Reverse Engineering Labs

Studienarbeit

Department for Computer Science
OST - Ostschweizer Fachhochschule
Campus Rapperswil-Jona

Semester: Autumn 2022

Autors: Gianluca Nenz
Ronny Mueller
Thomas Kleb

Project Advisor: Ivan Buetler
Release: E-Prints
Version: Tuesday 20th December, 2022

Department for Computer Science
OST Eastern Switzerland University of Applied Sciences

Abstract

Background As it stands today there is no module or learning unit at the OST to teach the concept of software reverse engineering.

Purpose Because reverse engineering is an important topic in the cybersecurity space, the goal of this SA is to create challenges in the topic of reverse engineering. These challenges will or can be used by lecturers of the OST in Rapperswil-Jona to teach the basics to the students.

Methods To achieve this goal we created challenges in ascending difficulty. At first the students will get information about the software they will use and the overall strategies of reverse engineering. After that they learn about some more advanced concepts. The focus here is creating the challenges in a manner, which should teach the basics in a easy to understand fashion. These challenges will be hosted on the Hacking-Lab platform. To ensure the quality of our challenges we did organize testing participants, who are cybersecurity students in their fifth semester as well. These tests should be the indicator if the goal was reached.

Results The primary goal was to create a total of up to 10 challenges in this SA. This goal was achieved with a total of 11 created challenges. The testing was also successfully carried out and the feedback overall positive. If there was some common feedback, we adjusted the corresponding challenge based on it. The section 4.1.4 covers this in detail.

Conclusions This project was overall very successful, but there has also to be said that the topic of reverse engineering is huge and only a tiny portion of it was covered in these challenges. There are many more techniques and tools which have not been covered yet and this could be a future work.

Contents

Glossary

Acronyms

Acronym	Description
AES	Advanced Encryption Standard
ASLR	Address space layout randomization
ASM	Assembly Language
DEP	Data execution prevention
ECB	Electronic Code Book version of AES
ECTS	European Credit Transfer System
GDB	GNU Project Debugger
GUI	Graphical user interface
IDA	Interactive Disassembler
OST	Ostschweizer Fachhochschule
RUP	Rational Unified Process
SEP	Software Engineering Practices

Terms

Term	Description
Binary	A pre-compiled, pre-linked program that is ready to run under a given operating system; a binary for one operating system will not run on a different operating system
Docker	Open source platform for building, deploying, and managing containerized applications
Domain	Everything that is needed for reverse engineering
Flag	A flag is a string of text which needs to be entered into the website to show that you have solved the lab
Ghidra	Free to use and developed by the NSA decompiling tool
GitLab	A versioning platform similar to GitHub but hosted on an OST server
Hopper	Payed tool which allows disassembling, decompiling and debugging
IDA Freeware	Free version of IDA with same base functionalities
Proof of concept	Evidence obtained froma a pilot project, that the idea is feasible
Radare2	Framework for reverse engineering usable from the command line

Chapter 1

Project Idea

1.1 Task

The goal of this project is to create a reverse engineering lab. The students who take part in the labs should learn to reverse step by step and get comfortable with tools like IDA Freeware, Radare2, Hopper or Ghidra. The labs should primarily be completed on either a Windows or a Linux system.
After a first analysis, the requirements for the reverse engineering labs and the exercises should be formulated. Afterwards, the students should create 8 - 10 engineering labs. It is important that there is a common thread through all the labs. The participating students should start with easier examples and slowly solve more complex tasks. ASLR and DEP should be a part of the labs.

1.2 Problem Domain

To have an easy overview of the lab structure, a problem domain was created (see figure 1.1) at the beginning of the project. This ensured the project's goal is reached at the end. In addition to that, this mindmap is displayed in each of the labs to show the participating student which part of reverse engineering is taught.

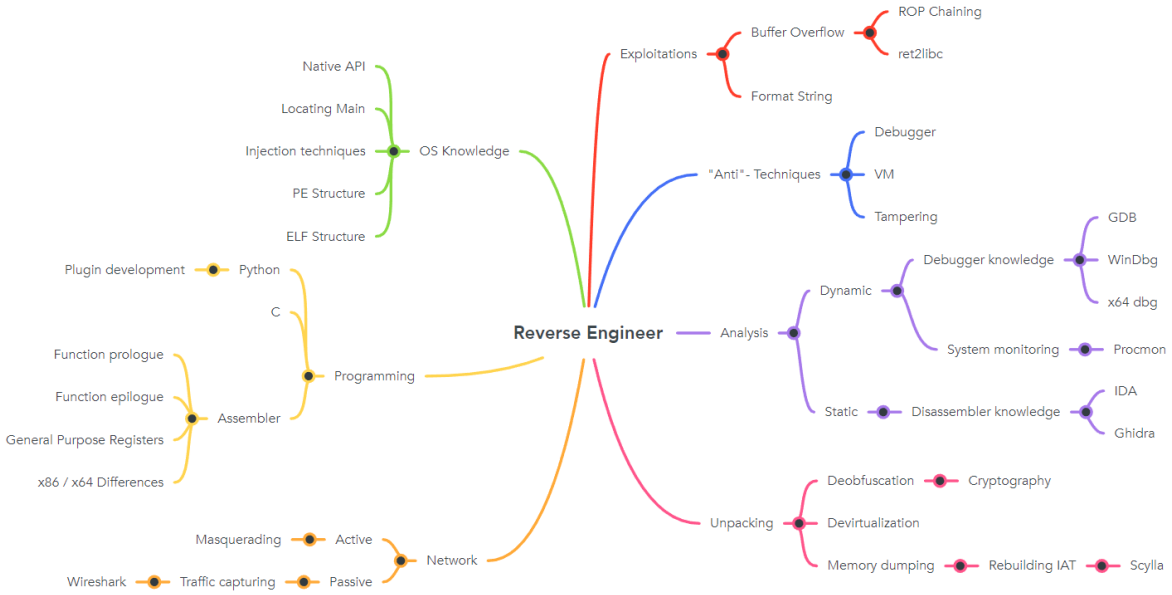

Figure 1.1: Mindmap of the knowledge a reverse engineer needs.

1.3 Learning Concepts

Before each lab is created, a proof of concept (POC) has to be defined. These concepts contain the information about which part of the problem domain in figure 1.1 is taught, how the lab is structured and what the teacher has to explain beforehand. The lab names and content were defined in a way that the student solving them has a clear thread to follow and goes from easier to harder exercises.
To assure the labs could be completed as planned it is assumed that the students have basic knowledge in assembly language (ASM), Python and C. Since ASM is the most important language to understand the fundamental structure of a binary, a lab for refreshing this knowledge is planned.
The lab titles and content changed over the time of the project and the final scope of the labs differs from the one at the beginning. The final list is shown in table 1.1.

Topic	Description
Tutorial for the tools: GDB	Introduction into GDB and its tools
Tutorial for the tools: x64dbg	Introduction into x64dbg and how to navigate the GUI
Tutorial for the tools: IDA Freeware	Introduction into IDA Freeware and how to navigate the GUI
Refresher	Give the students some little refreshing on the key topics (Assembly)
Static Debugging	Given a simple C file, students analyse it and try to find a key (Find Main function)
Dynamic Debugging	Given a simple C file, students analyse it and try to find a key (go more into GDB / x64)
First RE attempts	Given simple files compiled in different compilers to find a key
Remote Login	Inspecting binary locally and using RE to gain access to a remote docker-container
Pwntools	Exploiting a vulnerability found through RE with pwntools
AES Encryption	Not only finding out the password but writing a keygen for the program
Patching a Binary	Introduce new native API funcs / techniques like stack strings

Table 1.1: Overview of all the Labs.

Chapter 2

Management Summary

2.1 Overview

2.1.1 What is Reverse Engineering

Reverse engineering is the process of analyzing a product or system in order to understand how it works, how it was made, or how it can be improved. It involves taking apart the product or system, examining its components, and understanding how they fit together and interact with each other.
In the context of software, reverse engineering is the process of analyzing a computer program in order to understand how it works and how it was implemented. This may involve disassembling the program, studying its code and documentation to recreate or modify it. Reverse engineering can be driven by various motivations: learning about new technologies, fixing flaws / security vulnerabilities or creating competing products.
Most of the time, reverse engineering is a challenging and time-consuming process, as it requires a deep understanding of the underlying technologies and systems. It is often used by experts in fields such as computer science, engineering, and security.

