”What if someone steals it?” Hands-on evaluation of the software security work of a networked embedded system

ADINA BORG & HEDDA KLINTSKOG
MASTER´S THESIS
DEPARTMENT OF ELECTRICAL AND INFORMATION TECHNOLOGY
FACULTY OF ENGINEERING | LTH | LUND UNIVERSITY

"What if someone steals it?" Hands-on evaluation of the software security work of a networked embedded system

Adina Borg
ad4600bo-s@student.lu.se
Hedda Klintskog
he4125kl-s@student.lu.se

Department of Electrical and Information Technology
Lund University

Supervisor: Christian Gehrmann

Examiner: Thomas Johansson

June 9, 2022

Abstract

As information technology has grown and evolved, so has the need of securing the information. It is important to evaluate both the security of systems and the methods which are used for the evaluation. One method for finding security vulnerabilities in a system is penetration testing. The goal of this thesis is to evaluate some tools and methods for penetration testing.

The methods were examined by performing a penetration test on a network horn speaker. The penetration test followed the state-of-the-art methodologies and was performed in three steps: reconnaissance, scanning and exploitation. Free open-source tools were used to perform attacks. The testing evaluated the security of the speaker, considering both a network attacker and an attacker with physical access to the speaker. The security work done by the company that develops the speaker was evaluated by comparing the results from the testing with the vulnerabilities found by the security work. A conclusion can be drawn that penetration testing should not be the only method for securing a system, and that threat modeling is a good way of finding vulnerabilities and attacks.

An important conclusion from the testing is how a penetration tester should use tools to conduct attacks. The thesis shows the importance of using multiple tools for the same attack as well as the importance of not blindly trusting tools.

Most penetration tests described in the literature from the field are performed on websites or entire organizations; this thesis contributes with knowledge about how to evaluate the security of a network embedded system.

Acknowledgements

First, we would like to sincerely thank our supervisor at the company for her involvement and support. We would also like to give a big thank you to all employees at the company who have given us input along the way. We would like to thank Christian Gehrmann for his valuable comments and guidance throughout our thesis. Lastly, a big thank you to friends and family for all support.

Popular Science Summary

Today, information technology (IT) is constantly growing and our society becomes increasingly dependent on IT solutions. Some systems are negligible while others are very critical and their loss of data or downtime can be devastating. While IT grows so does the criminal activity in the cyber-world. Hackers have been around for as long as computers have been but the view of a "hacker" has changed. A lot of people associate the term "hacker" with teenagers, hoodies, basements and possibly with the hacking organisation *Anonymous*. These are relevant associations, however, a big part of hacking today is done by people who have it as their profession. Hacking is performed by criminal organisations as well as by Nation States and federal organisations. It can also be performed under the term "ethical hacking" which is when hacking is used legally to find and mitigate security vulnerabilities.

It is common for companies and other organisations to hire *Penetration testers* to try and hack their systems. A penetration tester is an ethical hacker who has permission to hack a system as a criminal hacker would do. The penetration tester report all security vulnerabilities found to the hiring organisation which they can then mitigate, making it more difficult for a criminal attacker to find vulnerabilities.

In this thesis, a penetration test is performed on a horn speaker, a speaker used in places such as airports and building sites. The penetration testing is partly performed to find possible security vulnerabilities. It is also performed as a way to evaluate different methods and tools used during penetration testing. It is studied how easy it is to find and use different methods and tools. Further, an evaluation is also done of the process which has been performed by the developers to make threat models and construct a speaker with as few vulnerabilities as possible.

To be able to understand the motivation behind attacking this type of speaker, different main goals an attacker could have were mapped out before starting the penetration testing. One goal could e.g. be to steal sensitive data while another one could be to play your own audio from the speaker and all speakers connected to it on the network. When the penetration testing began it started out from a *Black Box* perspective since we had no previous knowledge about the speaker. A system is seen as a Black Box if no knowledge exists of how the system works. During the penetration testing we used methods and tools which were recommended by experienced penetration testers and when their recommendations were not enough other popular options were looked for, compared and tested. It was shown that having good methods to follow is important to keep penetration testing structured. It was also shown that using existing tools can be both positive and negative since it can ease the work but it is also a risk of missing vulnerabilities if the penetration tester does not understand how the tools work.

After the different attacks were tried against the speaker it was possible to compare the approaches tried during the penetration testing with the threats in the threat models done by the company. Here we saw that threat modelling is a good method for finding many possible threats. The category captured by our penetration testing that was missing from the threat modelling was misuse of the system. This includes scenarios where an attacker intentionally uses the system wrong in order to hack into the system.

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

As information technology has grown and evolved, so has the need of securing the information. Cybersecurity is an important part today in everything from private usage to government organizations and companies. For companies developing technical products, it is not only important to secure the business data and structures but also the products they are selling, since it otherwise can affect and hurt the customers.

The thesis will be performed at a leading company in network solutions. The company's products can be found all over the world and the devices are placed in a wide range of areas, both more enclosed spaces such as within private companies but also in public places such as airports and metros. To secure their products from intruders, the company offers built-in cybersecurity features to protect against cyberattacks, prevent unauthorized access and battle vulnerabilities. For a company to be able to ensure promises like this, its products need to monitor for vulnerabilities and continuously test, develop and patch their products. One way of testing is through *penetration testing*.

1.1.1 Penetration Testing

Penetration testing is a legal attempt to locate and exploit vulnerabilities in a system in order to make that system more secure. The difference between a penetration test and a real attack is authorization, motivation and intent. An ethical hacker always obtains approval before performing a penetration test. The motivation of a penetration test is to improve the security of the target. The intent of a penetration test is to perform a simulation of an attack from a malicious hacker to provide useful information about which vulnerabilities an outsider can exploit. Someone who performs a penetration test is a *white hat hacker*. If a person attacks a victim for personal gain, this person is considered a *black hat hacker* [14].

1.1.2 Case Study

Another useful method when performing tests or evaluations is to define a *case study*. A case study is a study that focuses on a specific case and goes in-depth into that specific scenario [44]. In this thesis, the scope of the penetration testing will be set by defining a case study. The case study will restrict the project to focus on one of the company's products, a network speaker. The penetration testing will only focus on getting access to or information from this product in order to restrict the scope of the thesis. It will also be implicit that the speaker has been correctly installed and no other person with malicious intent has been involved during production or setup.

The speaker

As described in the previous section the target device of the case study will be a speaker. The speaker is described by the company to be an all-in-one horn speaker with built-in power amplifier and signal processing (DSP). The speaker contains onboard memory which can store pre-recorded voice messages, a built-in microphone and the possibility for security personnel to live speak through it. A logical view of the speaker can be seen in figure 1.1.

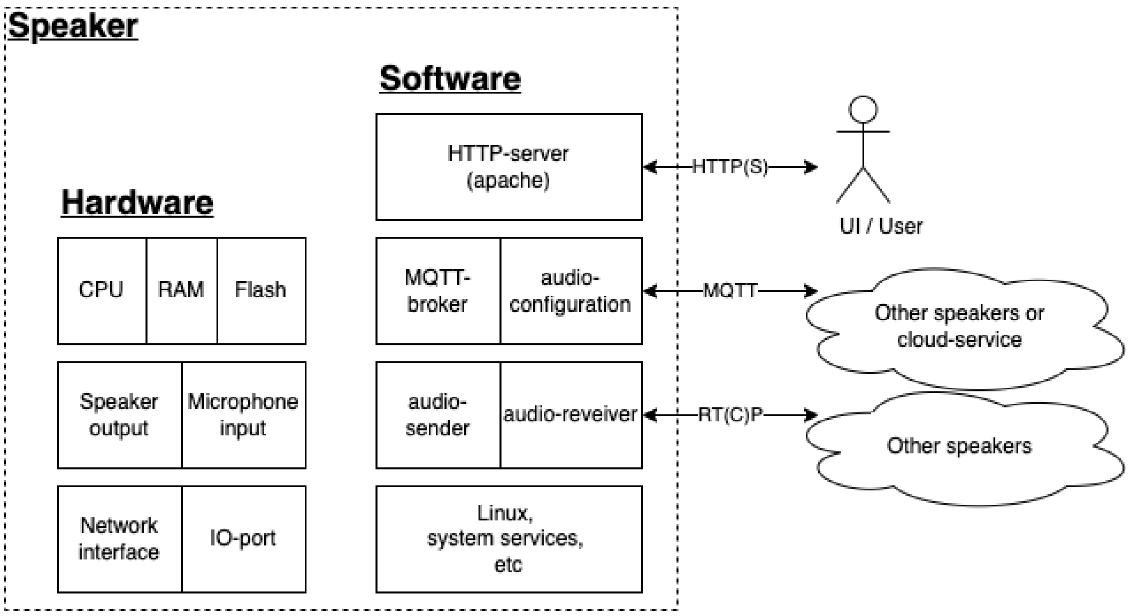

Figure 1.1: A logical view of the speaker.

The speaker plugs right into standard Internet Protocol (IP) networks. It supports Power over Ethernet (PoE), which means that a single cable provides both power and connectivity [47]. The speaker is also based on open standards such as Voice over Internet Protocol (VoIP) telephony (using SIP). It's possible for a user to manage different zones and schedule content for the speaker.

VoIP is a technology that enables voice communication over Internet Protocol (IP) networks. It is enabled by Sessions Initial Protocol (SIP) and Dual-Tone Multi-Frequency (DTMF) signalling. SIP is a signalling protocol used to initialize, maintain and terminate VoIP calls between two or more participants [4]. DTMF is a signalling system that can be used for sending commands to remote devices to make them perform certain actions [46].

A user can interact with the speaker via the company's own open API. With the help of this API, the user can for example restart a device, get a server report or see a list of all device parameters.

A speaker is often a part of a larger network with many other speakers. These speakers are controlled by a leader, which is used to control the audio output from the speakers. The size of the networks can vary a lot, some customers have networks consisting of up to 40 thousand speakers.

1.2 Project Goal and Motivation

The goal of this thesis is to find security vulnerabilities in a networked embedded system in order to evaluate the process of securing the system. Another part of the thesis will be to evaluate the performance of the tools and methods used to find vulnerabilities in the system.

Hacking is a wide area and can be done legally, illegally or in the grey area in between. It can be performed by everyone from a nation state hacker who has hacking as a profession or an individual doing it as a "hobby", to gain e.g. money or power [34]. The level of knowledge that a hacker possesses can therefore vary a lot. This results in attacks being unpredictable and hard to counter for, since the different backgrounds of the hackers decide how they will try to attack the system. Someone who works as a black hat hacker for a nation state has a lot of resources, but is more likely to perform the hacking at normal working hours and to use a predefined method while performing the hack. But someone who does it in their free time and is entirely taught might not be as predictable with time and methods. On top of this, black hat hackers do not follow any laws or rules which also adds another layer of difficulty.

When securing a system, a company can use existing methods and standards for assessing its system and the severity of different vulnerabilities that could apply to its system. Although well-tested standards are used to evaluate the system there is always a risk of missing or looking past a vulnerability.

This thesis will test how well the methods used by the company to evaluate their networked embedded systems actually cover possible vulnerabilities since we will try and find weaknesses of the system using both tested methods and other knowledge gained throughout our education.

Chapter 2

Theoretical Background

In this chapter, we describe the information collected during the literature study. We cover the theory of various areas such as tools, methods and recommendations from experienced penetration testers.

2.1 Literature Study

The literature study is the first part of the thesis work and explores what is already written in the chosen area. This includes books, articles, reports, organisation websites, blog posts and news articles. We started by looking at what is generally written about penetration testing and how common the testing method seems to be.

In penetration testing the target to be tested is often a big entity such as an organization or a company, but it can also be a specific system or product. In our case, we are going to perform a penetration test of a networked embedded system and therefore it is also relevant to read about penetration testing that has been done on other networked embedded systems.

2.1.1 State-of-the-art

Since penetration testing is often performed on a company's system or products it is generally not publicly available to read how the tests were performed and what the outcome was. However, it is possible to read books written by penetration testers and other professionals which describe the penetration testing methodology. These books often describe a penetration test step by step and also recommend tools and possible attacks. The books contain the author's personal opinion and experience about which tools to use and how each phase of a penetration test should be performed. During this thesis a number of this kind of books were studied, including [14] [15] [12] [19].

All books reviewed use an approach where they consider the different phases of penetration testing. The authors also discuss areas such as ethics and tool usage. In all books, the first phase in penetration testing is information gathering. This phase is often called the "reconnaissance phase" or "information gathering phase". Following this phase is a phase referred to as the "scanning phase" or "vulnerability identification". The next phase is commonly referred to as the "exploitation phase" and this phase is often divided further into specific areas such as network, web, databases and social engineering. These divisions look different depending on the different authors. The next step after the exploitation phase also varies a bit between the different books. In [14] the following step is to maintain access and then finish off by writing a report on the testing. In [15] the last step is to write a penetration testing report. Commonly the last step is to write a report.

After comparing different sources and getting a good overview of the different phases we looked further into the first three phases to see what their goals are and what is important to consider while performing the different steps. We also looked at what methods and tools the different authors recommend for the different phases.

2.1.2 Reconnaissance Phase

Reconnaissance in penetration testing is similar to reconnaissance in military operations, it is about obtaining information about the target using methods that observes the target without intervening with or "touching" it [19]. The reconnaissance phase is the first phase of penetration testing. The goal of the phase is to collect as much information as possible about the target to create a picture of how the target operates and to find possible weaknesses.

A big part of the reconnaissance phase is to look at available public information about the target. This step allows the pentester to gather a lot of useful information without sending a single packet to the target. Another aspect of mapping out the target is to create a list of possible IP addresses to attack [14]. This step is especially important when the target is a company, corporation, government, or other organization.

