Information browser box - at the start of this screen displays a welcome message, after that it can be populated with information from a selected port.

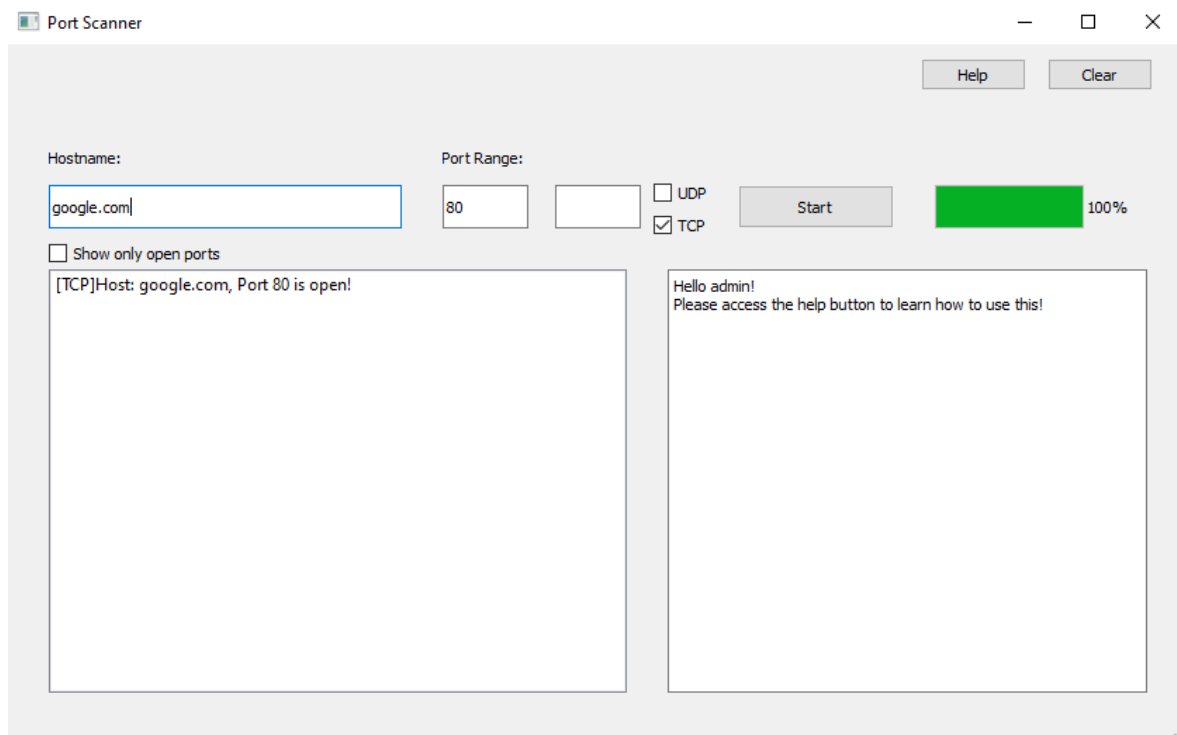


Figure 4.9: Port Scanner Screen

## Help Screen

The help screen is designed for users that require information about how to use the application or information about the application itself.

It contains a menu that expands and offers two options:

- How to

The How to part explains in an informal fashion how a user can use the application and how he cannot use as it may interfere with the application's implementation. The screen can be found in Figure 4.10

- Credits

The credits part offer the user details about the project's owner and a legal notice about the used technologies. It can be observed in Figure 4.11