2.1.2 Current Situation

For a computer scientist, it is always useful to have knowledge in cybersecurity subjects. To accomplish the task of showing the students the world of cybersecurity, the Ostschweizer Fachhochschule (OST) implemented several modules like "Cyber Security", "Secure Software" and the newest one "Cyber Defense". In these modules, the professors explained the different aspects using Hacking-Lab as platform for the exercises. The plan is to extend the current state with reverse engineering labs and exercises. The goal of them is to bring the students closer to this subject and explain the fundamentals of reverse engineering.

2.2 Approach

To achieve these tasks, new exercises will be added to Hacking-Lab OST environment, which is, as mentioned above, a platform with which students are already accustomed to. These exercises will be added in the form of challenges for the student to go through and will be built with the idea of future additions in mind.

2.3 Procedure

At the start of the project, the scope was defined. The defined scope was the base on which the labs are created. This scope includes the difficulty increase between each lab, the know how to be taught to understand the procedure and which tools are used by the student to finish the tasks. In addition to these points, a platform on which the student is intended to work on is defined. For the students to solve the given tasks, they needed a provided infrastructure to follow.

2.4 Technologies

The labs are created using either the Windows or Linux operating system, depending on the software needed. This allows the solving student to have the option to complete each lab on one of the two systems using a virtual machine if needed. All the labs are hosted on a Hacking-Lab tenant provided by the advisor, first on demo to test all the features and how to set them up, then on the OST tenant for official use.
Hacking-Lab is a website that offers a range of cybersecurity-related services, including training, simulations, and challenges. It is designed for professionals in the cybersecurity field, as well as students and enthusiasts who are interested in learning about and improving their skills in this area. Because of this, the OST uses it to host different exercises to teach the fundamentals of cybersecurity to its interested students.

2.5 Results

2.5.1 Goals Achieved

The goals were defined by the advisor in section 1.1. It was planned to have 8 - 10 labs finished until the end of week 12. This goal was reached thanks to a strict plan and coordination between the students.

2.5.2 Goals Not Achieved

Some goals were redefined during the iterative process of the project, but in the end, all of them were reached successfully. During the project, 11 concepts were defined and uploaded to Hacking-Lab.

2.6 Future

The labs created in this project are a base for future labs and should show participating students the initial steps of reverse engineering. The students plan to further add to the labs in the bachelor thesis, together with going into more advanced subjects and more complicated exercises.

Chapter 3

Technical Report

3.1 Introduction

3.1.1 Problem

The subject of cybersecurity is constantly growing in importance for the computer scientist in general. The demand for cybersecurity experts with broad knowledge regarding current problems and malware is ever growing [1]. The Ostschweizer Fachhochschule (OST) has recognized this demand and has added more and more lectures for the cybersecurity interested students [2]. One important aspect is still missing in the curriculum: reverse engineering.

3.1.2 Similar Work

To this date there is no work done regarding this subject in past projects. In the past, students have created different labs for the OST but none for reverse engineering. The infrastructure which is used for this project is already existing and established in the OST lectures. Hacking-Lab as the platform is known by the targeted student group and teachers alike.

3.1.3 Technologies Used

To create each of the labs and their documentation, multiple different tools and languages were used (shown in table 3.1 and 3.2).

Languages	
Assembler	As the base structure of a binary, it was taught in multiple courses before this. In the labs, ASM is used to understand the flow of a function and what it does when executed.
C	All of the binaries were written in C.
Python	As an easy-to-understand language, Python is used to write the exploits after analyzing the binaries.

Table 3.1: Overview of all the languages used to create the labs.

Tools	
VSCode	Each of the students of the project used VSCode for programming and documenting. This allowed for easier settings and more control of the output.
IDA Freeware	Each of the labs were tested before uploading to Hacking-Lab. All the tests were done in IDA Freeware since this software is used to show the solutions.
Ghidra	Each of the labs were tested before uploading to Hacking-Lab. Ghidra was used to check the pseudocode. This ensured that students using Ghidra instead of IDA Freeware have a solvable problem as well and can follow the steps given.
HL Demo Tenant	To test all of the labs, the demo tenant of Hacking-Lab was used. This allowed for free testing without interfering with the OST tenant.
Docker	A Docker container can be used on the Hacking-Lab platform to have a server side component to the challenges.
OST GitLab	To have versioning of the code OSTs GitLab was used.
Clockify	This software allowed for time management.

Table 3.2: Overview of all the frameworks and tools used to create the labs.

3.1.4 Goals

The goal for this project has multiple facets:

- The creation of different labs to show the students of the OST the aspects of reverse engineering
- The students should have the following learning objectives:
  - Gain an understanding of what reverse engineering is and what it can be used for
  - Know the basic handling of debuggers and disassemblers
  - Understand a binary programs control flow using static debugging
  - Understand a binary programs control flow using dynamic debugging
  - Be able to locate and exploit simple security flaws in a binary program.

3.1.5 Setup

Since every student uses his own machine with his own configurations, Hacking-Lab as a platform with its LiveCD was chosen as the foundation. This allows a simple, operating system independent approach to all excercises thanks to its webinterface and docker hosting capabilities.

3.2 Requirements for the Labs

3.2.1 Overview

This chapter lists the different requirements we have defined in order to successfully create and solve the reverse engineering labs.

3.2.2 Requirements

Language
The labs and their solutions will be written in English to guarantee each student can understand it.

Targeted Group of Students
The product of this project is aimed at students in their third year (fifth semester) or higher because it is an in-depth look at a cybersecurity subject. Students taking this course should have visited the mandatory subject "Cyber Security" to have basic information and maybe even "Secure Software" for more advanced knowledge of some exploitation techniques. A rule of thumb is the more security lectures a student has visited and finished, the better.

Time Requirements
Each of the labs has a different time requirement for the students. One lab should be solvable in an hour or less.

Grading
Depending on the lab, a student has to hand-in a flag and/or a writeup. These will be checked by the teacher or an automated system.

3.3 Lab Documentation

3.3.1 Overview

This section is used to list the different labs and their purpose. The project domain created at the start was utilized as reference throughout the project to decide which parts of reversing should be targeted for each of the labs. Because of this the domain map is used by the participating students to dispay the contents of the lab.
Since the labs are created for the purpose of teaching, objectives and grading was defined in addition ot the explanation why this part of the domain was addressed.

3.3.2 Tools Introduction: GDB

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

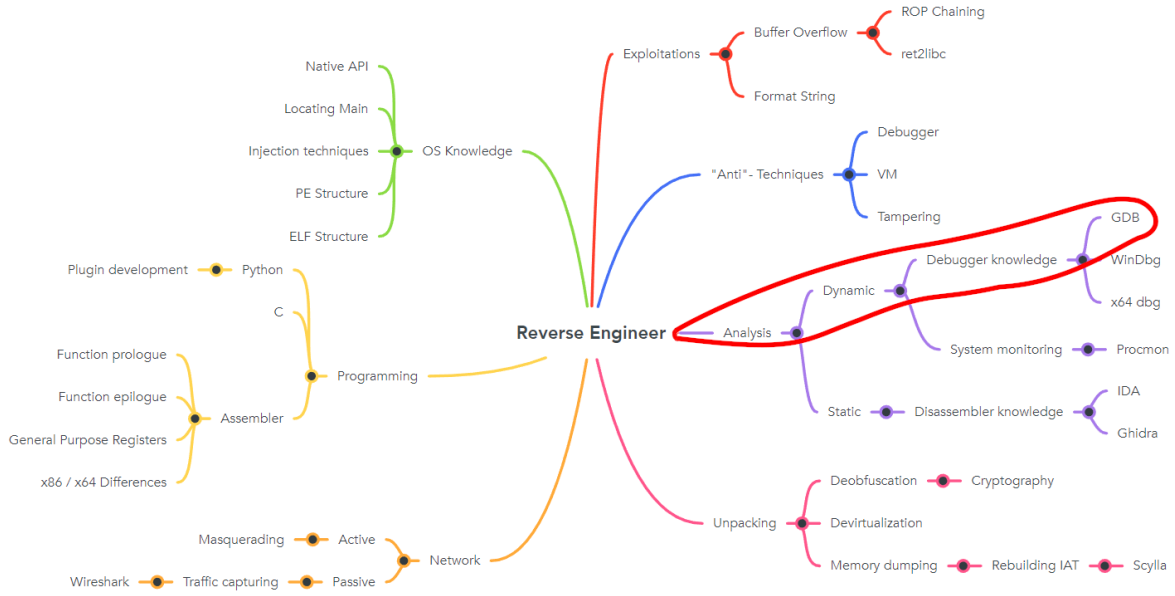

Figure 3.1: GDB domain overview

Content

This introduction is used to explain the basic functionalities of the tool used in some of the labs. In this case GNU Debugger (GDB).