Reconnaissance can be both passive and active. During passive reconnaissance, information is gathered without connecting to the target directly. Passive information can be gathered from third-party sources that have collected information about the target. Examples of this are news about the target or their social media and job postings [15]. In active reconnaissance, information is gathered by connecting directly to the target. Active reconnaissance includes examining the target's website, extracting information from their email server and surveying their network [15].

When Dr. Patrick Engebretson explains the reconnaissance phase in [14], he points out that the work done in this phase affects how well you succeed in the later stages. He also states that it is important to have a strategy and that a typical strategy includes both active and passive reconnaissance. The first thing performed is usually locating the target's website and then thoroughly reviewing it. One way of doing this is to use a tool that creates an offline copy of the site. Here you can find information about the business such as the number of employees, partnerships and information about the employees such as phone numbers or email addresses. Engebretson also mentions “News” and “Announcements" as important sources as well as looking at open job postings for the target company. The next area he talks about is how to effectively use *Google* to gather information about the target. He also talks about looking at other types of sites on the internet to find information on social media, forums and other places where employees might have left useful information. Useful information in this case can for example be company email addresses which can give hints on possible usernames.

2.1.3 Scanning

After the Reconnaissance Phase, it is time for Scanning. This phase can, according to Engebretson [14], be divided into three steps:

1. Determine if a system is alive.
2. Port scanning the system.
3. Scanning the system for vulnerabilities.

The first step is to determine if a system is alive and if it is possible to communicate with it. The method Engebretson suggests for this is to use pings and ping sweeps. Ping is a network packet called ICMP (Internet Control Message Protocol) packets which are used to investigate the reachability of a host. A ping sweep is "a series of pings that are automatically sent to a range of IP addresses" [14]. Engebretson also points out that regardless of the outcome of this step you should always continue with the next steps.

The second step is to identify which ports are open on the target network. A port is a data connection that enable information to be exchanged between computers. In his book, Engebretson describes different types of port scans, for instance *TCP Connect scan*, *SYN Scan*, *UDP scans*, *Xmas Scan* and *Null scans* [14].

The third step is to locate and identify weaknesses of the target network. After finding open ports in the system, it is time to find possible vulnerabilities for these ports. There exist tools that help the user find weaknesses that can be exploited for the open ports on a target. It is also possible to perform the vulnerability scan by looking at a database with known vulnerabilities to try and find applicable vulnerabilities for the target system [15]. The Common Vulnerabilities and Exposures (CVE) program can be used to find applicable vulnerabilities. The program identifies, defines, and catalogs a list of publicly known cybersecurity vulnerabilities [38]. After the vulnerability scanning is done it is time to start the next phase, the exploitation phase.

2.1.4 Exploitation

The goal of the exploitation phase is to be able to control the target system. This phase includes the attacks that most people associates with "hacking". Compared to the other phases, exploitation is the least structured phase of a penetration test. The purpose of this phase is to find vulnerabilities and exploit them, and this can vary a lot depending on the target. Examples of attacks that can be performed in this phase are password cracking, network sniffing, password resetting and man-in-the-middle (MITM) attacks [14].

Engebretson structures his chapter about the exploitation phase by explaining some tools and what goal they can help achieve. The first attack described is gaining access to a remote service. Engebretson states that a lot of hackers use "online password crackers" to brute force usernames and passwords. These password crackers use lists of passwords and/or usernames that are exhaustively tried to find a valid combination. To increase the chances of finding a successful login, Engebretson points out that one should use information from the reconnaissance phase where possible usernames and passwords might have been discovered. The tools "Medusa" and "Hydra" are mentioned as two popular tools to use and Engebretson goes into further detail about the Medusa tool.

The next tool mentioned by Engebretson is *Metasploit*, he talks about the history of Metasploit, why it is important for penetration testing and mentions some examples of what it can be used for and proceeds to show an example of it in use. Further, he talks about *John the Ripper* which is used for cracking password hashes, sniffing network traffic to gain access to the system using the tool *Wireshark*, the tool *macof* which is used to attack flood a switch and finishes the chapter with *Fast-Track Autopwn* which is a tool that automates the entire exploitation phase. In the next chapter, Engebretson focuses on web-based exploitation.

In [19] the Author Jeremy Faircloth has decided to not have one exploitation phase chapter but rather divide the exploitation phase chapters based on different exploitable areas and make them into their own chapters. The chapters concern the areas network devices, web application, databases, enterprise applications, client-side attacks and social engineering as well as wireless penetration testing. Some of these areas are similar to what Engebretson writes about in [14] and some are not. Generally the chapters in [19] goes more in depth of the specific areas compared to [14]. In [15] the author Thomas Wilhelm has a mix of the previous two approaches where he first presents different tools in an "exploitation phase" chapter and then goes more in depth on specific areas such as local system attacks, privilege escalation, network, web applications and support systems such as databases. Amongst the tools Wilhelm also focuses on *Medusa* for brute forcing and *Metasploit* for exploiting a handful of other services.

Looking at these three different approaches of the exploitation phase it shows that they all start by focusing on the results from the vulnerability scanning as well as brute forcing login credentials. Further, they focus on different areas and in what area and in which order depends on the target system and the results from the reconnaissance and scanning phases.

Chapter 3

Approach

The security evaluation will start with *black box testing.* A black box test is done without any knowledge about the internal infrastructure, hence the testers will know the function of the device but not how the function is implemented [35] [21]. At the beginning of the thesis, all that was known about the speaker was the public information about it. Because of this, it could be simulated that the speaker had been stolen. By starting with no inside information about the speaker, it was to be investigated how much damage an outside attacker could do and provide a new perspective about the security. The next step will be *white box testing.* In white box testing, the testers have full information about the system they are investigating and have good knowledge about the source code [21] [36]. In this phase, the documentation from the security work done by the company will be a source of information.

3.1 Attacker Goals

To better get into the mindset of a malicious hacker it needs to be understood what their intent could be when attacking the speaker. Here the threat model technique called *Attack Trees* was used, a method to analyse and describe threats against a system [2]. The process started by analysing what the end goals could be for an attacker who decides to hack the speaker.

One of the most common reasons for IT-related attacks is money. If attackers can access secret or sensitive data they could sell it to competitors. Therefore one of the main goals could be to "steal sensitive or secret data". Another way to make money for hackers is to perform ransomware attacks, an attack that locks out the victim from their system and requires payment in order to unlock the systems. Here the main goal could be "Extort the owner of the speakers" and performing a ransomware attack would be one way to carry out this goal.

As internet of things (IoT) devices increase so do embedded systems that are used by hackers in botnets. As mentioned in the Introduction, the speakers can be connected with up to 40 thousand other devices and if someone managed to take control of the entire network of speakers this could cause a lot of harm. The possibility to be able to control this amount of devices would seem appealing to for example botnet owners. The network of speakers being used as a botnet is a scenario where the speakers will be used to perform an illegal activity and this creates the main goal "Use speaker(s) for illegal activities".

The last two main goals are related to the function of this specific network embedded system. Since the testing is of a speaker made for announcements two goals an attacker could have is to prevent the announcements and important audio from being played or to play their own audio through the speaker. The main goals are therefore "prevent legitimate audio from being played" and "play own audio from speaker(s)".

In conclusion, this resulted in the five high level attacker goals:

1. Steal sensitive or secret data
2. Extort the owner of the speakers
3. Use speaker(s) for illegal activities
4. Play own audio from speaker(s)
5. Prevent legitimate audio from being played

From the high level attacker goals and the attack scenarios discussed in relation to them, a high level attack tree has formed and can be seen in figure 3.1.

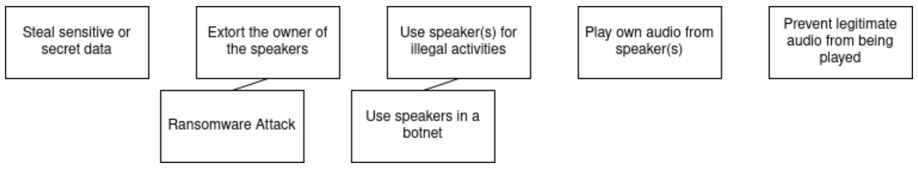
Steal sensitive or secret data
Extort the owner of the speakers
Use speaker(s) for illegal activities
Play own audio from speaker(s)
Prevent legitimate audio from being played
Ransomware Attack
Use speakers in a botnet

Figure 3.1: High Level Attack Tree.

The main goals are broken down into sub goals and specific attacks in order to get a clearer picture of how to prioritize different attacks during our penetration testing.

Steal sensitive or secret data

An attacker who wants to steal sensitive or secret data for profit can either focus on stealing data from the company that develops the speakers or from the organization that uses the speakers. Either way, the data stored in the speaker needs to be accessed or leaked. One way to get access is to impersonate a legitimate login session. This can be done by figuring out the correct username and password, finding useful information by analysing network traffic or using social engineering to trick a legitimate user to give up their credentials. Another way to gain access is to use an exploit on the web service or other parts of the system. It could also be possible to leak information without gaining access to the system, by using malware or attacks like buffer overflow or stack overflow. If the attacker has physical access to the speaker they could tamper with and analyse the hardware. These scenarios result in the tree seen in figure 3.2.

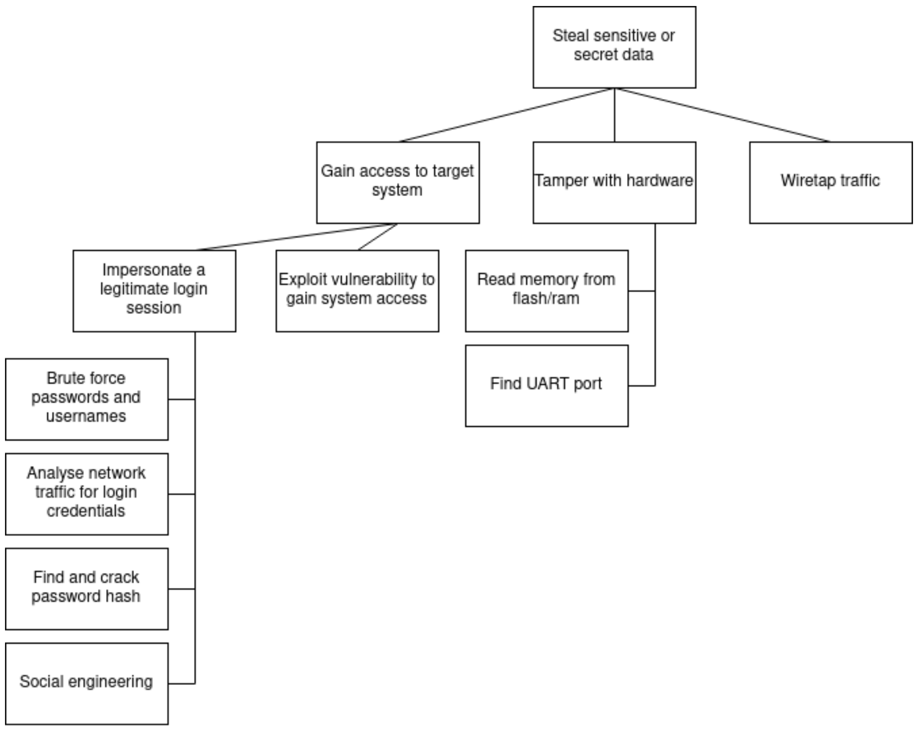
Steal sensitive or secret data
Gain access to target system
Tamper with hardware
Wiretap traffic
Impersonate a legitimate login session
Exploit vulnerability to gain system access
Read memory from flash/ram
Find UART port
Brute force passwords and usernames
Analyse network traffic for login credentials
Find and crack password hash
Social engineering

Figure 3.2: Attack tree for *Steal sensitive or secret data*

Extort the owner of the speakers

The idea of this goal is to force the victim to pay the attacker in order to get their business or systems back to normal functionality. This could be done by locking the victim out of their system through encryption or by threatening to expose their data. The exposure of data would require the attacker to have accessed the data which makes it identical to the previous main goal. Another way of preventing the speakers from proper function is to perform a denial-of-service (DoS) attack on the system. The tree of this goal can be seen in figure 3.3.

Use speaker(s) for illegal activities

This goal could be met by either physically stealing the speaker or taking advantage of the speaker without removing it from its position. An attacker who steals it could possibly want to mount it somewhere else and play their own audio. Or, as mentioned previously, they could intend to use the speaker in a botnet. The goal could also be to infect the speaker and use it to spread viruses to other devices or to let the attacker spy on the network. Further, if the attacker wants to hide themselves, they could use the speaker as a proxy to make it harder to track them down. The tree created from these scenarios can be seen in figure 3.4.

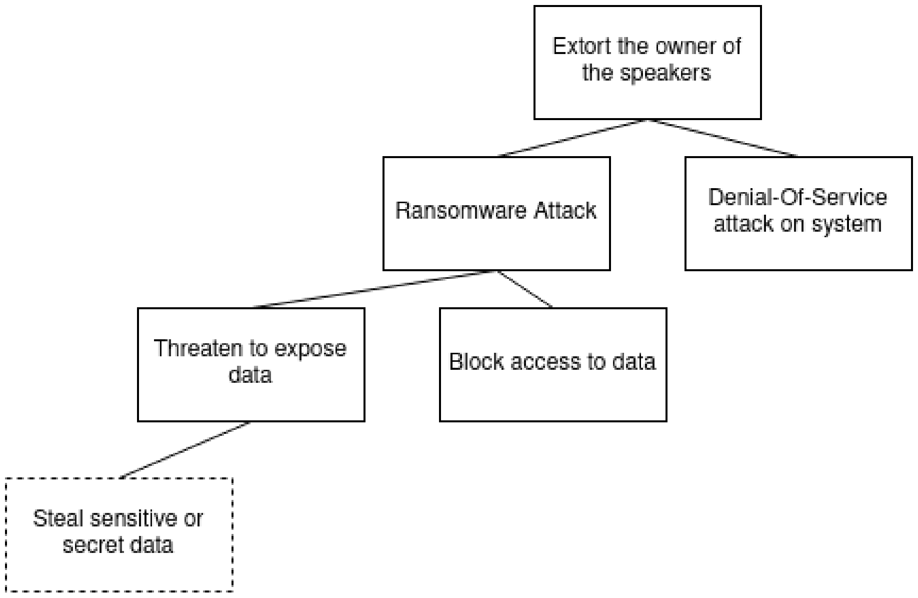
Extort the owner of the speakers
Ransomware Attack
Denial-Of-Service attack on system
Threaten to expose data
Block access to data
Steal sensitive or secret data

Figure 3.3: Attack tree for *Extort the owner of the speakers*.