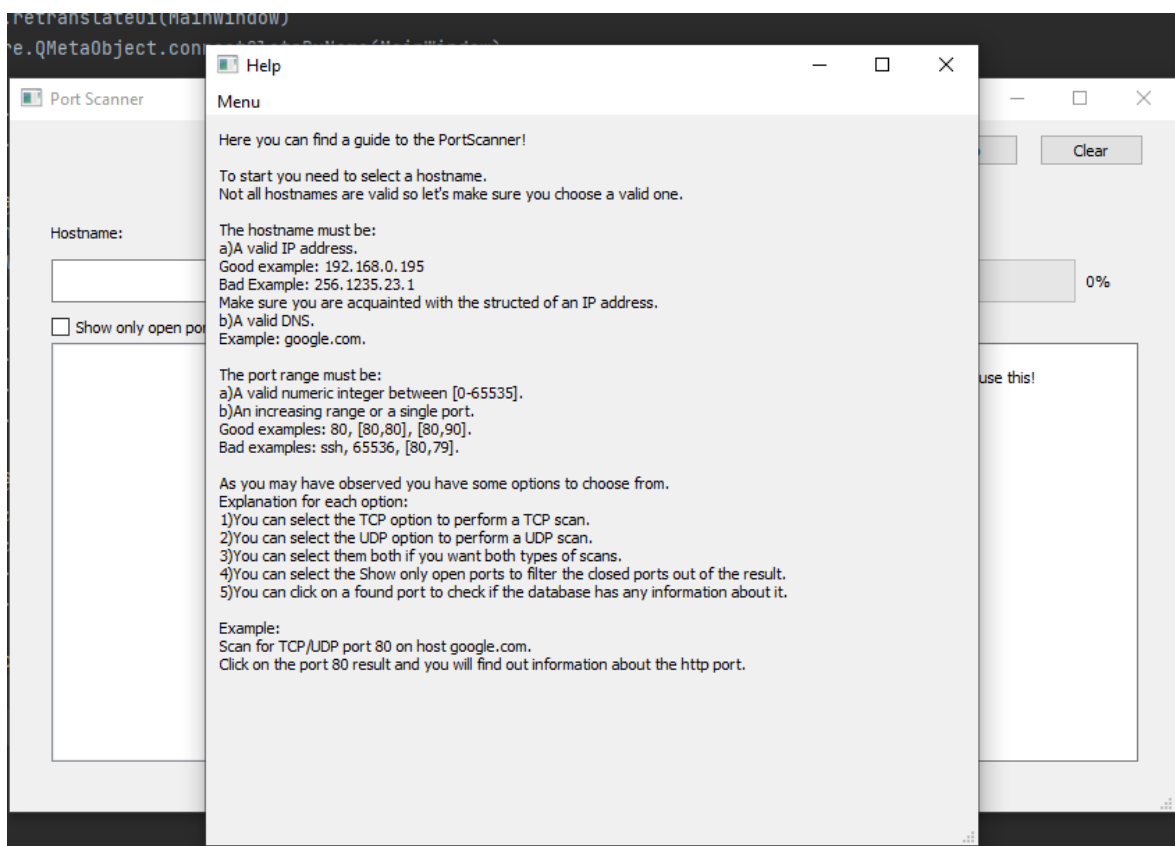


Figure 4.10: Tutorial Screen

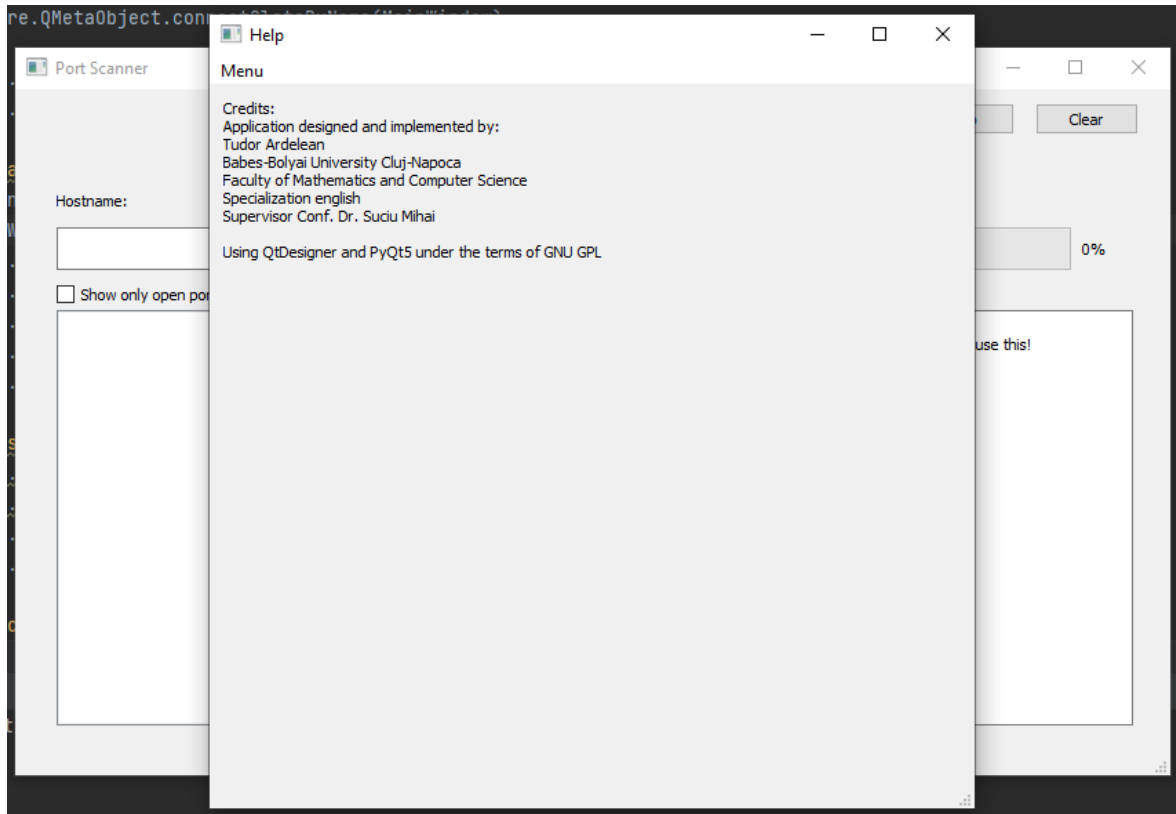


Figure 4.11: Credits Screen

## 4.4 Testing

The practice of assessing and confirming that a software product or application performs what it is intended to accomplish is known as software testing. Preventing problems, lowering development costs, and increasing performance are all key advantages of testing.

To provide a certain level of quality assurance i have decided to implement unit testing at the service level of the application as it handles most of the functions and methods used in the rest of the project.

Unit testing is a kind of software testing that examines individual software units or components. The goal is to ensure that each unit of software code works as intended. Unit testing is carried out by developers throughout the development phase of an application. They are used to isolate a piece of code and ensure that it behaves the way it needs to. In our case, the unit is the service and the tests are based on it's functions and methods as described in the list below.

- testScanTCP

The test used to check that the scanTCP function performs as intended by performing a TCP scan and checking it's results. It is also checking if the validation is done correctly and the exception are caught and handled.