Choice of Topic

The solutions and tips in the labs are given via screenshots or explanations. If a student wants to use a different tool, he / she is free to do so. It is important that the tools used in the example solutions are known to the students. This makes it easier for them to understand each step and, if they are stuck at any point, follow the instructions on how to solve it.

Objectives

- Learn about GDB and how to use the different commands available

Grading

The labs are not graded since they are only used as a lookup. They have a flag to make sure the students have read the tools.

3.3.3 Tools Introduction: x64dbg

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

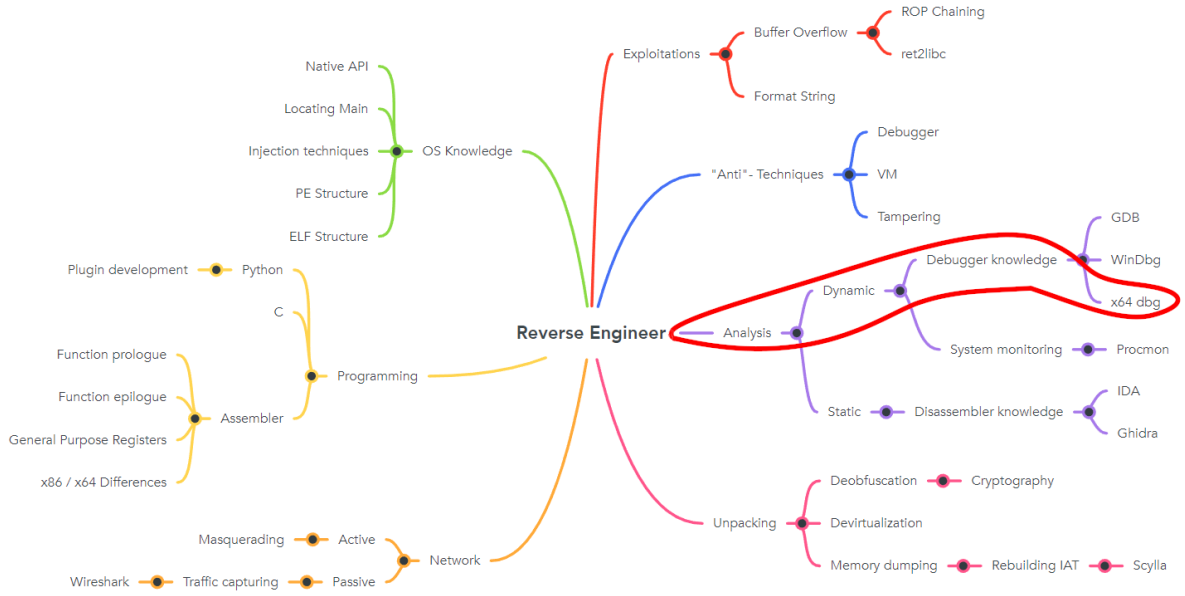

Figure 3.2: x64Intro domain overview

Content

Because most students have probably never touched a debugger on Windows before, this lab will help getting their feet wet with one of the most widely used ones by professionals. This lab also acts as the basis for the upcoming labs where x64dbg will be used.

Choice of Topic

Students working with Windows should also be given the opportunity to use a debugger. This lab will guide them through the GUI of the program and explain the basic functionality of what is coming up in future labs.

Objectives

- Install x64dbg and get a brief overview
- Get to know the functionality of x64dbg and be ready to use it on a binary

Grading

The introduction labs are not graded since they are only used as a lookup. They have a flag to make sure the students have read the tools.

3.3.4 Tools Introduction: IDA Freeware

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

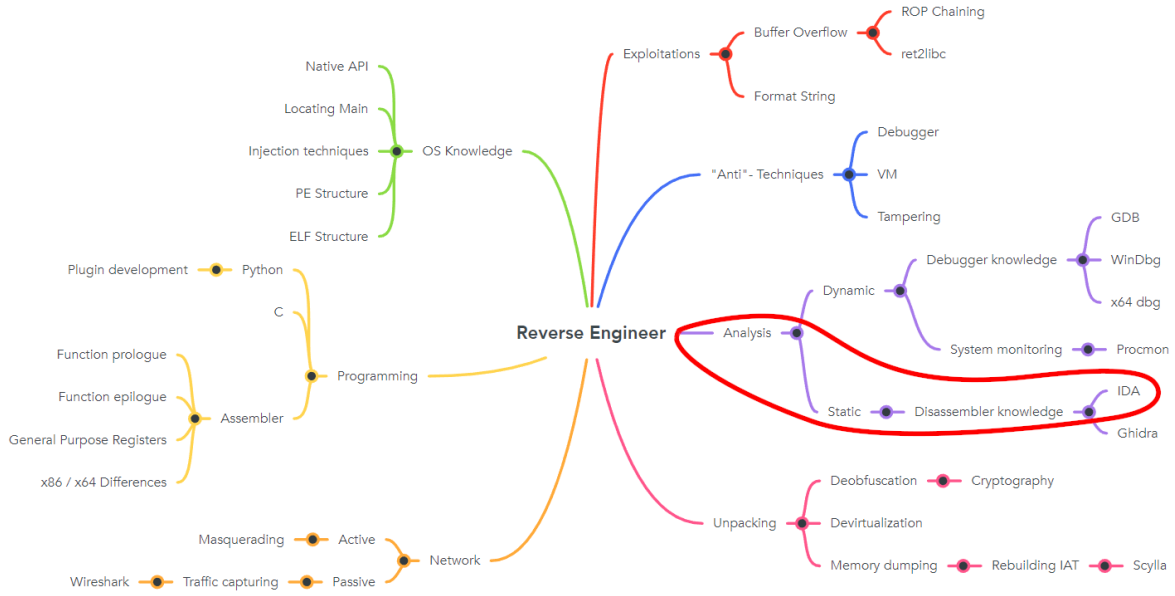

Figure 3.3: IDAIntro domain overview

Content

This introduction explains all the relevant functionalities of IDA and how to install it.

Choice of Topic

Some of the reverse engineer's most powerful tools are disassemblers. We purposely chose IDA Freeware over Ghidra for the beginning. This prevents the students from just generating pseudocode instead of actually reading and understanding the assembly instructions in later labs. Because of this, the solutions of future labs are presented using IDA Freeware.

Objectives

- Install IDA Freeware and get a brief overview
- Learn to orient yourself in IDA and get ready to use it on a binary.

Grading

The introduction labs are not graded since they are only used as a lookup. They have a flag to make sure the students have read the tools.