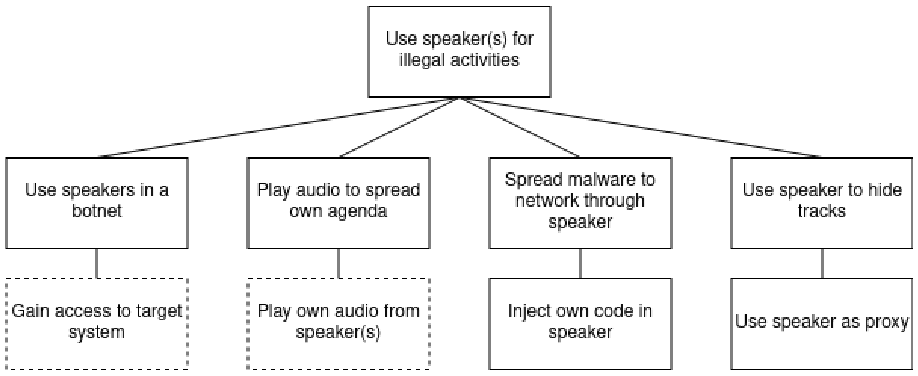
Use speaker(s) for illegal activities
Use speakers in a botnet
Play audio to spread own agenda
Spread malware to network through speaker
Use speaker to hide tracks
Gain access to target system
Play own audio from speaker(s)
Inject own code in speaker
Use speaker as proxy

Figure 3.4: Attack tree for *Use speaker(s) for illegal activities*.

Play own audio from speaker(s)

There are two options regarding the attackers playing their own audio. Either to play from only the targeted speaker or from all speakers in the network. To play from all speakers the attacker would want access to the leader. A sub goal would hence be to get access to the leader through the target speaker. Another option that could work for both scenarios is to impersonate the leader to the speakers.

As with the "Steal sensitive data" goal, one way to play your own audio would be to impersonate a legitimate user and login on the speaker (or leader) by breaching the correct login credentials. Another option here as well is to look at the network traffic and try to find login credentials or to perform a replay attack where the attacker would try to resend packets or modify the content of the packets. The tree created from these scenarios can be seen in figure 3.5.

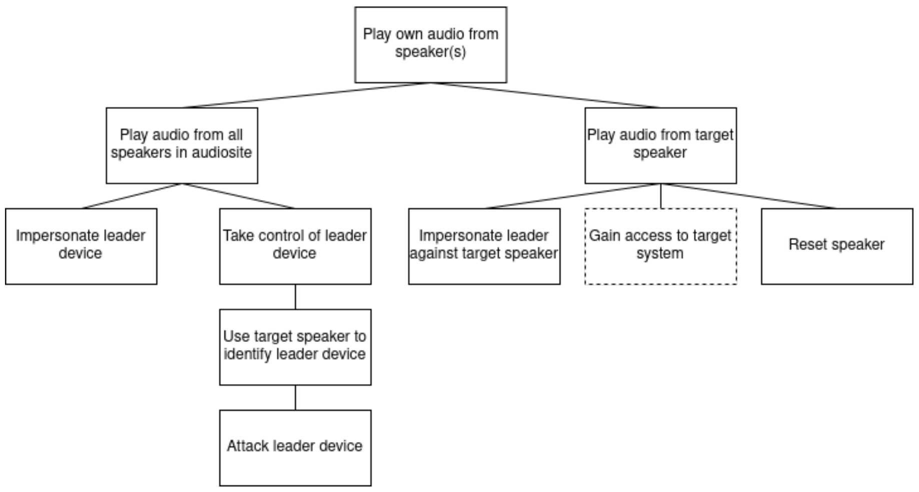
Play own audio from speaker(s)
Play audio from all speakers in audiosite
Play audio from target speaker
Impersonate leader device
Take control of leader device
Impersonate leader against target speaker
Gain access to target system
Reset speaker
Use target speaker to identify leader device
Attack leader device

Figure 3.5: Attack tree for *Play own audio from speaker(s).*

Prevent legitimate audio from being played

Identical to "play own sound", this goal could target one speaker or all speakers in the network. As the speakers are used for example fire and flood alarms, the consequences of preventing audio from being played could be lethal. A common way to prevent systems from performing their tasks is to execute a DoS attack. If the attacker only wants to prevent audio from the target speaker then that one speaker could be DoS:ed. If all speakers are to be prevented, the target of the DoS attack would instead be the leader. Another way to prevent the audio is to prevent the packets from reaching the correct destination or to remotely turn off the speakers when an announcement is playing. It is also a possibility for the attacker to insert malicious software in the speaker or tampers with the hardware. The tree created from these scenarios can be seen in figure 3.6.

All leaves at the end of these five trees provide more specific attacks which can be tested during the exploitation phase. This information provides more material to use when prioritising and selecting which areas and attacks to focus on during testing.

3.2 Deciding on Strategies and Attacks for Execution Phase

When preparing for the penetration testing it is needed to decide on methodologies and strategies to follow. Since all books in the literature study followed a similar methodology, it was decided to use a similar approach. We have been working with three major phases: reconnaissance, scanning and exploitation. The last step in the books was commonly reporting. In this thesis, the reporting will look different

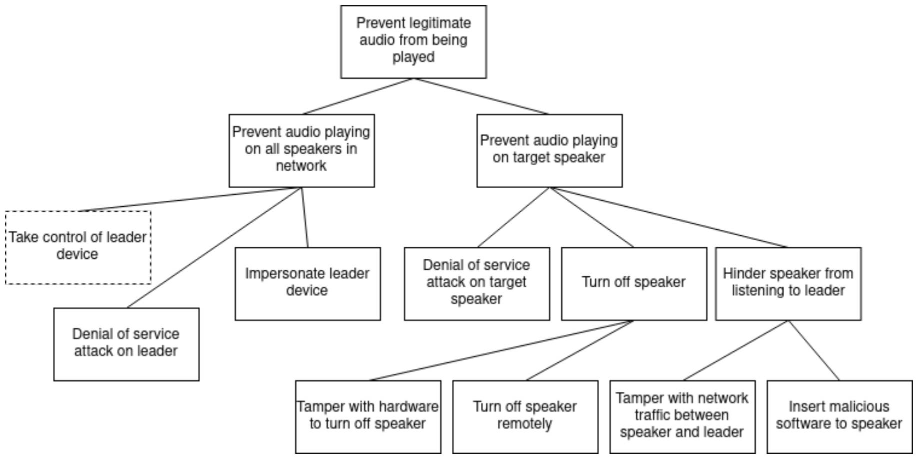
Prevent legitimate audio from being played
Prevent audio playing on all speakers in network
Prevent audio playing on target speaker
Take control of leader device
Impersonate leader device
Denial of service attack on leader
Denial of service attack on target speaker
Turn off speaker
Hinder speaker from listening to leader
Tamper with hardware to turn off speaker
Turn off speaker remotely
Tamper with network traffic between speaker and leader
Insert malicious software to speaker

Figure 3.6: Attack tree for *Prevent legitimate audio from being played.*

from penetration testing in the field. As some of our identified vulnerabilities are company confidential, we here only report our non-confidential findings.

All sources used to find a methodology treat the reconnaissances phase and scanning phase similar which makes it more straightforward to follow these steps and use the tools and methods recommended by the authors. The exploitation phase is the phase that need the most decisions taken regarding deciding direction. The decisions will be based on the areas brought up in the literature study and the attacks from the threat modelling as well as what is relevant in our case study.

As mentioned by Engebretson in [14] the first thing hackers tend to do if they during the scanning find services with remote access is to try and brute force the passwords and usernames to those services. Depending on what open ports show up during the scanning of the speaker it is reasonable to start by trying to brute force the login credentials given that a service with remote access is found. If an attacker were able to find a password, they would be able to do a lot of harm.

From the information gathered in the literature study, there is no fixed order to perform different tests within the exploitation phase. Instead, the focus is to decide what areas and attacks are relevant to focus on and can be applied considering our case study.

From the attack tree, it is visible that a lot of attacks will be available to perform once the attacker cracks the login credentials. Therefore we will look at multiple ways of retrieving these credentials and not stop if we manage to crack them using one method. The customers access the device through their browser which gives us another area to focus on when choosing attacks.

One area that will be examined in this thesis is which kind of different attacks can be done if the attacker has physical access to the speaker. The speakers can be placed in remote places, where they could easily be removed without it getting noticed by the owner. It is therefore important to find out what an attacker could do if they stole a speaker. Different attacks that require physical access to the speaker will be tried during the execution phase to see what information is possible to gain. One of the main attacker goals from the Attack Tree modelling is to *Steal sensitive or secret data* and one of the ways to achieve this is going through the hardware and physically accessing the memory storage or looking for the debug port that developers use while developing the device and investigate the possibility to access information through that.

Since the speaker will most likely be connected to a local area network together with other speakers and a leader device, it will be relevant to look at what damage can be done from analysing and/or interfering with the network traffic to and from the speaker.

The speakers can be used for emergency warnings, and a likely attack against this is a distributed denial-of-service (DDoS) attack. An attacker could do a lot of harm by preventing the user from accessing the speakers. Because we have limited resources, the focus will be on simpler DoS attacks and not full DDoS attacks.

There are some areas mentioned by the different authors from the literature study that this thesis will not focus on unless they prove to be relevant from the reconnaissance phase, scanning phase or other tests. Examples of these areas are databases and wireless penetration testing.

This thesis will not focus on social engineering. According to Watson in [17], one definition of social engineering could be "the art of eliciting sensitive information and/or manipulating individuals into performing actions that may result in a security breach". Using techniques from social engineering to try and retrieve usernames and passwords is a likely scenario Social engineering requires that there exist possible victims for the attacker to manipulate. The target speaker in the testing is not used by a customer, so there is no one to trick into giving away information.

After the attack tree threat modeling and following discussion, we have put together table 3.1 to follow when deciding on attacks.

3.3 Using Penetration Testing Tools

Thomas Wilhelm states in [15] that a pentester should never blindly trust tools. His experience as a professional is that there are many cases when one tool misses something that another tool can find. He thinks that tools should be combined to make sure that all possible vulnerabilities are found. Wilhelm emphasizes the importance for a tester to understand how the tools work and what they do. If the tester uses an automated tool, they should know how to replicate what the tool does manually [15]. Because of this, a second tool will be tested if an attack was unsuccessful with the first tool. In some cases, an attack will not succeed because the target has protected itself against the attack, but multiple tools will still be tested so that nothing is missed.

In [9], the authors have performed a test where they compare how well a number of penetration testing tools performed on detecting Structured Query Language (SQL) injections compared to penetration testing experts. The tool that performed best in the study only found 50.8% of the vulnerabilities that the experts found. Of the total number of vulnerabilities that the best tool found, 14%

Table 3.1: Attacks to try in testing phase

Attacks	**Relevant to look at?**
Brute force passwords and usernames	Yes, this attack could help fulfil a lot of main goals and will be tested
Listen to network traffic	Yes, this attack could help fulfil a lot of main goals and will be tested
MITM attack	Yes, possibly in combination with *Listen to network traffic*
Find and crack password hash	Yes, if a password hash is found
Find debug port on circuit board	Yes, relevant to try
Retrieve memory from flash/ram	Maybe, if time and resources exists
Use speaker as proxy	Maybe, if time and resources exists
Exploit vulnerability to gain system access	Yes, we will look for applicable CVE:s and exploits
Denial-of-service on speaker	Yes, relevant to try
Replay attack	Yes, relevant to try
Impersonate leader device	Yes, if *Replay attack* is successful
Impersonate target speaker	Yes, if *Replay attack* is successful
Buffer overflow	Yes, relevant to try
Inject own code in speaker	Yes, could be looked at in combination with other attacks such as *Buffer overflow*
Reset Speaker	Maybe, if it contributes to an attack or conclusion
Turn off speaker	Maybe, if it contributes to an attack or conclusion
SQL Injection	Yes, relevant to try
Social Engineering	No, not relevant for our use case
Attack on database	Only if discovered to be relevant
Attack wireless communication	Only if discovered to be relevant

were false positives. According to this study, when it comes to finding SQL injections, tools only found about 50% of the vulnerabilities at best. This study shows the importance of not depending only on tools, but also to look for vulnerabilities manually.

3.4 Set Up

During the testing, the prerequisite for the attacks is that the attacker either has network access to the speaker or physical access to the speaker. When performing attacks with network access the speaker could either be connected to the lab network used by the developers at the company or create our own closed off local area network (LAN). When attacks performed produced a lot of network traffic, such as brute force or DoS attacks, a smaller LAN was used to not disturb other users of the lab network. The set up of the smaller LAN consisted of a speaker and one or two computers connected to a switch, see figure 3.7. In many attacks, one computer represented the attacker device and the other represented the user. This setup affects some attacks performed, and some attacks would have a different outcome for a different setup. For example, if another switch were to be used or a hub instead of a switch.