- testScanUDP

It works with the same strategy as in the previous test, but for the UDP variant(scanUDP).

- testGetInformationFromPorts

It tests the getInformationFromPorts that provides a list of information chunks to see if it is performing as intended and the result is correct.

- testDiscardClosedPorts

It uses an artificial created list of ports that the function must modify and the test makes sure it does it correctly.

- testCheckCredentials

It checks if the log in is working as intended so that the application may not develop security breaches.

- testCheckRegister

The checkRegister function must determine whether the provided data exists in the database and the test makes sure it is acting as intended.

- testRegister

Test that checks if unsafe data can't harm the database and breach it.

The tests were run with the help of Python's unit test built in framework unittest and the results test results were all passing, as it can be identified in Figure 4.12

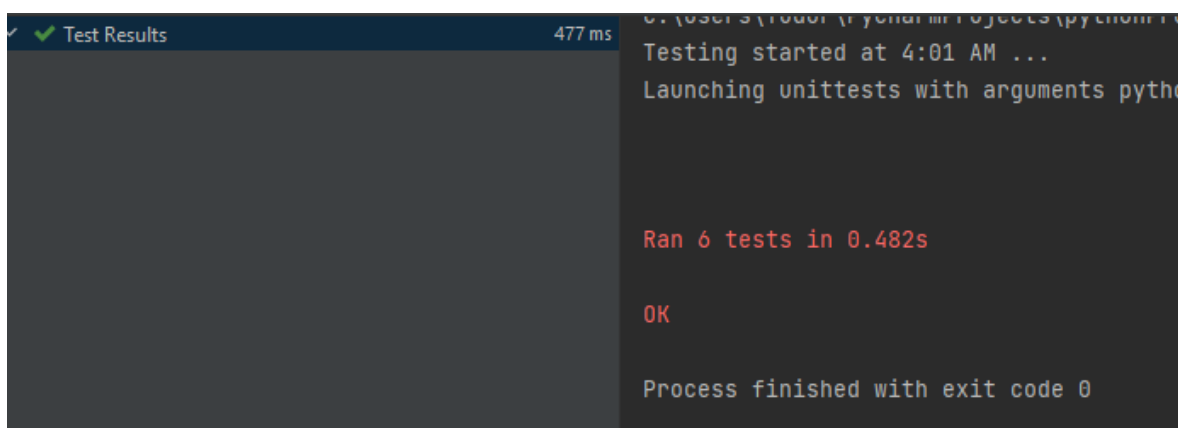


Figure 4.12: Test results

## **4.5 User Manual**

With knowing that i feel that the application has a very friendly user interface that even provides the users with the option to test it with the provided example, i decided to also make a basic user manual that references the things i talked about in the previous subsections.