3.3.5 Lab 1: Asm-Refresher

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

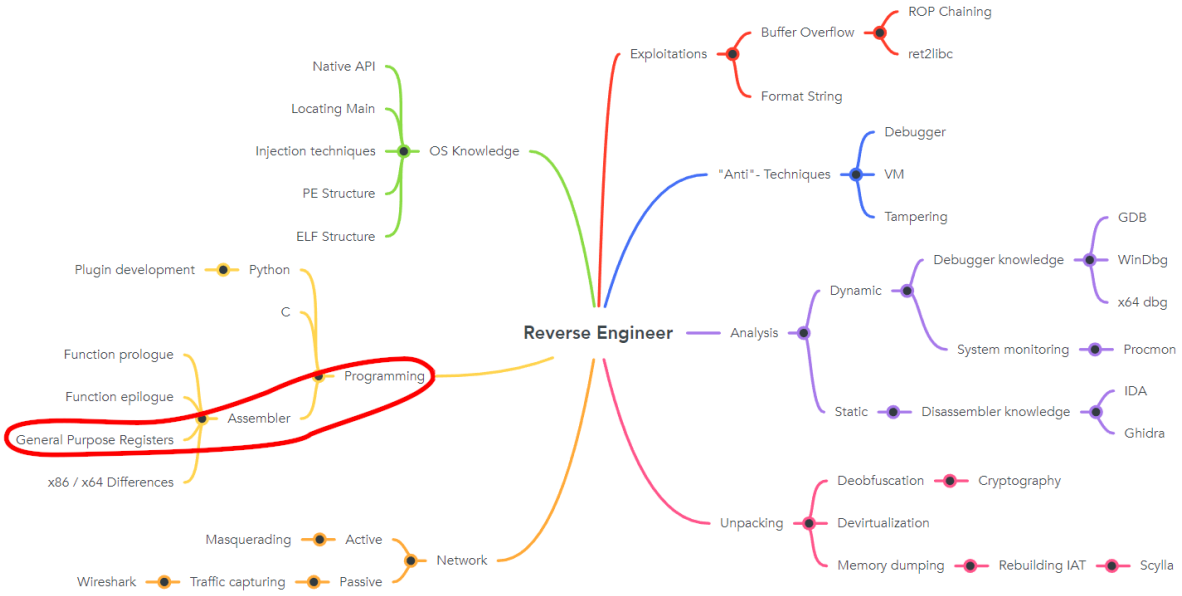

Figure 3.4: ASM refresher domain overview

Content

This lab is designed to be a refresher for those who have not done anything assembly related in a while. It covers the basic assembly instructions, explains the general purpose registers and rounds up with a guide on how to write a "Hello World" program.

Choice of Topic

A reverse engineer has to understand the basic concepts of assembly. Not only when reading through dissasembled code but also later on when automatic pseudocode generation fails.

Objectives

- Refresh knowledge about the general-purpose registers.
- Basic understanding of assembly instructions

Grading

This lab will not be graded since it is meant to help the students in later labs if they struggle with assembly.

3.3.6 Lab 2: Static Debugging

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

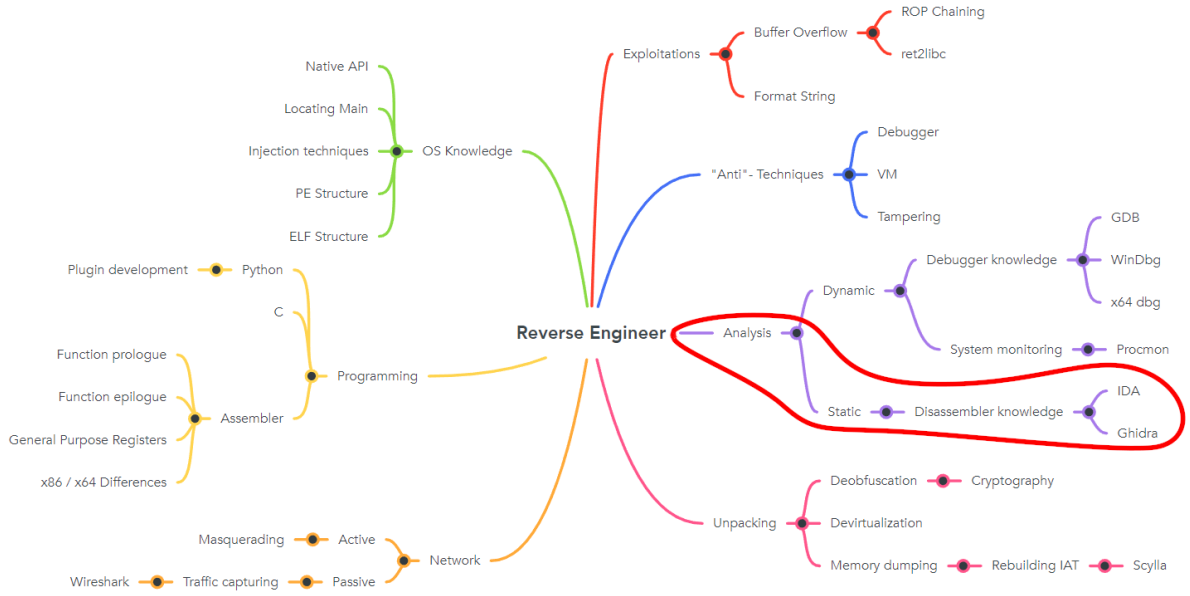

Figure 3.5: Static debugging domain overview

Content

In this lab, the student will learn how to use IDA Freeware to statically reverse a given binary. The goal is to find the main function in a binary with and one without symbols. This will show the student that seemingly simple things such as finding the main function can be a real problem if you have insufficient knowledge of the inner workings of a binary and the automatic detection of IDA fails.

Choice of Topic

Program execution starts from the main() function which is why it is an important skill of a reverse engineer to find it and start understanding the rest of the binary from there on.

Objectives

- Find the main function in a normal binary
- Find the main function in a binary with symbols stripped

Grading

The student has to inspect the main functions and find a flag in both of them. The final hand-in will be the combined flag.

3.3.7 Lab 3: Dynamic Debugging

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

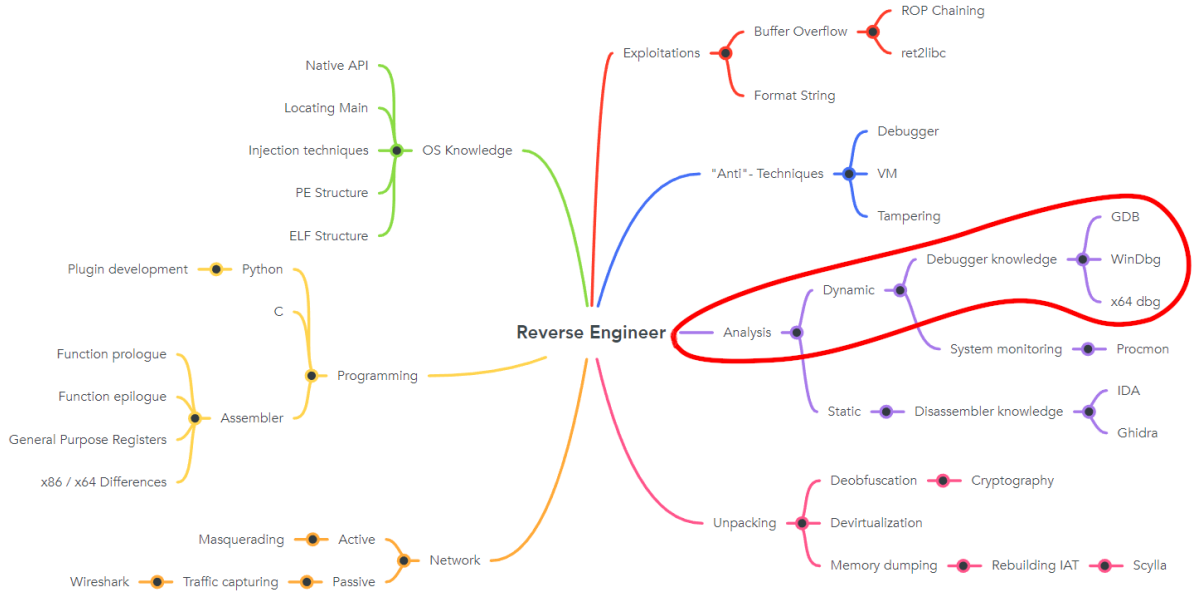

Figure 3.6: Dynamic debugging domain overview

Content

In this lab, the student will learn how to use GDB and x64dbg to dynamically reverse a given binary. In a first step the lab explains how to dynamically debug the binary given in lab 2 and then, in a second step, the student will be presented with a new binary containing a flag.
The goal of this lab is to show the student how this skill is used and then have him do it on his own to solidify the steps he has completed before.

Choice of Topic

Next to static debugging, dynamic debugging is used in a wide range of reverse engineering. Because of this it is important to have a student go through the steps and explain how it is done.

Objectives

- Use GDB or x64dbg to find the flag of the binary
- Find the flag of a new binary using dynamic debugging

Grading

The student has to use the newly aquired skills to find the flag in a new binary using dynamic debugging skills.