The speaker is used by a big variety of customers and is used in everything from airports to grocery stores. Therefore, it is reasonable to assume that the security of the network that the speaker is connected to will vary as well. Some things will be common for most setups, such as the speaker being a part of a network with other speakers and other devices. There will probably exist a leader device used by the customer to control the speakers on the network. Hopefully, the customer uses a closed network. Some attacks require that the speaker is connected to the user's network, such as sniffing attacks and DoS. These attacks will be more dependent on the user's network setup. Other attacks do not require that the speaker is still accessible to the customer, such as password cracking, anonymous File Transfer Protocol (FTP) and buffer overflow. These attacks are still possible to perform if the attackers have stolen the speaker and connected it to their own network. The results from these attacks are therefore less dependent on the network setup.

The setup of the network the speaker is connected to will depend on the customer and not on the company developing the speaker. There is no standard that all the customers use, and the company can neither check nor control how the speakers are used. It is therefore not in the scope of this thesis to test different network setups. Network attacks that sometimes depend on the network setup will still be described. This is so the reader can follow the process and understand how the information used to exploit the speaker is gathered.

Attackers might require further attacks and social engineering to reach the target network. How secure the customer's network is can differ a lot and is not related to the security of the speaker. Because of this, this thesis will not examine different attacks for breaking into the customer network. Some attacks performed in execution, for example, DoS and Address Resolution Protocol (ARP) spoofing, require that the attacker first breaks into the network. Because the focus is to evaluate the security of the speaker, attacks to breach the network are out of scope.

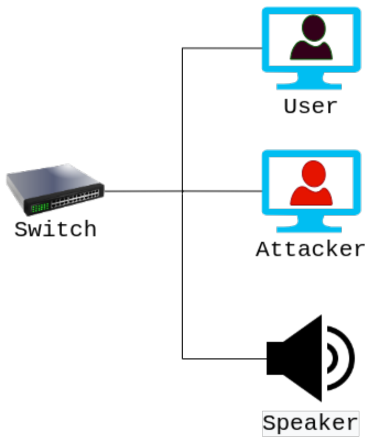

Figure 3.7: The setup of our small LAN.

Chapter 4

Execution

The execution phase of the thesis began with black box testing. In the literature phase it was found that penetration testing often is divided into reconnaissance, scanning and exploitation. This methodology was followed during the execution and began with the reconnaissance phase.

4.1 Reconnaissance Phase

In the first step of the reconnaissance phase, good sources of information about the speaker were looked for. These sources were found by using the methods recommended by the authors of the books in the literature study. After searching the company website and using Google to find documents about the speaker, it was found that a big information source is the user manual for the speaker. It is also possible to find product information directly on the company's website. These information sources are meant for customers of the company but since they are publicly available for anyone to read they can provide useful information when trying to breach the speaker.

One useful part found in the user manual is a username. When explaining how to change the password of the speaker it is stated that "The default administrator username is root". This provides the knowledge that the username will likely be *root* if the login credentials were to be cracked. Regarding the password, the manual gives recommendations which help create a stronger password. However, it is also stated that the company do not impose a password policy for its devices, as they may be used in various types of installations. This means that weaker passwords are still a possibility and therefore dictionary attacks could be successful. The manual also contains a lot of guides on how to perform tasks in the web interface. Such as setups, calibrations, resets, firmware settings, protocol specifications and a list of API commands. Another useful find shows that the company uses their own Linux-based operating system. This tells more about what can be expected from the speaker and what to search for when looking for exploits to use against the system.

The company's and distributor's websites provide information about the environments in which horn speakers are being placed and what use cases it is meant for.

One of the main tasks of the reconnaissance phase is to find target IP addresses.

A specific device with a preset IP address was given because of the case study. In the user manual, it reads that the default IP address is 192.168.0.90 if no DHCP server is available, which could be helpful information when resetting the target speaker or for finding the device on a network. In the speaker's user manual there exist help resources for identifying a device on the network that an attacker could utilize to find their target IP.

4.2 Scanning Phase

The first step of the scanning phase is to check if the system is alive. The IP address that was obtained in the reconnaissance phase is checked with the help of the ping command: `ping <IP address>`. The ping response indicated that the speaker is alive.

The second step is to perform a port scanning. For port scanning, the tool *Nmap* was used. Nmap is a free, open source tool used for network discovery and can be used to scan large networks or target a single host. It can perform a variety of tasks, including version detection, OS detection, and ping sweeps, but was originally developed to be an efficient port scanner [53]. From Nmap's perspective, a port can have one of the following states:

- open: a port is open if there exists an application that accepts TCP connections, UDP datagrams or SCTP associations on it. To find an open port is often the purpose of a port scan.
- closed: a closed port receives and responds to packets from Nmap, but there is no application listening on it.
- filtered: if a port's filter prevents the packets from Nmap to determine the state, that port is considered filtered.
- unfiltered: if Nmap can access a port, but are unable to decide if it is open or closed, the port is considered unfiltered.
- open|filtered: this state means that Nmap is unable to decide if a port is open or filtered. This can happen for scan types where open ports give no response.
- closed|filtered: this state is only used for IP ID idle scan, and is used when Nmap is unable to decide if a port is closed or filtered. [54]

Following is the result of the port scanning:

Tool: Nmap
Motivation: It is a popular tool that is downloaded by thousands of people each day [53], and it is used in the majority of the sources about port scanning from the literature study. In [14] Engebretson describes Nmap as the tool to choose if you had to choose only one tool to conduct port scanning. In [19], the author Jeremy Faircloth describes Nmap as "a standard item among pen testers and network auditors". In the paper [25], Nmap is described as one of the important tools among

the wide variety of existing penetration testing tools. Nmap was chosen to use because of a combination of its popularity, its good reputation and its accessibility.
Result: See table 4.1 for the results from the different Nmap scans. The open ports found during the scanning can be seen in table 4.2.

Table 4.1: Results from different port scannings using Nmap.

Scan type	Command	Result
TCP connect	`nmap -sT -p- -PN <IP address>`	6 open ports
SYN scan	`nmap -sS -p- -PN <IP address>`	6 open ports
UDP scan	`nmap -sU <IP address>`	Top 1000 ports are open|filtered

Table 4.2: The open ports found in the scanning phase.

PORT	STATE	SERVICE
21/tcp	open	ftp
22/tcp	open	ssh
80/tcp	open	http
443/tcp	open	https
554/tcp	open	rtsp
49152/tcp	open	unknown

Nmap is also used to find the operating system (OS) of the device.

Tool: Nmap
Command: `nmap -O -V <IP address>`
Result: Unable to determine the operative system
It makes sense that Nmap was unable to determine the OS considering it was found in the reconnaissance phase that the speaker runs an internally developed OS. The third step of this phase is to scan the system for vulnerabilities.

Tool: Flan Scan
Motivation of choice: Flan Scan is a network vulnerability scanner developed by Cloudflare and can be used to find relevant CVEs affecting your network. It was first developed and used in-house by Cloudflare but was later released as an open source project [37].
Result: No known vulnerabilities were found for the open ports on the speaker.

4.3 Exploitation Phase

After the port scanning, it was time for the exploitation phase. In this phase, the knowledge gained in the earlier phases was used in order to take control of the target system.

4.3.1 Anonymous FTP

The File Transport Protocol (FTP) is within the TCP/IP protocol suite and is used to exchange files between two networks. A possible weakness with this protocol is *anonymous FTP*. There exists a user account named "anonymous" with limited access. Usually, this account accepts any string as a password [1]. Anonymous FTP is therefore a way of entering the FTP port without knowing a password. During the scanning phase, an open FTP port was discovered. To check if the speaker had anonymous FTP enabled, *Metasploit* was used. Metasploit is an open source penetration testing framework. It contains a large database of exploits that can be used on targets. It is also possible to write your own exploits with Metasploit [55][12].

Tool: Metasploit
Motivation of choice: In [14], Engebretson describes Metasploit as his favourite tool and as the quintessential hacker tool. It was known that Metasploit could be useful for a big range of different attacks during the exploitation phase. This was a good time to get familiar with the tool.
Exploit: `auxiliary/scanner/ftp/anonymous`
Result: Unsuccessful

To make sure that the anonymous FTP was not enabled, the anonymous credentials were also used to connect manually to the FTP port via the terminal.

Method: Manual testing
Motivation of choice: A quick way to confirm results from Metasploit.
Result: Unsuccessful

4.3.2 Password Cracking

The result of the scanning phase shows three open ports which contain services with login possibility. This provides the ability to try and brute force the login credentials as a way to gain access to the target system. A brute force attack is an attack where, theoretically, every possible combination of letters, numbers and symbols is tested until a password is found. Theoretically, this attack will always yield a valid password, however, it could take years to find [43]. An option is to use a dictionary attack, see figure 4.1, which instead uses a list with common passwords from e.g. dictionaries or password leaks [45]. This technique does not guarantee that a valid password will be found, however, humans have a tendency to use ordinary words that have a meaning to them since it is easier to remember [24]. Dictionary attacks are a good option to use in a scenario where a username is known, and a valid password is to be found.

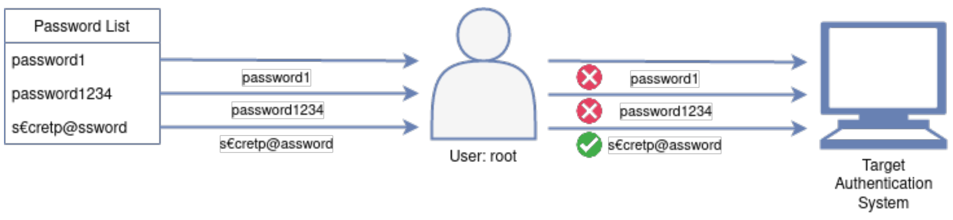

Figure 4.1: Dictionary Attack

Step one: Choosing password cracking tools to try

To perform the password cracking, there are multiple password cracking tools to choose from. To find the most suitable ones for this case study, the top most popular tools were compared. *John the Ripper*, *Medusa* and *THC Hydra* are described and mentioned by more than one of the sources in the literature study and are also commonly mentioned when looking for popular password cracking tools. Another commonly mentioned tool is *Ncrack*. The *Metasploit Framework* also contains password brute forcing attacks, making it another option. See table 4.3 for the different tools requirements and suitability for this case.

Table 4.3: Considered password cracking tools

Tool	**Requires**	**Suitable for dictionary attack**
Hydra	Password file, Username file	Yes
Medusa	Password file, Username file	Yes
Metasploit	Password file, Username file	Yes
John the Ripper	Password hash	No
Ncrack	Password file, Username file	Yes

Step two: Finding password lists

When using a password cracking tool, the attacker needs to provide a password list and one or more usernames. The next step is therefore to find or generate possible password lists. From the reconnaissance phase, it is known that *root* could be a valid username. It was therefore assumed that the username was *root* and the focus was on finding a belonging password.

Two GitHub repositories containing password lists ready to use were found when searching the internet [30] [22]. The content of the lists varied a lot, the repositories contained everything from leaked passwords to lists with bible passages. It was decided to focus on the lists with leaked passwords and popular passwords.

To complement this password list, a shorter list was written containing words connected to the company which were estimated as possible options, since these will likely not be amongst common or leaked passwords.

A script was written that takes a list of passwords and extends it by adding numbers, capital letters and exchanging a letter for a digit with a similar appearance. The idea with this script was to make the same changes to a word as a human might do to make their password stronger. The script was used to extend the list with passwords related to the company. See figure 4.2 for an example. The following things were done by the script to the words in the list:

1. Add a number between 0 and 100 or between 1950 and 2022.
2. Change a lower case letter to an upper case letter.
3. Change from an o to a 0.
4. Change from an i to a 1.
5. Change from an a to an @.

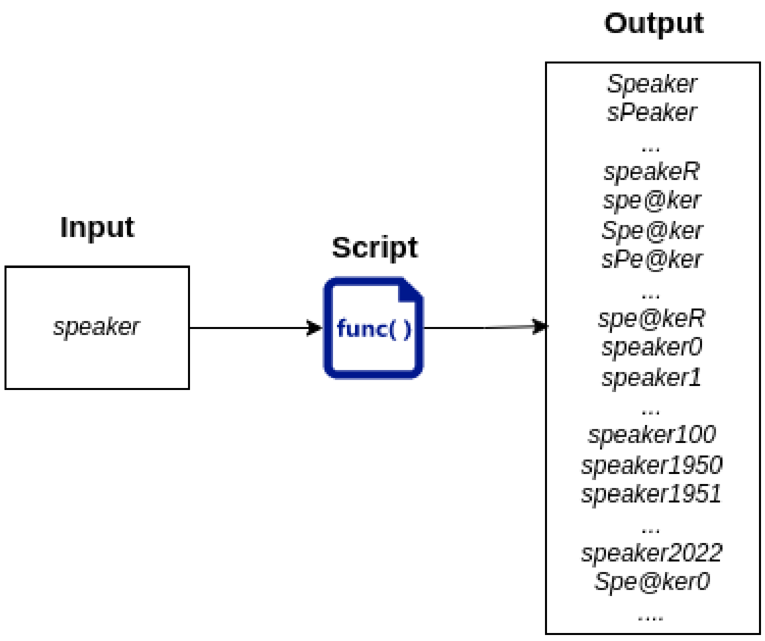

Figure 4.2: An example run of the script.