First of all, the main screen that the user is presented when opening the application is the Log in screen. It allows the user to Log In with the corresponding credentials or jump to the Register form screen to create an account. The details about each object that can be interacted with can be found in Log In Screen 4.3.

If the user does not possess a personal account and was directed to the register screen, it can create an account by submitting the form with his personal details, form that is described in Register Screen 4.3.

After the authentication has been performed, the user is directed to the main application, the port scanner. There, the user can begin testing it's functionalities or opting for being provided with further information about how to use the app. The scanner's features are presented in Port Scanner Screen 4.3.

The help screen provides the user with how the scanner was designed to be used by providing examples with valid and invalid data. It also contains the credits which the user can read by choosing it from the menu. Help Screen 4.3 offers the representation of the help screen.

Lastly, by doing something that the application's internal structure cannot support or allow by it's logical definitions, the user is presented with a MessageBox that wraps an exception. It is exemplified in Exceptions 4.3.

# Chapter 5

## Conclusions and Future Work

### 5.1 Conclusions

As i started on this journey of a thoroughgoing study, i never expected to be so drawn to this part of the IT industry as it provided me with a lot of valuable information about the dangers of unsafe software developing and faulty networking implementations. As the pool of knowledge is as deep and vast as an ocean, i never expected to get that proficient from following this study and, to be honest, i don't think it got me there yet as i found that i have so much more work to put in to better myself and so much information i haven't mastered yet. But with all being said, i do believe it represents a starting point in my professional career as i dipped my toes in the water and, as the times goes by, i will continue to develop myself into an individual capable of understanding more and developing even better solutions for this field of study.

The perils of the vast web are expanding with each day passing by and most of us in the software developing community are not prepared for the threat it presents. With that being said, hope is not lost as new solutions for protection are being developed by individuals from all around the globe.

The port scanner that i developed is proof of that, proof of the fact that people are willing to learn and spread information about how such threats can affect us and the industry. I know that, at this time, my implementation is not capable of performing more than providing people that are willing to start a journey of their own in this vast domain with a place to start and get accustomed to the idea of scanning and identifying potential threats from a network but i plan on further developing it in the future to maybe, one day, be recognised as a valid option for recon work.

## 5.2 Future Work

Here are some ideas that i plan on developing in the future:

- Global database connection

I always envisioned a global database connection that can provide the users with real time information that is being updated on a constant basis, as new data about software vulnerabilities is found constantly.

- Packet crafting technology

The technology the port scanner is using is a basic socket connection trial and error, but if packets can be modified in a way that the attackers can get past firewalls and find further entry points into the system, then we also must be able of replicating that attack vector and see how we can fix the existing problems.

- Logging and monitoring

Monitoring the application is a must if it is ever to be launched as an all around tool.

- Data exporting

I think highly of the idea that two different software products can share data between them that can help one or both of them so exporting the findings to files that can be used later would a nice addition.

- User accounts

The application also needs a better accounting structure as the one implemented is just a mock up that is used for testing the features.

# Bibliography

- [CA07] Felix Lindner Gerardo Richarte Chris Anley, John Heasman. *The Shell-coder's Handbook: Discovering and Exploiting Security Holes*. Wiley, 2nd edition, 2007.
- [DS11] Marcus Pinto Dafydd Stuttard. *The Web Application Hacker's Handbook: Finding and Exploiting Security Flaws*. Wiley, 2nd edition, 2011.
- [Eri08] Jon Erickson. *Hacking: The Art of Exploitation, 2nd Edition*. No Starch Press, USA, 2nd edition, 2008.
- [Fou21] OWASP Foundation. Owasp. <https://owasp.org/>, 2021.
- [Kim18] Peter Kim. *The Hacker Playbook 3: Practical Guide To Penetration Testing*. Independently published, 2018.
- [Lyo09] Gordon “Fyodor” Lyon. *Nmap Network Scanning: The Official Nmap Project Guide to Network Discovery and Security Scanning*. Nmap Project, USA, 1st edition, 2009.
- [MD06] Justin Schuh Mark Dowd, John McDonald. *The Art of Software Security Assessment: Identifying and Preventing Software Vulnerabilities*. Addison Wesley Professional, USA, 1st edition, 2006.
- [Sea19] Chad Seaman. Anatomy of a syn-ack attack. <https://blogs.akamai.com/sitr/2019/07/anatomy-of-a-syn-ack-attack.html>, 2019.
- [WS17] Lawrie Brown William Stallings. *Computer Security: Principles and Practice*. Pearson, 4th edition, 2017.