3.3.8 Lab 4: First Reversing Attempts

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

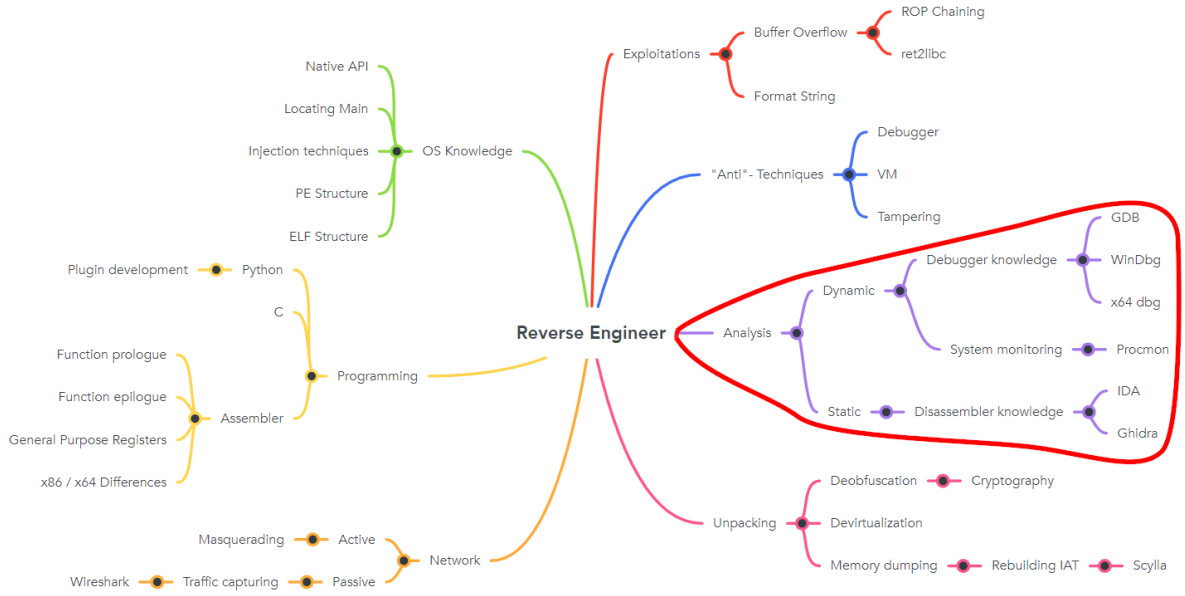

Figure 3.7: RE attempts domain overview

Content

In this lab, the student will deepen his knowledge of reversing binaries to find out how they work. This challenge contains two binaries having different requirements that have to be met in order for the student to receive the flags.

Choice of Topic

It is important to give the students simple examples where they can play around and learn by doing without providing a step by step guide.

Objectives

- Use static or dynamic debugging to find the requirements of the programs
- Find out how program arguments are handled in assembly

Grading

The student has to solve both binaries to get a flag.

3.3.9 Lab 5: Remote Login

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

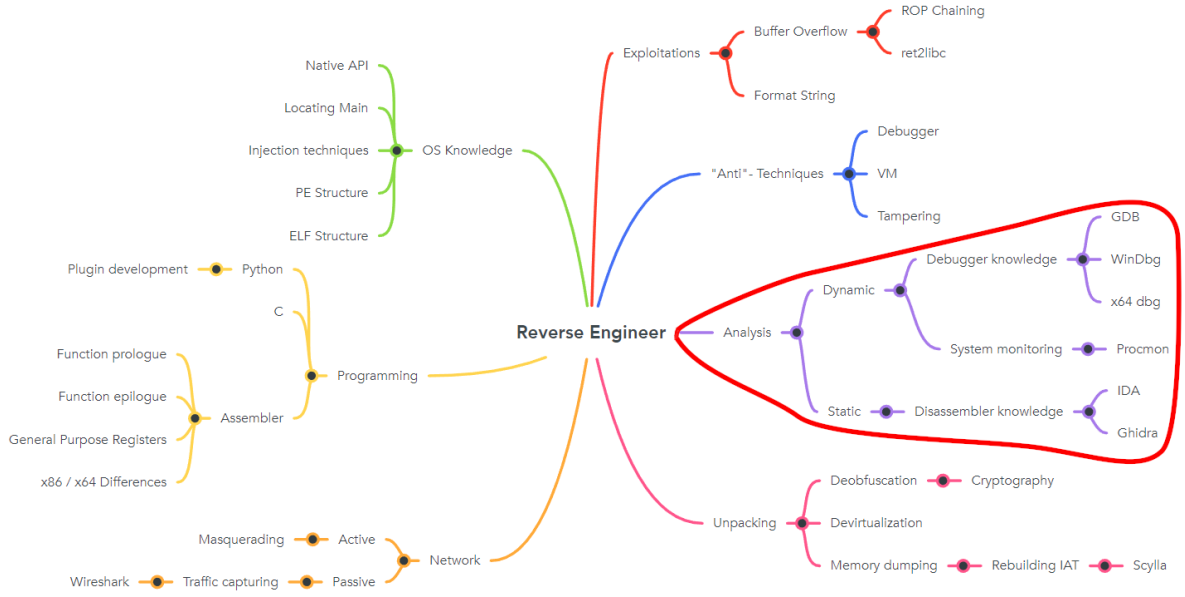

Figure 3.8: Remote login domain overview

Content

This lab dives deeper into the reverse engineering rabbit hole and introduces a new concept. The student first starts a docker container off a provided resource. This docker container exposes a web server and a port where an application runs. The application running on the server can be downloaded from the server's website by the student.
The student then has to combine dynamic and static reverse engineering to figure out the password in order to log into the server and expose the flag.

Choice of Topic

Giving the students an idea that reverse engineering can be used to gather information that you can later exploit on a target system.

Objectives

- Learn to use the tools you got introduced to
- Apply basics of dynamic and static debugging and find a way to exploit remote target

Grading

The student has to find out the flag and submit a small writeup to answer the security questions provided.

3.3.10 Lab 6: Pwntools - Introduction

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

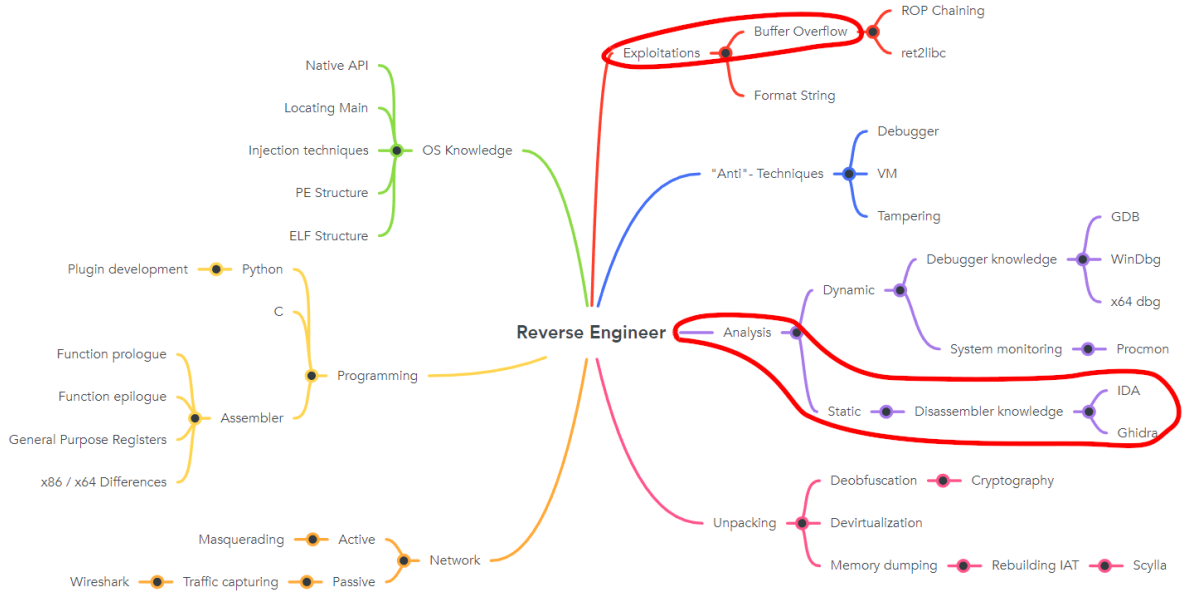

Figure 3.9: Pwntools domain overview

Content

In this lab, the student has a first introduction into the pwntool module of python. The student first has to start a docker container from the provided resource. This docker exposes both a web server and a port where the application is running on. The student then downloads the compiled application to analyze it and search for a weakness to exploit. This weakness can then be exploited through the use of pwntools. Since this is a first introduction, the whole process of creating the script is guided as a walkthrough.