Step tree: Cracking the password

After looking at password cracking tools, Ncrack, Medusa, Metasploit and THC Hydra were left as options. All four crackers have the possibility to attack both the FTP port and the Hypertext Transfer Protocol (HTTP)/Hypertext Transfer Protocol Secure (HTTPS) ports. When deliberating on what port to start attacking, Ncrack was tested on both the FTP and the HTTP port using the same password file. The HTTP port seemed to be faster, so it was decided to start with this port.

Tool: Ncrack
Result: Claimed that no password was correct.

Tool: Medusa
Result: Claimed that all passwords were correct.

Tool: Metasploit
Result: Could not start the cracking. Showed error message "no URI found that asks for HTTP authentication".

Tool: Hydra
Result: Could not start the cracking. Also showed an error message.

These results show that none of the tools correctly managed to send and receive the password and response. To solve this, the cause needed to be found and understood. After looking into the problem, it seemed that there was no response received from the web service on whether the login was successful or not, which resulted in the confusing results from the different tools.

When looking at the options and use cases of the different tools, *Hydra* seemed to be the most suitable tool for overcoming this problem.

Tool: Hydra, with tweaked options
Result: Valid password found in under three minutes.

The password found was a five letter word, without any capital letters, numbers or special characters. The word can be found in both an English and a Swedish dictionary. The cracked login credentials were tried on the web service which yielded a successful login session. The credentials were also tried on the FTP port and to use Secure Shell (SSH) to connect to the target. The credentials were valid for both of these services.

This successful attempt showed that the speaker had no defence against brute forcing the password on the HTTP port. The attacker IP was not blocked and the *root* account was not locked.

Trying Hydra Against the File Transport Protocol Port

FTP have no defence against password guessing, there is no limit on how many times a user can try incorrect passwords [23]. The FTP port is therefore another possible way into the speaker. Ncrack was used against the FTP port using a password file with 10 million entries, including the correct password. This attempt kept running until the cracking was manually interrupted since it was taking too long. Instead, a new attempt was started with a shorter list containing the correct password. This time the cracking finished and successfully found the correct password. This shows that it is possible to crack the password using the FTP port but it is slower than the HTTP port. After the attempts on the FTP port, the supervisor at the company noticed the logs of the target system which consisted of warning statements stating that "Max connection attempts" have been reached for the IP of the machine used during the attack. However, these log warnings did not trigger anything that stopped the attacker from making more attempts.

4.3.3 Sniffing Attacks

The attack tree analysis concluded that sniffing the network traffic or tampering with the network traffic to and from the speaker is useful to achieve different goals. To be able to perform a sniffing attack, the attacker needs to listen to the network at a place where traffic containing sensitive information passes by. This is a MITM attack, an attack where the attacker takes control of the communication channel between at least two endpoints [33], see figure 4.3. There are a number of possible attacks that can be used to perform a MITM attack [14].

This scenario assumes that the attacker has gained access to the target network to which the speaker is connected. The attack targets devices on the network in addition to only targeting the speaker. Because of this, the result of these attacks depends on the setup of the target network [14]. To find a suitable attack, common network attacks were looked at.

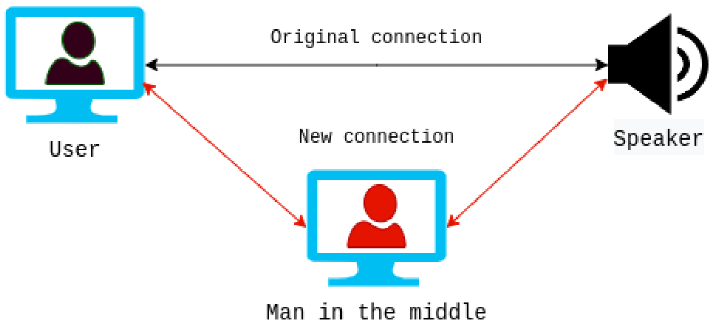

Figure 4.3: A MITM attack.

The setup of the sniffing attack consisted of the speaker, the user's computer and the attacker's computer which were all connected to the lab network at the company. The network traffic was then analysed using *Wireshark*, a network protocol analyzer that lets the user inspect the network at microscopic levels [48]. Wireshark is familiar to us as well as recommended by many, e.g. by Engebretson in [14].

ARP Spoofing

Successfully performing an ARP Spoofing attack would allow seeing the traffic going between a user and the speaker.

ARP is used to map media access control (MAC) addresses with IP addresses. ARP request packets are broadcasted on the network, asking for the MAC address that belongs to a certain IP address. The host with the concerned IP address answer with their MAC address. All hosts have an ARP cache where they save the mapping between IP and MAC address. A host will cache all ARP replies they receive, even if they didn't send out an ARP request. A host has no way of checking if the information in the ARP replay is correct, so they will blindly trust anything they receive. The entries in the table can be either static or dynamic. Static entries are added by a network administrator. Dynamic entries are learnt from the network and age out after a fixed time interval. ARP spoofing is only possible on a network with dynamic entries [6].

The attacker can perform an ARP spoofing by sending ARP replies to the victim periodically, claiming to be another host on the network. A host will cache all ARP replies they receive, even if they did not send out an ARP request. When the victim communicates with the host the attacker is impersonating, the attacker will be able to see the traffic [6].

The attack was first performed with Metasploit.

Tool: Metasploit, Wireshark for network monitoring
Exploit: `auxiliary/spoof/arp/arp_poisoning`
Result: It was possible to spoof the speaker's IP. In Wireshark, packets sent from the user to the speaker's IP were detected, however, the user did not get any responses from the speaker while the exploit was running.

It seemed that the exploit did not forward packets to the correct receiver. Examining the code behind the exploit showed no evidence of packets being forwarded. The exploit did make it possible for the attacker to impersonate the speaker, but it did also perform a DoS attack against the user. An alternative ARP spoofing tool is needed.

The documentation of a Metasploit exploit is often only a couple of sentences, if the user wants to understand how the exploit works, they need to examine the source code of that exploit. The ability to view and edit the source code of exploits is probably the reason why Metasploit is a popular tool for experts. For a new user, it can be time consuming to understand such a flexible tool as Metasploit. It might be easier to use a less flexible tool, but one with more documentation and that requires less user involvement. For this reason, the tool *Ettercap* was chosen to be tried next.

Tool: Ettercap, Wireshark for network monitoring
Motivation of choice: Ettercap is a suit of tools for MITM attacks, including ARP spoofing [49]. It is a tool that is recommended in the books and papers read during the literature study, for example in [14]. While Metasploit is a tool that can be used to test a wide variety of exploits, Ettercap focuses on only MITM attacks.

Ettercap was used via its graphical interface. The attacker chooses an IP to impersonate and the victims to fool from a list of the available IP addresses on the network.

Result: An ARP spoofing was successfully performed where the packets were forwarded to the intended destination after passing the attacker device. The user could therefore communicate with the speaker unhindered, which gave the attacker the possibility to sniff useful data.

For this situation, Ettercap was better suited than Metasploit. Ettercap performed what was needed and presented it in an easy way. This could have been done in Metasploit as well, by editing the exploit to behave as a MITM attack, but this would have been a more time consuming task.

Analysing traffic

When the ARP Spoofing was successful, the attacker device could impersonate the speaker and see packets destined to the speaker's IP. It is possible to enable HTTPS (Hypertext Transfer Protocol Secure) on the speakers. However, HTTP is used by default and the target device has the default settings. This results in the content being sent in cleartext.

The packets were investigated to try and find exploitable information. The background traffic was studied as well as a simulated scenario where one person impersonated a user performing different use cases. One use case was the user logging in. Here the password was not shown in cleartext, however, some interesting fields to look further into were noticed. It was suspected that this information could be used to crack the password.

4.3.4 Cracking Sniffed Password Hashes

When a user logs in to the speaker via the browser, HTTP packets are sent with the credentials. The speaker uses HTTP with *Digest Access Authentication.* Digest Access Authentication is considered a more secure scheme than Basic Authentication, since the latter sends both username and password in cleartext. For the Digest scheme, a valid response contains a checksum of the username, the password, the given nonce, the HTTP method and the Uniform Resource Identifier (URI). By using this checksum, the user can be authenticated without sending their password in the clear. Even if Digest Authentication is more secure than Basic Authentication, it still has weaknesses. For example, an eavesdropper can look at the header and then try different passwords to produce the right checksum [3].

All the information needed to produce the checksum is included in the HTTP request, except for the password. The values in the HTTP request can be used to crack the password. To do this, a Python script was written that calculated the checksum according to the algorithm used for Digest Access Authentication. The script was used to calculate the checksum for possible passwords from a password list. If the checksum for the correct password was calculated, it would match the checksum in the HTTP request.

Tool: Self-written script
Motivation of choice: The script consists of few rows of code. The script was made by translating the algorithm used for the Digest Access Authentication into

python code. This is a more time efficient solution than searching for an existing tool or script. The password list used in the attack with hydra could be reused here.

The algorithm used in the script:

```
for password in password_list:
    HA1 = MD5(username:realm:password)
    HA2 = MD5(method:digestURI)
    calculated_response = MD5(HA1:nonce:nc:cnonce:qop:HA2)
    if calculated_response == checksum:
        return password
```

Result: When the correct password list is used, the script quickly terminates with the correct result.

This way of cracking a password is faster than using a cracking tool on the login page. Using the script with a password list containing one million words took less than eight seconds. If the same list is used when cracking the password with hydra, the time estimation for the entire list is two hours. The script runs locally and no time is spent sending requests to the target. This method does not leave any footprints of failed attempts to login, meaning there is no risk of getting blocked by the target.

The biggest downside to this method is that the attacker needs to sniff the network to retrieve the needed values. Redirecting the traffic produces a lot of traffic which can be detected by a network administrator. Thus, even if this method leaves no traces of failed attempts to login, there are still many ways to be detected.

Just as for an attack against the login page, the attacker needs to have the correct password list to succeed. If the user has a difficult enough password, the hacker will not be able to crack it within a reasonable time.

4.3.5 Denial-of-Service

A successful DoS attack results in a user that is unable to access devices, information systems, or other networking resources. This is usually achieved by making the target too busy to handle requests from a real user [29]. One use case for the speakers is playing alarms for emergencies. If an attacker blocked user access to the speaker at the time of an emergency, it could result in a life-threatening situation. It was, therefore, important to look closer at DoS attacks.

To perform the attack, the speaker was connected to a LAN together with two computers, see figure 3.7. One computer was used to perform the attack and the other was used to perform use case tasks in the web interface. This attack requires that the attacker have network access to the speaker.

At first, Metasploit was used to performing a Transmission Control Protocol (TCP) SYN flood attack against the speaker. In this attack, the hacker sends a high volume of SYN packets to the target. Each request initiates a connection without ever finalizing it. This forces the target to spend resources on half open connections [8].

Tool: Metasploit, Wireshark for network monitoring
Exploit: `auxiliary/dos/tcp/synflood` on port 80
Result: Initially, the web interface was getting slower and some functions failed. It was not possible to upload new music or to add a planned announcement to the schedule. Eventually, the entire page stopped working and lost connection. The user did not get any responses when trying to ping the speaker. This attack did successfully deny a user from accessing the speaker which could be used by an attacker to prevent legitimate audio from being played. However, the already scheduled music, such as background music, continued to play the entire time. This is because the attack was performed on port 80, which affects the user interface and the traffic going between users and the speaker. To also affect the audio streaming it would be needed to attack the service which is responsible for the audio streaming instead. When the attack stopped, the user was able to access the speaker as usual again.

Another attempt was made using Hping3 to perform a Smurf attack. A Smurf attack is a type of DoS attack where the target system is flooded with spoofed ping messages [11].

Tool: Hping3, Wireshark for network monitoring
Motivation of choice: Hping3 is a command-line oriented tool and is used for security testing but also for stress testing networks [51].
Result: The user was unable to reach the browser login page during this attack. The user got some ping responses during the attack, but the response time was slower than usual. After the attack, the user could access the speaker again.

Other DoS-attacks could be tried as well but the outcome of Dos/DDos attacks is also dependent on the network of which the target speaker is a part of. An interesting test for future work would be to perform a DDoS-attack against a big network of speakers and their leader and look at the outcome.

4.3.6 Buffer Overflow

Buffer overflow occurs when a buffer of a specific size is filled with more data than it can handle [5]. If the system does not prevent it, this big amount of data will overfill parts of the system's storage that is not reserved for it. This can result in parts of the storage that control the execution sequence being overwritten, which give the attacker the possibility to manipulate the system through the storage. The first step of this attack is to find out if the attack is possible, this is done with fuzzing [7].

Fuzzing is a kind of dynamic testing, that is performed by sending random data to a system until it crashes [20]. To detect a possible buffer overflow vulnerability, longer and longer parameters are sent to the target. If the system crashes, it is an indication of a buffer overflow [7].

Existing python scripts were looked for in order to send big payloads to the speaker and thereby overflow the speaker's buffer. One script was used that connected to a port and then entered username and password. The input that was supposed to overflow the buffer was either sent as the username or as the password. The fuzzer tries to send more and more data to the speaker until it crashes because of the overflow. If this happened, the size of the overflow would be known and the attack would be able to move on. During the time fuzzing was performed, the speaker never crashed or went down. Different approaches against both HTTP and Real Time Streaming Protocol (RTSP) were tried.

The speaker continued to function as usual and no odd behaviour was detected. This makes it difficult for an attacker to know if a buffer overflow is possible and how big the buffer is. The next step of a buffer overflow attack requires that the attacker knows the size of the target's buffer.

Another aspect at this stage was to look at existing CVEs to try and find a known exploit that could be applicable to the target. Some CVEs related to the version of RTSP were looked at but none turned out to be relevant. CVEs for similar network embedded system target devices were also studied, and the description of the exploits was read in order to find some that would be relevant for this use case. One of the closest matches found was an exploit targeting the RTSP port on a *HiSilicon Video Encoder* [27]. The script associated with the CVE was tested against the target and it showed that it could successfully connect to the RTSP port however it was not able to disrupt the speaker in any way.