Choice of Topic

Pwntools is an important module to learn for exploiting. Reverse engineering in general is widely used to find vulnerabilities to exploit, which means it is important for a student to not only know how to find the weakness but also how to exploit it.

Objectives

- Use acquired skills to find a vulnerability in a binary
- Create a pwntools script to exploit that vulnerabilty

Grading

The grading is done by sending in the printed out flag when the script is run on the exposed port and a writeup answering the provided security questions.

3.3.11 Lab 7: Crypto Lab - AES ECB

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

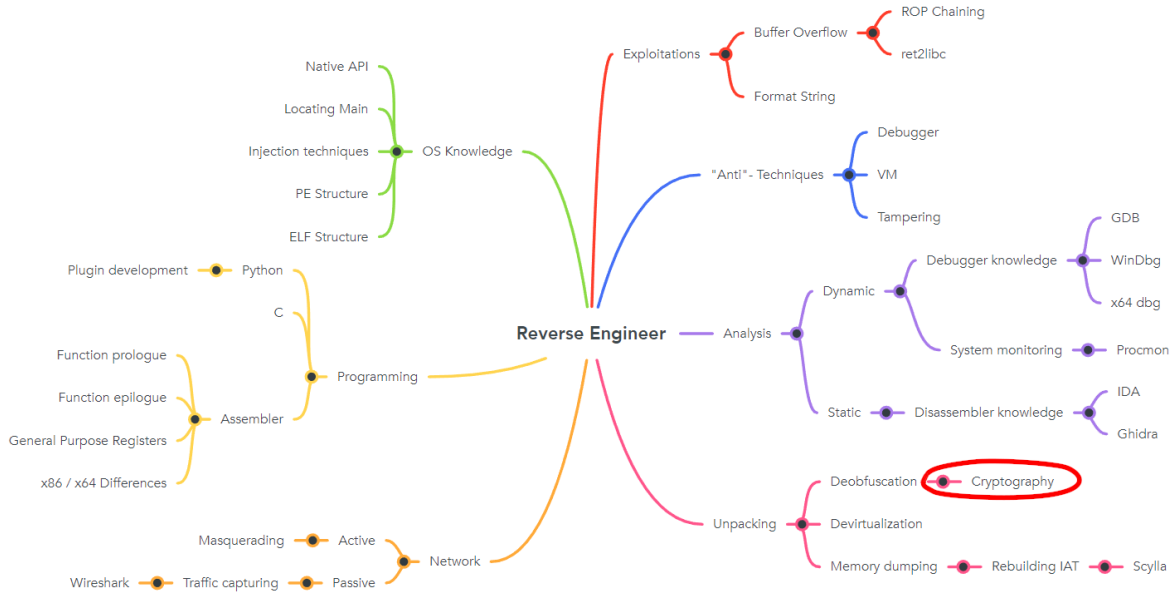

Figure 3.10: AES domain overview

Content

The student is presented a docker container which runs a web server and exposes a port on which the script for the student to crack is running. The student first inspects and analyzes the script itself and then follows a walkthrough on how to write a simple script which exploits a common vulnerability in AES ECB mode.

Choice of Topic

Encryption is a widely used form of obfuscation. Because of this it is important as a cybersecurity student to be informed about certain weaknesses these ciphers have and how to exploit them if needed.

Objectives

- Find out how the script works
- Find patterns to exploit
- Create a pwntools script to crack the encryption

Grading

The grading is done by sending in the printed out flag when the script is run on the exposed port in combination with a writeup answering the provided security questions.

3.3.12 Lab 8: Patching Lab

Problem Domain

This lab covers the following aspects of the reverse engineering problem domain shown in section 1.2:

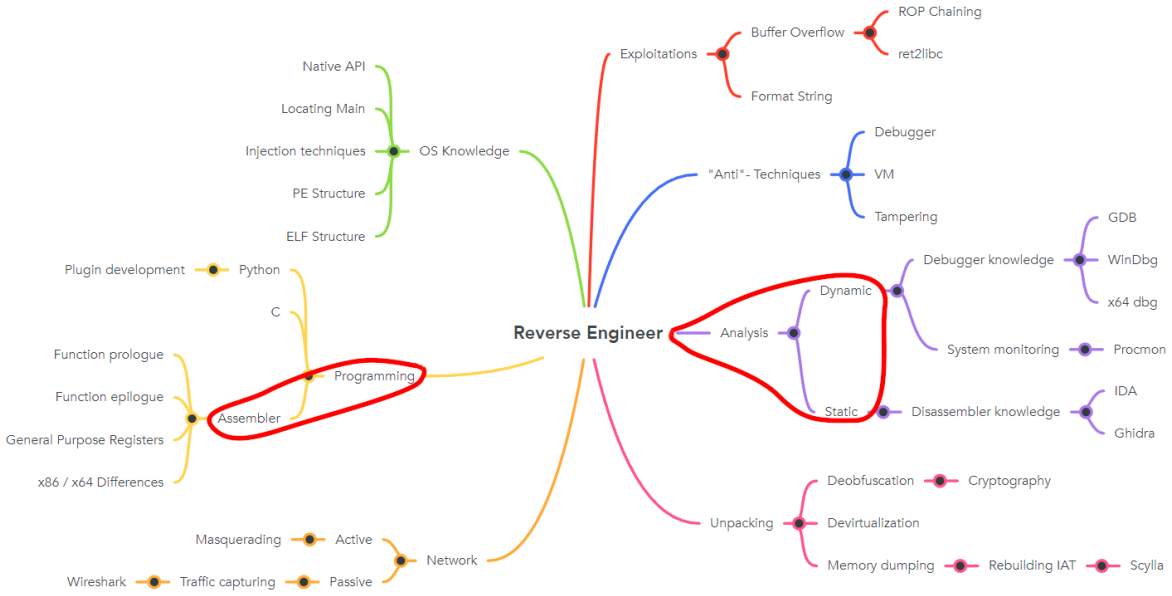

Figure 3.11: Patching domain overview

Content

The student is presented a binary which does not operate correctly. The student is tasked to find the location to patch and fix it. In order to check the patch generated by the user, the provided docker container starts up a web server where the student can upload the patch file. The web server then tries to apply the patch to a copy of the binary and checks if the bug is fixed. If fixed, the web server shows the flag to the student. If not fixed, the student has the possibility to reset the binary and try again.

Choice of Topic

Reverse engineering also comes in handy when encountering bugs in software where the source code is unreachable. Therefore it is important to locate found bugs in an executable and know how to apply patches.

Objectives

- Find the mistake of the binary
- Change the ASM code and upload the patched file to the website

Grading

The grading will be done via a writeup and a flag.

3.4 Results

We have achieved all the iteratively defined goals.
In the original project description, it was mentioned that ASLR and DEP should be a part of the labs. We have updated these requirements in our continuous process of constructing the labs with their corresponding learning goals. While ASLR is included, DEP is out of scope for now and is better suited to be handled in the future continuation of this project, the bachelor thesis.

3.5 Conclusion

We were able to create the amount of labs wished and even had time to implement one more. It covers the basics needed for a smooth start into reverse engineering and also follows a common thread while posing increasingly difficult challenges to the solver.
This also acts as a fundament for future work and can be extended with more advanced reverse engineering exercises.

Chapter 4

Project Documentation

4.1 Project Plan

The goal of this project is to create and organize a lab, which shows and explains future students of the Ostschweizer Fachhochschule (OST) how reverse engineering is performed and which tactics are used to get information out of a program. To accomplish this task, the lab will have several exercises organized in the different domains. These exercises will be accessible through the Hacking-Lab website.

Hand-In

The finished report will be handed in according to the rules set by the "Studiengangsleitung Informatik" and the supervisor:

- The PDF version will be sent to the advisor and to the OST archive.
- The printed version will be handed in to the supervisor for reading and grading.

4.1.1 Management

Time Management

The project started on the first week of the semester (KW 38) and ends in week 51 giving us around 14 weeks to be done with the Hand-In.
Since the module has a total ECTS of 8 each of the students has to work around 240h during the semester which can be seen in table 4.1 together with the total planned time investment. This means, that per week each student should work around 17.1 hours.

Planning and Project Management

In the past modules Software Engineering Practices (SEP) 1 and 2 we were introduced to different ways to plan and organize a project. The main tools we learned, Rational Unified Process (RUP) and Scrum, are mainly used in software development but can be

Name	ECTS	Time spent per Week [h]	Total Time spent [h]
Gianluca Nenz	8	17.1	240
Ronny Mueller	8	17.1	240
Thomas Kleb	8	17.1	240
Total	32	52.3	720

Table 4.1: Time Investments

adapted to other projects aswell. They both use different aspects of time management and organisation which is why we intend to apply them to our project.
We use RUP to section our project into Inception, where we get a first insight into the project and how we want it to resolve; Elaboration, to plan our project, define the workload-distribution and setting up first concepts of the finished labs; The construction phase is mainly used to plan, build and test the labs while the last phase, the transition phase, is used as buffer and to finish our product.
To make sure everything works as planned we use Scrum with its sprints to setup milestones and tasks which help structurize the development.

4.1.2 Organisation

Participants

The "Studienarbeit"-team consists of three students: Gianluca Nenz, Ronny Mueller and Thomas Kleb. The work on the project and documentation will be evenly distributed between these three participants. Bigger decisions are made as a team in either the meetings with or without the advisor (the advisor will be notified on any change made).

Advisor

The teams advisor for the "Studienarbeit" is Ivan Buetler who is teaching cybersecurity modules at the OST.

Division of Labor

The project has multiple facets that need to be taken care of. This is why the team has decided to distribute the work load between the three. This doesn't mean that the work is done by only the chosen student but rather that he is the one responsible that it works as planned.

Gianluca Nenz
Sprint Meetings
Lab 3: x64dbg
Lab 4: ASM - Refresher
Lab 6: Dynamic - debug
Lab 8: UN-PW

Ronny Mueller
Meeting Notes
Lab 1: IDA
Lab 5: Static-debug
Lab 7: First Reversing Attempts
Lab 11: Patching

Thomas Kleb
Testing
Documentation
Lab 2: GDB
Lab 9: Pwntools
Lab 10: AES

Table 4.2: Work Distribution per Student

4.1.3 Planning and Milestones

Phases and Iterations

The project is comprised of the four steps of RUP. Each of those phases has multiple iterations which create the different sprints for the project. The meetings with the advisor

will be on thursdays while the team meetings will be held tuesdays. Each iteration / sprint will be of a seven day length.
We started the "Studienarbeit" before we began with the regular school. In the week before we each made research and plans about the upcoming project. After having a talk with the advisor it was decided to first find out the level of knowledge each student has to allow easier planning for the advisor.

Inception			
Iteration	Start	End	Description
0	12.09.2022	18.09.2022	Collection of ideas and planning first meeting
1	19.09.2022	25.09.2022	First meeting and handout of exercises to assess the knowledge of the students
2	26.09.2022	02.10.2022	Working on the exercises and receiving solutions for harder ones

Table 4.3: RUP: Inception Phase Planning

The elaboration phase is used to plan and assess the possible risks in this project. This consists of a documentation structure, the project plan and the risk management to make sure the construction phase has no major hickups.

Elaboration			
Iteration	Start	End	Description
3	03.10.2022	09.10.2022	First big meeting with advisor; Creating project plan and risk analysis.
4	10.10.2022	13.10.2022	Project plan and documentation is set; Problem domains and learn concepts are defined
5	14.10.2022	25.10.2022	Lab concepts are defined

Table 4.4: RUP: Elaboration Phase Planning

The construction phase is where the labs are primarily built.

Construction			
Iteration	Start	End	Description
6	21.10.2022	01.11.2022	POC for static and dynamic challenge; started testing
7	01.11.2022	08.11.2022	Lab static and dynamic finished; POC First RE Attempts and UN-PW with testing
8	08.11.2022	15.11.2022	Finishing lab UN-PW and RE Attempts; POC Pwntool lab
9	15.11.2022	22.11.2022	Introduction labs, AES and testing
10	22.11.2022	29.11.2022	POC patching lab and assembly lab
11	29.11.2022	06.12.2022	Internal testing of all labs finished; all labs complete

Table 4.5: RUP: Construction Phase Planning

To make sure enough time is planned a buffer week was added to the transition phase. This phase is also mainly used to finish up the documentation and implement the different challenges to Hacking-Lab. The last week is used to clean up and hand in the documentation and abstract to both the OST and the advisor.

Transition			
Iteration	Start	End	Description
12	06.12.2022	13.12.2022	Documentation completion
13	13.12.2022	20.12.2022	Finalizing documentation
14	20.12.2022	23.12.2022	Preparing for hand-In

Table 4.6: RUP: Transition Phase Planning

Milestones

To guarantee the success of the project milestones were defined with a deadline.

Milestones	Deadline	Description
M1 - Solving RE Exercises	05.10.2022	The Team tries to solve the given exercises to find the level of RE knowledge.
M2 - Defining problem domains and learn concepts	13.10.2022	Problem domains are defined, first learn concepts are planned
M3 - Lab Concepts	25.10.2022	Lab Concepts are defined to start working on the construction.
M4 - Setup Labs	06.12.2022	Labs are setup and tested.
M5 - Hand-In	23.12.2022	Document is handed in to the advisor and OST

Table 4.7: Milestones set for the project

Time Tracking

For time tracking the team has decided on using a third party application named Clockify [3]. To have an overview on what the team spents its time on we decided to create different tags:

- Documentation: Working on the documentation
- GitLab: Maintaining the infrastructure of the repositories
- Meetings: Sprint- and advisormeetings
- Preparation: Research before the start of the project
- RE-Labs: Creating the different labs
- Research: Researching for the labs
- Testing: Testing the labs with different volunteers

Issue Tracking

The issue tracking is done on GitLabs own interface to have as few difficulties as possible. To have an easier overview of the different issues the team has created tags to differentiate between the issues and their assigned student.

GANTT Diagram

Figure 4.1: GANTT chart

Meetings

The team has meetings each tuesday to elaborate problems and check up on the progress. These meetings are also used to distribute the work load.
On thursdays the team meets the advisor Ivan Buetler to inform him on the progress done and the problems that came up. These meetings have different time schedules to fit everyones calender.
Each meeting will be documented and uploaded to the GIT repository. After each meeting the participants should know what to do and how to contact each other if any problems arise.

Project managment

The whole project will use a GitLab repository. To make sure no confusion happens a multirepo principle is used where one repository is only for the documentation and protocols and the other one is for code, information gathered, etc. Each student works on a branch and creates a merge request where the other students check the proposed changes for errors and when nothing is found, they merge the changes.

4.1.4 Testing

This chapter describes how the quality of the challenges was determined and the feedback was evaluated.
The labs were tested by many different participants. This testing was done to ensure the challenges were solvable and understandable for the intended purpose. The testing can be divided into two parts.

Internal Tests

After every challenge was completed and uploaded to the Hacking-Lab platform, the creator checked the challenge again. After the creator finished his testing, he signaled the other students from the group to test the challenge. Any questions, mistakes or feedback from these tests were immediately reported to the creator. He then decided which parts he had to change. Additionally, at the next supervisor meeting the challenge was presented and the supervisor was also able to give feedback to the challenge.

Testing with third-party students

In order to get realistic feedback from students, we asked cybersecurity students in their fifth semester to solve the labs and provide feedback. This feedback was collected using Google Forms. This tool allows creating simple surveys, which can then be analyzed in graphs automatically.
Only some of the challenges were solved by this group. The feedback received was helpful. However, since many of these students were also busy with their own projects, it was a bit difficult for them to find the motivation to solve them thoroughly. Testing the labs meant a significant amount of work and therefore these tests were only done by a few individuals for all the challenges.

Feedback

The challenges were perceived as educating and easy to follow by the testing participants. The main takeaway from these tests were the following points:

- Challenges are clear and interesting
- Challenges teach concepts in a fun way
- The time requirements for each challenge vary by much

Conclusion

The feedback was mostly positive. Most of the feedback mentioned typing errors or some errors with setting up the challenges on Hacking-Lab.com. Some feedback also mentioned imprecise steps in the solutions. They were very useful to create the challenges in a way which should be understood by other students.
But we also think that our testing process could need some refining. We didn't really have a broad spectrum of knowledge and motivation in our testing participants because they had to do it in their leisure time. This way, we only got very motivated students,

who probably also had way more knowledge than the average one. In future projects, we should strive to achieve a more normalized testing group by asking to be able to test some challenges in a security class or similar.