After trying different approaches of buffer overflow without seeing any signs of successfully disrupting the speaker's functionality it was decided to move on in order to have time to test more attacks during the black box phase.

4.3.7 SQL Injection

One of the most known attacks is probably SQL injection, and it is one of the most successful attacks against web applications [31]. An SQL injection is an attack where the hacker can manipulate the SQL queries that an application sends to a back-end database [10].

Method: Manual testing of different usernames and passwords to the web interface.
Motivation of choice: The study about penetration testing tools' effectiveness for finding SQL injections mentioned in section 3.3 showed that experts found twice as many vulnerabilities as the best tool. Also, many of the tools found are only applicable on PHP web pages, which is not the case for the speaker's web interface. It was therefore decided to start looking for SQL injections manually.
Result: A number of different combinations of credentials were tried, but none of them was successful. The web interface did not behave differently compared to "normal" attempts to login. All combinations can be seen in table 4.4. For some usernames, the password will not affect the outcome if the injection works. In these cases, *pass* is used as password.

Table 4.4: Different combinations of SQL injections.

Username	**Password**
1’ OR ’1’ = ’1	1’ OR ’1’ = ’1
root	’ or ’1’=’1
root	" OR "" = "
root	" or "a"="a
" or ""="	" or ""="
root’))/*	*pass*
root"))/*	*pass*
1’ OR ’1’ = ’1’))/*	*pass*
root’–	*pass*
root"–	*pass*
root’ #	*pass*
root" #	*pass*
’ or 1–	*pass*
");	*pass*

4.3.8 Adding a Backdoor

A backdoor is a way of accessing a system without going through the normal authentication process [13]. Adding a backdoor to a system is a way for the attacker to sustain access to the system even if the original way in is removed [16].

After discovering the password the possibility to add a backdoor was investigated. A Metasploit exploit was used to create an SSH backdoor.

Tool: Metasploit
Exploit: `post/linux/manage/sshkey_persistence`
Result: First, an active SSH session was started in Metasploit with the username and password. The exploit was loaded and a new private key file stored locally on the computer was created. Using this file, it is possible to log in to the speaker's SSH port. A backdoor was successfully added.

4.3.9 The Universal Asynchronous Receiver Transmitter

One scenario for attacking the speaker is to physically steal it. When stealing the speaker, it is possible to disassemble the speaker and analyse the different hardware components. A common feature to start looking for is a serial port used by the developers for debugging the device. This debug port is commonly a Universal Asynchronous Receiver Transmitter (UART). An UART port is an interface used for transferring data between two devices [28]. If the UART port is identified, it is possible to directly connect to it with the attacking computer and get a boot loader. The possibility then exists to get shell access to the system.

The UART port on the target device was identified, necessary parts soldered on and a connection was made using a UART to USB cord. Then *picocom* was used, a "minimal dumb-terminal emulation program" [50], to receive a serial port based Linux console. This method resulted in seeing the boot process and then a login prompt appeared.

It was confirmed that the previously obtained login credential is valid. However, it would be useful to find a way to get past the username and password as if the speaker had been stolen. One way of doing this is breaching the login credentials, similar to what had been done earlier, and another way is to find an exploit or a bug that elevates the access past the login prompt. Similar situations were searched for and some projects where others had gained access to IoT devices through the debug port were found.

One frequent way of accessing IoT devices seems to be trying common admin credentials such as admin:admin, root:admin or test:test. Since the correct login credentials are known, it was suspected that none of these would work. Although it is not impossible that the developers have their own developer-login and thus it was decided that it was worth trying some of the most common pairs. The credentials tried can be seen in table 4.5. None of the combinations in the table resulted in a successful login.

Table 4.5: Login credentials tried on debug port

Username	**Password**
admin	admin
root	admin
user	user
test	test
support	support
guest	guest
1234	1234
operator	operator
root	root

One interesting thing noticed while entering different usernames and passwords is the system reacts differently depending on whether the correct username is entered or not. If the username entered is an existing user in the system it responds with `invalid password for '<username>' on /dev/console` but if the username does not match with an existing user the system responds with `invalid password for 'UNKNOWN' on /dev/console`. Since it reads in the user manual that the default username is root, the username was already known but if someone were trying to figure out a valid username to try passwords with, this would be a good clue.

Another solution people have used to gain shell access is bypassing the login by transmitting a newline immediately upon powering the device on. This attack worked on another type of chip compared to the one used in the speaker. However, it is worth trying this trick in case it would also work on the speaker.

This method is performed by pressing the enter key while booting the system. The goal is to transmit a newline immediately upon powering the device and then be granted root access. As expected it did not work on the speaker. It was also tried to press some other special keys to see if any unusual behaviour could be detected. However, this was not successful.

In conclusion, no new way was found to exploit or attack the speaker going through the debug port.

4.4 Results From the Black Box Phase

Table 4.6 summarizes the tools used for each attack during the black box testing and contains insights about the tools from the attacks. Table 4.7 describes insights into the different phases of penetration testing.

4.5 White Box Testing

The next step in the testing was white box testing. This phase was started by examining the security work done by the company regarding the speaker. The company has made threat models for different components of the speaker. The threat models are produced by employees that develop the speaker and contain possible vulnerabilities and countermeasures. In this phase, attacks from the threat models were tried. After examination of the threat models, different attacks were considered. The attack it was decided to focus on was a memory dump attack. Employees at the department that develops the speaker had earlier expressed concern about this attack and its consequences. Because the attack in this phase is based on the company's security work, it can not be described in as much detail as the attacks in the black box testing.

4.5.1 Memory Dump

In this attack, it was possible to extract the memory of a speaker. The memory was dumped into one big file. To be able to interpret the content of the file, different tools were used in order to recreate the file system from the memory. No tool was able to recreate anything useful. It was possible to manually search the file for useful information, but this required knowledge about which information was looked for, and how it would be formatted in the speaker file system.

To be able to dump the memory, the attacker needed to have physical access to the speaker and a firmware image for the device, the firmware image was available on the company's website.

Table 4.6: Insight about tool usage in Exploitation phase

Attack	**Tools**	**Insight**
ARP spoofing	Ettercap, Metasploit, Wireshark	The Metasploit exploits do not have a lot of documentation, the user needs to look at the source code to understand more. To use Metasploit to the full extent, the user should be able to write and edit exploits. This shows the importance of testing different tools for the same attack.
Password cracking	Metasploit, Ncrack, Hydra, Medusa	By trying multiple tools the chance of finding a vulnerability increases. The Lack of error messages complicates the user's understanding of the outcome of the attack.
SQL injection	Manual testing	Most SQL injection tools are made for webpages that use php.
Cracking password hashes	Self-written script, Wireshark	Easier to write scripts than to find suitable tools for calculating checksum
Anonymous FTP	Metasploit, manual testing	Tools and manual testing can be used to complement and confirm each other
Adding a backdoor	Metasploit	Smooth to use previously downloaded tool and not look for a new one
Gaining access via the UART port	Picocom	Difficult to search for tools in an area where very little previous experience exists.
SYN flood	Hping3, Metasploit	Many DoS-tools found are old. Hping is no longer under development but can still be useful.
ICMP flood	Hping3	Same as *SYN flood*
Buffer overflow	Self-written script	For some attacks, it is easier to write scripts than to find suitable tools. With a self-written script, it is possible to adjust the attack after the specific target.

Table 4.7: Insight about the different phases of the penetration testing.

Phase	**Insight**
Reconnaissance phase	- Hard to know which kind of information should be gathered. The real process was not as linear as described in the literature. During the exploitation phase, the information gathering continued. The need for new information was discovered during the reconnaissance phase.
Scanning phase	- Results are dependent on the findings in the Reconnaissance phase. - Short testing phase when only one device is examined compared to a bigger system.
Exploitation phase	- The results of an attack guide the choice of the next attack. - It is important to decide on an ambition level for the attacks in advance to know when to move on from an unsuccessful attack. - Tools are good help but should not be fully relied on. - It was easy to find tools in almost every attack. - It is important to understand both the tools and the targeted part of the system during an attack.

Chapter 5

Evaluation of Security Work

After the penetration testing, the opportunity was given to look at the security work done by the company. The results from the case study could then be compared with the methods used by the company to find overlaps and misses. The company uses a *security development model* inspired by Microsofts Security Development Lifecycle (SDL) [52]. An employee, who has introduced the SDL to the company, informed us about the principles and how the development teams work with SDL in practice. He explained that a lot of the security work is done by the development teams. Finding bugs early in development is cheaper and therefore it is important that the developers are aware of their product's security. An activity that can be performed early in the process is *threat modeling.*

5.1 Threat Modeling

The security of the speaker is evaluated with threat modeling. The models are based on different use cases and are performed on the different components which are used in the speaker and other devices. The severity of the risks found in each threat model is calculated with the Common Vulnerability Scoring System (CVSS) [41]. The development team prioritize to remedy the risks with the highest severity score.

One of the difficult parts of threat modeling is to capture all possible attacks. The threat models are based on use cases and explore situations where an attacker exploits a use case or when a user messes up. Misuse of the system is not covered by the threat models to the same extent as the normal use cases. As an example, adding a backdoor is not covered in the threat model derived during the product development.

The reason why backdoors currently not are included in the threat model is that these are pure use case based and from those backdoor installation has not been identified as a real threat. Hence, this seems to be a thing that has so far been overlooked. It is recommended to consider this in future threat analyses used in the internal product development process.

Table 5.1 below shows which of the penetration test attacks are covered in the currently used threat model and which are not. Table 5.2 shows the same for remaining attacks from the attack tree modeling in chapter 3. The threat models cover almost half of the attacks in table 5.1 and a majority of the attacks in table

Table 5.1: Comparison between the attacks from the penetration testing and the threat models.

Attack	**Covered in threat model**
Anonymous FTP	No
Password cracking	No threat model of web service found
ARP spoofing+cracking password hashes	Yes, sniffing credentials is covered
SYN flood	Yes, DoS attacks are covered
Smurf attack	Yes, DoS attacks are covered
Buffer overflow	Yes, but against other parts of the system than the login service.
SQL injection	No threat model of web service found
Adding a backdoor	No
Bypassing login on UART	No threat model found for this part

5.2. The remaining attacks either belong to a component where the threat model is missing or have not been included in the threat analysis. It should be noted though, that the company's threat models also covered many attacks which were not identified in the penetration testing or the attack tree analysis. These attacks are targeting components of the system which have not been identified or are similar to the attacks performed but more specific to the situation. For example specific ways of using services to perform DoS attacks.

5.2 Vulnerability Management

Another part of the security development model is keeping up with the discovery of new vulnerabilities and exploits. The security of the login service seems to mainly be updated using this method. If threat modeling was performed on this service, password brute forcing would score high on the CVSS scoring which would require actions from the responsible team.

5.3 CVSS Scoring

CVSS is a framework used to calculate a numerical score reflecting the severity and capture the principal characteristics of a vulnerability. The numerical scores go from 0.0 to 10.0 and can be converted to one of these 5 qualitative representations: none, low, medium, high and critical [41].

CVSS scoring is used in the company's security work to get a score on vulnerabilities found. The score is used to prioritizes vulnerabilities. The CVSS scoring connects to the threat modeling since both divide attackers into similar categories, for example network attackers, physical attackers and internet attackers. In the

Table 5.2: Comparison of the attacks from table 3.1 and the threat models.

Attack	**Covered in threat model**
Exploit vulnerability to gain system access	Yes, using known vulnerabilities for components are mentioned.
Retrieve memory from flash/ram	No threat model found for this part
Replay attack	Yes, similar attacks are covered.
Impersonate leader device	Yes, similar attacks are covered.
Impersonate target speaker	Yes, similar attacks are covered.
Inject own code in speaker	Yes
Reset Speaker	No
Turn off speaker	Yes
Social Engineering	Partly mentioned, such as making users click malicious links
Attack on database	Yes

penetration test, the main focus has been on network attacks and physical attacks. Using CVSS as a guide while performing threat modeling can prevent missing a category of attackers.

When scoring a vulnerability in CVSS, one category considered is privileges required for the attack. This is also mentioned in the company's threat modeling. However, since brute forcing the password instantly provided root access, different levels of privileges have not been a big focus during the penetration testing.

Another positive effect of using CVSS was explained to be the CVSS calculator. The calculator can be used to calculate different scores for the same vulnerability, based on the conditions for the vulnerability. Examples of conditions are the complexity of an attack and the attacker's required privileges [40]. A developer can use the calculator to explore which conditions need to change to make the CVSS score decrease for a vulnerability. The CVSS calculator is a graphic and fast way of understanding how to make an attack less severe. It is also possible to look at the scoring of generally known attacks to easier evaluate attacks in the threat modeling. When looking at the performed threat models of speaker components, the severity of the found attacks matches our perception of their severity while performing the penetration test.

A disadvantage mentioned is that CVSS is not the only factor when prioritizing vulnerabilities during threat modeling. A company also has to consider its business and what is relevant to its business model and products. This can explain some things found during penetration testing and why it was possible to perform some attacks. The attack could be captured in a threat model and get a high CVSS score but then get assessed low priority from a business perspective.

One big reason to use CVSS is to achieve a uniform way of scoring attacks and vulnerabilities. Different tools use different ways to set scores. Using CVSS makes the threat models more uniform and easier to perform by the developers. It also makes it easier to understand for someone who has not been a part of the threat modeling.

Chapter 6

Discussion

This section discusses the results obtained during the tests and the different dilemmas and thoughts that were raised during the approach and execution. Since this thesis has three different project goals, the discussion will be divided into three parts: *security vulnerabilities of the speaker*, *evaluation of tools and methods* and *evaluation of the methods used to secure the speaker.*

6.1 Security Vulnerabilities of the Speaker

During the penetration testing, the thing most circled back to was password cracking. After looking at different layers of the speaker and trying to find vulnerabilities, it is clear that there is more than one way to try and brute force the password. After the password was cracked during the Password Cracking, the supervisor at the company looked into why the IP was not blocked from making that many attempts in a short amount of time and found out that customers can turn on password throttling, but it is not a default setting.

Both choosing a secure password and deciding to activate password throttling are up to the customer to do. This is positive in one sense since it provides a lot of flexibility for different customers and their use cases, but it also puts pressure on their awareness of cybersecurity. Here, the designers have made a compromise between security and user friendliness.

6.1.1 Default Settings

The speaker uses HTTP by default, but the user can change the settings so that HTTPS is used instead. Just as for the protection against password guessing, the user needs to make an active decision to use the more secure alternative. Studies show that people are generally more likely to stick with a default option than to actively chose another option. This is called the *default effect* [32]. It is therefore important how options are presented, decision makers can be nudged into choosing an option if that option is a default [39].

By making the settings that improve the speaker's security to default, the customers will be more likely to choose the secure alternative. In the current state of the speaker, even customers that prioritize security over ease of use might still keep the default option because of the default effect. By setting secure options as default, even users that are uninterested in cybersecurity will be nudged into keeping the more secure alternatives. Because of the results from the password cracking, a ticket to make password throttling a default setting has been made at the company.

6.1.2 Password Policy

The difficulty of finding the correct password for the root account on the speaker depended heavily on the security level of the password. Since it was a common word it only took minutes to find while if it had been a longer, more complicated password containing randomly generated letters, numbers and symbols it would have taken more resources and time to calculate. When a new password is chosen the user is informed about the security level of the password, but there is no password policy and the customer is free to choose any password. *Hive Systems* is a company that works with cybersecurity and they have put together a table showing the time it takes a hacker to brute force passwords (in 2022) depending on password length in combination with the usage of lower case letters, upper case letterers, numbers and symbols [42]. The table shows that the weaker passwords can be cracked instantly compared to the more complex passwords which can take 438tn years to brute force.

Our supervisor at the company did some investigation into why the speaker lacked a password policy. The reason seems to be that the password policy should be flexible so it can fit the customer's own security policies. The speakers are not consumer products, they are made to be part of a bigger system with other devices. The company wants the security of the speaker to be flexible enough to be able to fit in whatever system the customer uses.

According to [26], users say that the two most important features of an authentication method are security and memorability. The speaker's current password policy satisfies the user's demand for memorability. If users value security one could imagine that they would choose complex passwords, but studies show that they generally choose easy passwords or use work around, such as writing their password on a post-it next to their computer [18]. Even if users claim to value both security and memorability, most people seem to prioritize memorability. It is therefore naive to develop a product with a poor password policy and then expect the users to choose secure and complex passwords. If a company wants to produce secure products, they need to keep in mind that users often choose the easiest solution, not the most secure.

To force the user to choose more complex passwords is probably a better defense than limiting the number of times a user can enter the wrong password. Even if an attacker would be blocked from performing a password cracking attack, they could find the password in other ways. For example, by sniffing the network for the password hash. If the password is complex enough, it would take the attacker an unreasonable time to crack it, regardless of which approach they choose.

Using more complex passwords is not the only way to improve the security. Another possible improvement would be to use some kind of two-factor authentication. This would prevent an attacker from gaining access just by trying different passwords [18]. Another solution could be to use SSH key authentication. Passwords are easy for the user to remember, but can therefore also be easy for an attacker to crack. By instead using longer keys that are not chosen by the user, the possibility for an attacker to brute force the password in a reasonable time disappears.

6.1.3 Network Setup - Factors Beyond the Speaker

During the testing, a lot of attacks were performed that required access to the network. It was also needed to perform some network related attacks that attacked a part of the network instead of only the speaker, for example ARP poising. The result of these attacks will depend on the setup of the user's network. How the user sets up their network will affect how difficult it would be for a hacker to attack the speaker. In our testing, a position has been assumed where access to the network to which the speaker is connected already has been gained. A major way of protecting the speakers is to have a high level of security on the network. This increases the difficulty of accessing the speakers. Even if an attacker physically steals a speaker and manages to obtain the login credentials it will not be especially useful as long as they do not have network access and can reach the other devices in the network.

In real life situations, the speakers are often part of a bigger network, containing many other network embedded devices and there are security policies put into place which affect all devices including the speakers. This is also a factor not considered in our testing. A proposal for future work is to either pen test an already existing network with speakers or to set up a new, bigger infrastructure of speakers. Then it would be possible to test a scenario closer to a real life situation.

6.2 Evaluation of Tools and Methods

When looking at tools and methods there are different areas to assess. The first one is finding good and trustworthy tools and methods to use. The second area is the actual usage of the tools and methods. The third area is the positive and negative effects of tool usage.

6.2.1 Finding Tools and Methods

In the literature phase of the thesis, a number of books about penetration testing written by professionals were read [14] [15] [12] [19]. In our quest to find good tools and methods we have based our choices on the recommendations from these books. The authors have long experience in using penetration testing tools in real life situations and basing both the general penetration testing methodology as well as tools and methods for specific attacks was a good decision. It provided a foundation to base decisions on.

When no useful tool could be found within the literature study sources it was still easy to find websites that describe different popular tools for the purpose.

Another choice made was buying commercial tools or using only open source software. Considering the scope of this project only open source tools have been used. This decision also helps with understanding how a tool works since it is easier to understand how a tool functions with access to the code behind it. It was not a problem finding tools when feeling the need to use tools. There were always a lot of different tools available, the difficult part of finding tools was to sort out what tools are good and will work for our use case.

6.2.2 Using Tools and Methods

One of the biggest struggles during this phase was when a tool was used to perform a specific attack and the attack did not work, it was then needed to figure out if it was due to the tool being used wrong or if the security of the target correctly prevents the attack. To handle this issue time was spent understanding the tool and the part of the system being attacked. As an example, one of the first things that were decided to exploit was password brute forcing. There are many tools for this purpose and we started by looking at the most popular ones. However, when running these different password brute forcing tools, a couple of different results were obtained. Some tools did not proceed to run at all, one tool thought every tried password was correct and another one that no one was correct. If only the tool that claimed no password was correct was tried, we might have thought that the correct password did not exist in the tried password file. Since many different tools were tried and got varying results it was understood that the problem could also be with how the password was parsed by the web service. After figuring that out we managed to crack the password using the same password files as originally tried, showing that the problem depended on the limitations of the tools as well as us not using the correct settings of the tools. It was tried to follow this method of using different tools as well as understanding the theory behind them during all of the black box testing but this method also has limitations. If no successful outcome has been seen after trying different approaches, a decision has to be made on whether to continue trying to exploit that part of the system or if it is time to move on to another possible vulnerability.

As described earlier, the approach to using tools was to use multiple tools for the same attack and to try attacks manually when it was suited. For some attacks, it was reasonable to move on after two attempts, but others require more. It is hard to decide on one exact rule that will work for all attacks because the size of attacks can vary a lot. For example, an anonymous FTP attack can not be performed in many different ways. For this attack, it is arguable to move on from the attack after two different attempts. For other attacks, such as buffer overflow, the attack can be performed in more different ways and it is therefore reasonable to test more before moving on.

6.2.3 How Tools and Methods Affect the Penetration Testing

One present factor during the black box testing was the relation to the usage of tools. Tools can be a quick and easy way to find and exploit vulnerabilities, however, it is also important to understand how the tool functions while using it. Not understanding the structure of the tool provides the risk of misusing the tool and missing vulnerabilities. It is also important to understand how different tools work while choosing which one to use and whether using a tool is actually needed to perform the attack.

The choice of not using commercial tools affects the outcome of our testing since the methods and tools used affected the result of the attacks.

6.3 Evaluation of the Methods used to Secure the Speaker

In this thesis, penetration testing has been compared to the threat modeling done by the company to better secure their products.

Threat modeling is performed by people with knowledge about the system they are trying to protect. They know which parts the system consists of, and they use their knowledge to try and find as many vulnerabilities as possible. This provides an advantage over an outside attacker. The outside attacker has the advantage of only needing to find one vulnerability, while the defender, theoretically, would need to find all vulnerabilities to make their system completely secure.

The company uses CVSS to score vulnerabilities in threat models. The CVSS score has both advantages and disadvantages. It is a good way of producing a score that can be used to prioritize which vulnerabilities to patch first. It is also a uniform measurement of how to score severity, that can be used for communication with customers. A disadvantage of the CVSS score is that it does not consider how important or sensitive a product is. A vulnerability in a very important product can get the same score as a vulnerability in an insignificant product. The CVSS score does not consider the business value of a product. This prioritization needs to be done by the user without help from CVSS.

A black box penetration test is done without the tester knowing any inside information about the system. This can lead to the pentester spending time on exploiting the wrong part of the system, but can also be a good way of understanding an outsider's perspective. A pentester's goal is to find one way into the system, and a penetration test might therefore find a smaller quantity of vulnerabilities compared to a threat model.

Pentesting is not, and should not, be the only method used to secure a system. Everything included in the threat models was not captured by the penetration testing. On the contrary, the threat models covered the majority of the vulnerabilities mentioned in this thesis, as well as additional attacks against the system.

An advantage that pentesting provides here is the possibility of finding examples of misuse that can be hard to cover in the threat models. Pentesting is also a good way of evaluating whether the security actually works as intended. Both pentesting and vulnerability management need to be performed after the system is done, and is therefore expensive. Threat modeling can be performed before and during the time a system is developed, and it is therefore cheaper to fix the found risks.

The part we focused on the most during the testing was the login service, a part of the system without a threat model. The security would likely improve if a threat model was performed on this component too.

6.4 Contribution to the Field

Even if the results from the exploitation phase of this thesis are specific to a certain device, the way of working during the thesis can be applied to other situations as well. None of the attacks tested during the thesis are only applicable to the specific speaker examined. The majority of the attacks can be performed against other kinds of targets as well, such as web pages. The same goes for our approach to using penetration testing tools.

The conclusions about passwords and default settings are also applicable for other cases. The importance of a complex password is universal. All companies need to consider that their users in most cases will choose a password that is easy to remember over a safe password.

During the literature study a lot of research was done about penetration testing. The majority of the literature found was about penetration testing against websites or entire organizations. There was not as much about penetration testing firmware. This thesis will therefore contribute knowledge about firmware testing, and which kind of attacks can be used on a network device.

It was decided to not try any attacks to breach into a costumers network because it was determined that it was outside the scope. If anyone wanted to use this thesis as a guide to penetration test a network device that is used actively by a customer, they would need to look for that information elsewhere. The same goes for social engineering.

6.5 Limitations

A lot of limitations with the findings and conclusions relate to the specific case study. One limitation is that the testing is performed on a device that does not belong to a bigger network of speakers. In a setup closer to a real life situations the speaker would be a part of a network with other speakers and occasionally other network embedded devices. Performing the testing on a scenario closer to an existing setup would yield more accurate results.

When evaluating the security of an implementation the goal is to find all possible threats and vulnerabilities. However, within penetration testing, the goal is often to find one way to breach the system, not all possible ways. Different approaches have been tried but the nature of penetration testing causes difficulty in evaluating the overall security of a system.

Part of the testing was performed on the company's internal lab-network used by developers to test devices. Since this network is used for testing, it is likely set up differently compared to a customer's network. Some attacks performed over the network could be stopped or detected by security features on the network, while in our case it was not as necessary to consider.

The device was set up to use the default settings. This is another limitation since the customers/installers can and hopefully will change the settings. The device in our case study used HTTP instead of HTTPS, had no limit for password guessing and had a weak root password. The results from the black box phase would differ if a more secure device was tested instead.

6.6 Conclusions

This thesis shows the importance of considering the work and methods of experienced penetration testers while simultaneously adapting the testing to a specific system or scenario. The already existing methods to follow provide a stable foundation and guideline throughout the process. Following this reduces the risk of missing vulnerabilities as well as provides structure to the testing. However, it is also shown that it is important to be adaptable and creative in every step to find new vulnerabilities.

The information gathering done at the beginning of penetration testing is an important stage, it affects the ambition levels of the testing and the outcome of the exploitation phase. However, gathering information was one of the most difficult tasks to perform as a beginner at penetration testing since it was hard to sort out useful information prior to performing attacks.

During this thesis, multiple times an attack tried have failed with one tool but succeeded with another tool. It is therefore important to try multiple tools for the same attack. A penetration tester should never blindly trust a tool. The tester needs to understand how a tool works to be able to understand why it fails and what needs to change to perform the attack correctly. Using automated tools is not always the easiest way to perform an attack. In some cases, it is more effective to use a self-written script adjusted for the specific situation or to test the attack manually.

Because of the big variety of possible tools to use, it is possible to conduct a penetration test using only free open-source tools. This approach makes it possible for the tester to examine the source code of the tool, which is a way of understanding how it works and what it does.

Penetration testing should not be the only method used to secure a system. It is a good complement to threat modeling, as it can find vulnerabilities that are hard to fit in threat models. Threat modeling can cover more vulnerabilities because it is performed by people with knowledge about the system they are securing.

Chapter 7 Future Work

There are many possible ways to extend the work done in this thesis. One example of future work could be to perform a penetration test on an existing customer's setup of network embedded systems, including for example the speaker used in this case study or a similar product. This would include breaching into the customer's network, using social engineering to gain information and stealing a device.

The kind of speaker that was the target in this thesis is often part of a large network of other speakers. Future work could examine the risks when an attacker takes control of a large network of connected devices.

This thesis focus on free, open-source tools. Future work could focus on evaluating commercial tools, to see if the same conclusions can be drawn as for open-source tools.

Bibliography

[1] P. Deutsch, A. Emtage, and A. Marine, *How to Use Anonymous FTP*, RFC 1635, May 1994. DOI: `10.17487/RFC1635`. [Online]. Available: `https://www.rfc-editor.org/info/rfc1635`.

[2] B. Schneier. "Attack trees." (1999), [Online]. Available: `https://www.schneier.com/academic/archives/1999/12/attack_trees.html` (visited on 04/20/2022).

[3] P. J. Franks, P. Hallam-Baker, L. C. Stewart, *et al.*, *HTTP Authentication: Basic and Digest Access Authentication*, RFC 2617, Jun. 1999. DOI: `10.17487/RFC2617`. [Online]. Available: `https://www.rfc-editor.org/info/rfc2617`.

[4] Network Working Group. "Sip: Session initiation protocol." (2002), [Online]. Available: `https://www.ietf.org/rfc/rfc3261.txt` (visited on 04/13/2022).

[5] J. C. Foster, V. Osipov, N. Bhalla, N. Heinen, and D. Aitel, "Chapter 1 - buffer overflows: The essentials," in *Buffer Overflow Attacks*, J. C. Foster, V. Osipov, N. Bhalla, N. Heinen, and D. Aitel, Eds., Burlington: Syngress, 2005, pp. 3–23, ISBN: 978-1-932266-67-2. DOI: `https://doi.org/10.1016/B978-193226667-2/50037-2`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/B9781932266672500372`.

[6] V. Ramachandran and S. Nandi, "Detecting arp spoofing: An active technique," in *International conference on information systems security*, Springer, 2005, pp. 239–250.

[7] M. Osborne, "Chapter 13 - application security flaws and application testing," in *How to Cheat at Managing Information Security*, ser. How to Cheat, M. Osborne, Ed., Burlington: Syngress, 2006, pp. 281–302, ISBN: 978-1-59749-110-5. DOI: `https://doi.org/10.1016/B978-159749110-5/50020-8`. [Online]. Available:

https://www.sciencedirect.com/science/article/pii/B9781597491105500208.

[8] W. Eddy, *TCP SYN Flooding Attacks and Common Mitigations*, RFC 4987, Aug. 2007. DOI: 10.17487/RFC4987. [Online]. Available: https://www.rfc-editor.org/info/rfc4987.

[9] N. Antunes and M. Vieira, "Comparing the effectiveness of penetration testing and static code analysis on the detection of sql injection vulnerabilities in web services," in *2009 15th IEEE Pacific Rim International Symposium on Dependable Computing*, 2009, pp. 301–306. DOI: 10.1109/PRDC.2009.54.

[10] J. Clarke, "Chapter 1 - what is sql injection?" In *SQL Injection Attacks and Defense*, J. Clarke, Ed., Boston: Syngress, 2009, pp. 1–27, ISBN: 978-1-59749-424-3. DOI: https://doi.org/10.1016/B978-1-59749-424-3.00001-3. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9781597494243000013.

[11] G. R. Zargar and P. Kabiri, "Identification of effective network features to detect smurf attacks," in *2009 IEEE Student Conference on Research and Development (SCOReD)*, 2009, pp. 49–52. DOI: 10.1109/SCORED.2009.5443345.

[12] D. Maynor, *Metasploit toolkit for penetration testing, exploit development, and vulnerability research.* Elsevier, 2011.

[13] T. Wilhelm and J. Andress, "Chapter 16 - sabotage," in *Ninja Hacking*, T. Wilhelm and J. Andress, Eds., Boston: Syngress, 2011, pp. 267–284, ISBN: 978-1-59749-588-2. DOI: https://doi.org/10.1016/B978-1-59749-588-2.00016-0. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9781597495882000160.

[14] P. Engebretson, *The basics of hacking and penetration testing: ethical hacking and penetration testing made easy.* Elsevier, 2013.

[15] T. Wilhelm, *Professional penetration testing: Creating and learning in a hacking lab.* Newnes, 2013.

[16] J. Andress and S. Winterfeld, "Chapter 6 - logical weapons," in *Cyber Warfare (Second Edition)*, J. Andress and S. Winterfeld, Eds., Second Edition, Boston: Syngress, 2014, pp. 103–136, ISBN: 978-0-12-416672-1. DOI: https://doi.org/10.1016/B978-0-12-416672-1.00006-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780124166721000064.

[17] G. Watson, "Chapter 1 - an introduction to social engineering," in *Social Engineering Penetration Testing*, G. Watson, A. Mason, and R. Ackroyd, Eds., Boston: Syngress, 2014, pp. 1–17, ISBN: 978-0-12-420124-8. DOI: https://doi.org/10.1016/B978-0-12-420124-8.00001-6. [Online]. Available: https://www.sciencedirect.com/science/article/pii/B9780124201248000016.

[18] C. Everett, "Are passwords finally dying?" *Network Security*, vol. 2016, no. 2, pp. 10–14, 2016, ISSN: 1353-4858. DOI: https://doi.org/10.1016/S1353-4858(16)30017-4. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1353485816300174.

[19] J. Faircloth, *Penetration tester's open source toolkit.* Syngress, 2016.

[20] M. Felderer, M. Büchler, M. Johns, A. D. Brucker, R. Breu, and A. Pretschner, "Chapter one - security testing: A survey," in ser. Advances in Computers, A. Memon, Ed., vol. 101, Elsevier, 2016, pp. 1–51. DOI: https://doi.org/10.1016/bs.adcom.2015.11.003. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0065245815000649.

[21] K. Shaukat, A. Faisal, R. Masood, A. Usman, and U. Shaukat, "Security quality assurance through penetration testing," in *2016 19th International Multi-Topic Conference (INMIC)*, IEEE, 2016, pp. 1–6.

[22] kennyn510. "Wpa2-wordlists." (2017), [Online]. Available: https://github.com/kennyn510/wpa2-wordlists (visited on 03/08/2022).

[23] D. Silnov, A. Prokofiev, G. Berezovskaya, V. Perevozchikov, S. Troitskiy, and I. Shumakov, "A method of detecting a malicious actions using http and ftp protocols," in *2017 Intelligent Systems Conference (IntelliSys)*, IEEE, 2017, pp. 1083–1088.

[24] I. Del Pozo, M. Iturralde, and F. Restrepo, "Social engineering: Application of psychology to information security," in *2018 6th International Conference on Future Internet of Things and Cloud Workshops (FiCloudW)*, IEEE, 2018, pp. 108–114.

[25] H. M. Z. A. Shebli and B. D. Beheshti, "A study on penetration testing process and tools," in *2018 IEEE Long Island Systems, Applications and Technology Conference (LISAT)*, 2018, pp. 1–7. DOI: 10.1109/LISAT.2018.8378035.

[26] R. Alomari and J. Thorpe, “On password behaviours and attitudes in different populations,” *Journal of Information Security and Applications*, vol. 45, pp. 79–89, 2019, ISSN: 2214-2126. DOI: `https://doi.org/10.1016/j.jisa.2018.12.008`. [Online]. Available: `https://www.sciencedirect.com/science/article/pii/S2214212618305027`.

[27] Alexei Kojenov. “Hisilicon video encoders - unauthenticated rtsp buffer overflow (dos).” (2020), [Online]. Available: `https://www.exploit-db.com/exploits/48903` (visited on 05/05/2022).

[28] S. Harutyunyan, T. Kaplanyan, A. Kirakosyan, and A. Momjyan, “Design and verification of autoconfigurable uart controller,” in *2020 IEEE 40th International Conference on Electronics and Nanotechnology (ELNANO)*, 2020, pp. 347–350. DOI: `10.1109/ELNANO50318.2020.9088789`.

[29] W. Bonasera, M. M. Chowdhury, and S. Latif, “Denial of service: A growing underrated threat,” in *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, 2021, pp. 1–6. DOI: `10.1109/ICECCME52200.2021.9591062`.

[30] danielmiessler. “Seclists.” (2021), [Online]. Available: `https://github.com/danielmiessler/SecLists` (visited on 03/03/2022).

[31] B. Gogoi, T. Ahmed, and A. Dutta, “Defending against sql injection attacks in web applications using machine learning and natural language processing,” in *2021 IEEE 18th India Council International Conference (INDICON)*, 2021, pp. 1–6. DOI: `10.1109/INDICON52576.2021.9691740`.

[32] C. K. Johnson, R. S. Gutzwiller, J. Gervais, and K. J. Ferguson-Walter, “Decision-making biases and cyber attackers,” in *2021 36th IEEE/ACM International Conference on Automated Software Engineering Workshops (ASEW)*, 2021, pp. 140–144. DOI: `10.1109/ASEW52652.2021.00038`.

[33] R. Petrovic, D. Simic, S. Stankovic, and M. Peric, “Man-in-the-middle attack based on arp spoofing in iot educational platform.,” *2021 15th International Conference on Advanced Technologies, Systems and Services in Telecommunications (TELSIKS), Advanced Technologies, Systems and Services in Telecommunications (TELSIKS), 2021 15th International Conference on*, pp. 307–310, 2021, ISSN: 978-1-6654-4442-2. [Online]. Available: `http://ludwig.lub.lu.se/login?url=https://search.`

ebscohost.com/login.aspx?direct=true&AuthType=ip,uid&db=edseee&AN=edseee.9606392&site=eds-live&scope=site.

[34] S. Vandervelden, M. M. Chowdhury, and S. Latif, "Managing the cyber world: Hacker edition," in *2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, IEEE, 2021, pp. 1–6.

[35] Wikipedia. "Black-box testing." (2021), [Online]. Available: https://en.wikipedia.org/wiki/Black-box_testing (visited on 02/24/2022).

[36] ——, "White-box testing." (2021), [Online]. Available: https://en.wikipedia.org/wiki/White-box_testing (visited on 11/22/2021).

[37] Cloudflare. "Introducing flan scan: Cloudflare's lightweight network vulnerability scanner." (2022), [Online]. Available: https://blog.cloudflare.com/introducing-flan-scan/ (visited on 01/28/2022).

[38] cve.org. "Frequently asked questions (faqs)." (2022), [Online]. Available: https://www.cve.org/ResourcesSupport/FAQs (visited on 04/20/2022).

[39] G. Elisa, Z. Micaela Maria, N. Raffaella, and G. Fiorella, "Default rules in investment decision-making: Trait anxiety and decision-making styles.," *Financial Innovation*, vol. 8, no. 1, pp. 1–26, 2022, ISSN: 2199-4730. [Online]. Available: http://ludwig.lub.lu.se/login?url=https://search.ebscohost.com/login.aspx?direct=true&AuthType=ip,uid&db=edsdoj&AN=edsdoj.3657f72c73654c64a23134df9424a916&site=eds-live&scope=site.

[40] FIRST.org. "Common vulnerability scoring system version 3.1 calculator." (2022), [Online]. Available: https://www.first.org/cvss/calculator/3.1 (visited on 05/24/2022).

[41] ——, "Common vulnerability scoring system version 3.1: User guide." (2022), [Online]. Available: https://www.first.org/cvss/user-guide (visited on 01/28/2022).

[42] Hive Systems. "Are your passwords in the green?" (2022), [Online]. Available: https://www.hivesystems.io/blog/are-your-passwords-in-the-green?utm_source (visited on 03/07/2022).

[43] OWASP. “Blocking brute force attacks.” (2022), [Online]. Available: https://owasp.org/www-community/controls/Blocking_Brute_Force_Attacks (visited on 02/08/2022).

[44] Wikipedia. “Case study.” (2022), [Online]. Available: https://en.wikipedia.org/wiki/Case_study (visited on 01/27/2022).

[45] ——, “Dictionary attack.” (2022), [Online]. Available: https://en.wikipedia.org/wiki/Dictionary_attack (visited on 02/08/2022).

[46] ——, “Dual-tone multi-frequency signaling.” (2022), [Online]. Available: https://en.wikipedia.org/wiki/Dual-tone_multi-frequency_signaling (visited on 04/13/2022).

[47] ——, “Power over ethernet.” (2022), [Online]. Available: https://en.wikipedia.org/wiki/Power_over_Ethernet (visited on 04/13/2022).

[48] Wireshark.org. “About wireshark.” (2022), [Online]. Available: https://www.wireshark.org/ (visited on 02/11/2022).

[49] ettercap-project.org. “Ettercap.” (n.d.), [Online]. Available: https://www.ettercap-project.org/ (visited on 02/21/2022).

[50] gnu.org. “Picocom 3.1.” (n.d.), [Online]. Available: https://guix.gnu.org/packages/picocom-3.1/ (visited on 03/25/2022).

[51] hping. “Hping.” (n.d.), [Online]. Available: http://www.hping.org/ (visited on 03/23/2022).

[52] Microsoft. “What are the microsoft sdl practices?” (n.d.), [Online]. Available: https://www.microsoft.com/en-us/securityengineering/sdl/practices (visited on 05/12/2022).

[53] Nmap.org. “Nmap security scanner.” (n.d.), [Online]. Available: https://nmap.org/ (visited on 01/27/2022).

[54] ——, “Port scanning basics.” (n.d.), [Online]. Available: https://nmap.org/book/man-port-scanning-basics.html (visited on 01/27/2022).

[55] Rapid7. “Metasploit.” (n.d.), [Online]. Available: https://www.rapid7.com/products/metasploit/ (visited on 02/21/2022).

Printed by Tryckeriet i E-huset, Lund 2022

Series of Master's theses
Department of Electrical and Information Technology
LU/LTH-EIT 2022-868
http://www.eit.lth.se