4.2 Risk Analysis

4.2.1 Risk Managment

For this project, the "Project Management Triangle" is lacking the cost dimension, while the time dimension is fixed (strict deadlines). As a result, any risks that appear, automatically lead to a reduction of the project scope if there is no spare time. Because of this, we will prioritize dealing with risks above regular tasks and prioritize essential tasks over nice-to-haves, but we do not intend on planning in a flat time margin as we have no way to negotiate for more time.

4.2.2 Estimated Risks

The numbers in the following picture describe the position of the risks on the following page from top to bottom.

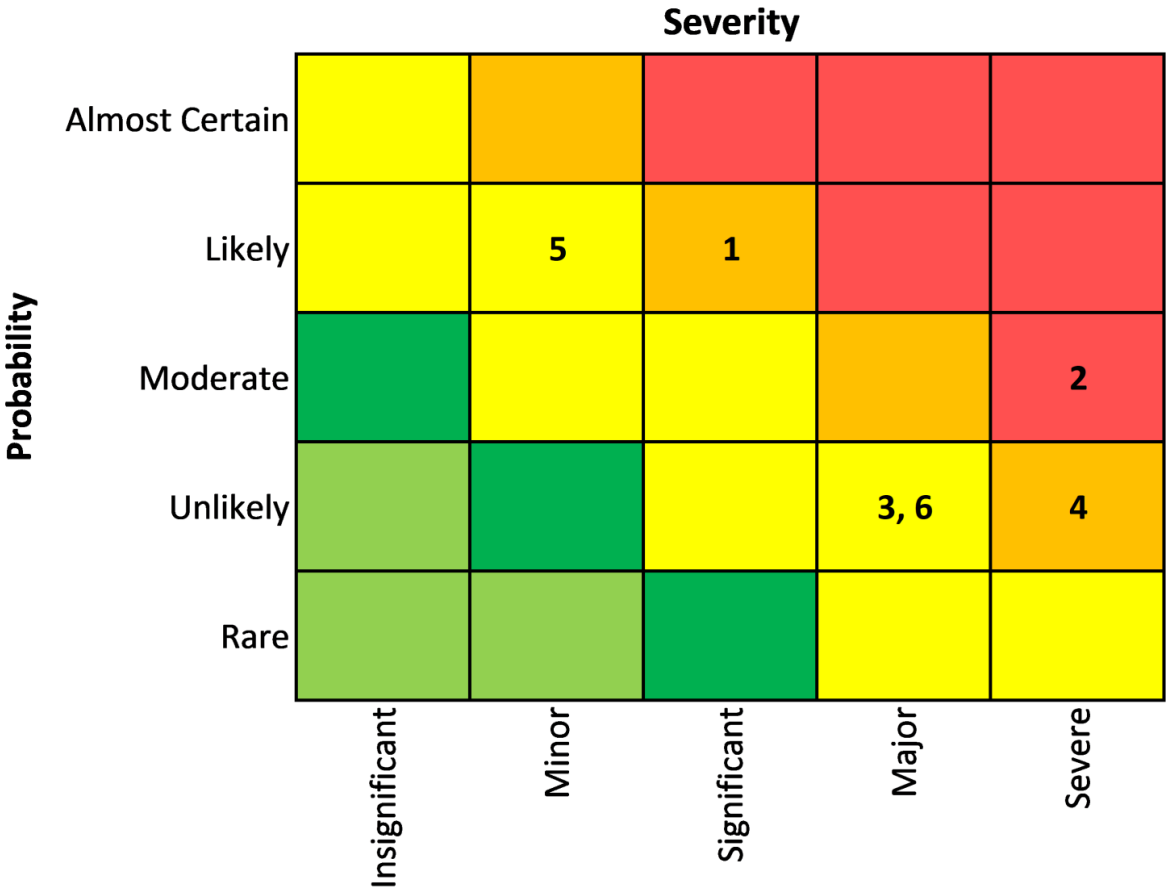

Figure 4.2: Risk matrix

Risk 1:	
Description	Finding testing participants
Severity	Medium
Probability	High
Mitigations	Already got participation confirmation from testers
New probability	Low

Table 4.8: Testing participants risk

Risk 2:	
Description	Being able to create reversable programs with additional difficulties
Severity	Very High
Probability	Medium
Mitigations	Assured the advisor is available for consultation
New probability	Low

Table 4.9: Create programs risk

Risk 3:	
Description	Not enough time for the actual challenges because of too much programming etc.
Severity	High
Probability	Low
Mitigations	Creating challenges in chronological order and in an iterating fashion
New probability	Very Low

Table 4.10: Not enough time risk

Risk 4:	
Description	Irreparable corruption of git server
Severity	Very High
Probability	Low
Mitigations	Weekly off-site git server backups
New severity	Low

Table 4.11: Corruption of git risk

Risk 5:	
Description	Lost work due to unpushed work
Severity	Low
Probability	High
Mitigations	Frequent reminders to push changes
New probability	Low

Table 4.12: Unpushed work risk

Risk 6:	
Description	License problems with used software
Severity	High
Probability	Low
Mitigations	Trying to use opensource or public software
New probability	Low

Table 4.13: Licence problems risk

4.3 Project Monitoring

4.3.1 Overview

This section of the documentation is used to overview the different states of our project in comparison to the goal we set in planning or in the meetings hold with the advisor protocolled in the meetings chapter found in the appendix.

4.3.2 Milestones

Milestones	Deadline	Notes
M1 - Solving RE Exercises	05.10.2022	Milestone finished without complications
M2 - Defining Problem Domains and Learnconcepts	13.10.2022	Milestone finished without complications
M3 - Lab Concepts	25.10.2022	Updated Milestone to have the concepts be worked on parallel to the construction
M4 - Setup Labs	06.12.2022	Milestone was finished on time but a docker setup error did delay testing
M5 - Hand-In	23.12.2022	Milestone finished without complications

Table 4.14: Monitoring Notes for the Milestones

4.3.3 Time Tracking

We removed the last week of the project from the time tracking. Because we wanted to be finished with the documentation the week before.

Time spent per Teammate

Name	Average Time spent per Week [h]	Total Time spent [h]
Gianluca Nenz	15.1	196.27
Ronny Mueller	15.17	197.2
Thomas Kleb	15.49	201.36

Table 4.15: Recorded Time Investments

Time spent per week

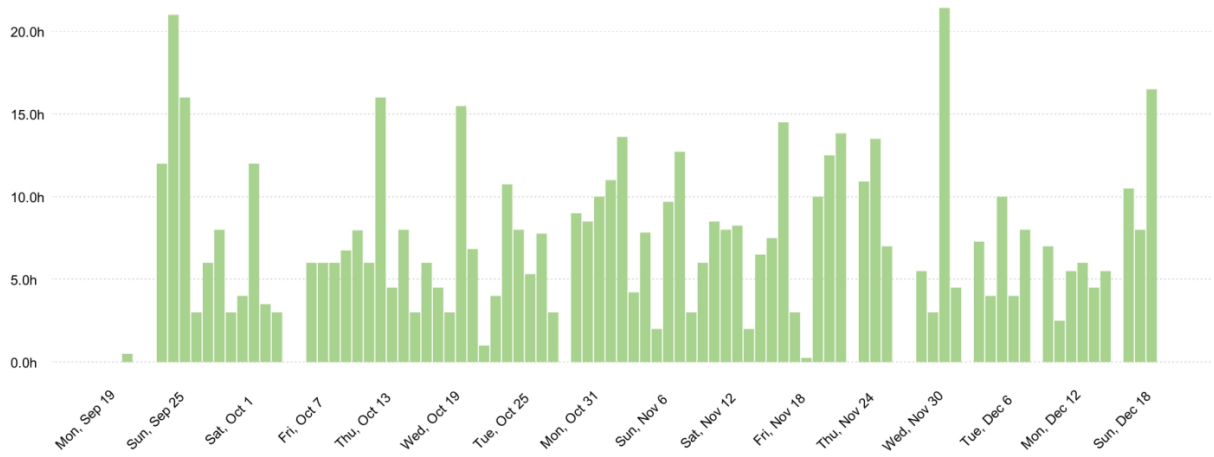

Figure 4.3: Our time spent per week

Time spent per category

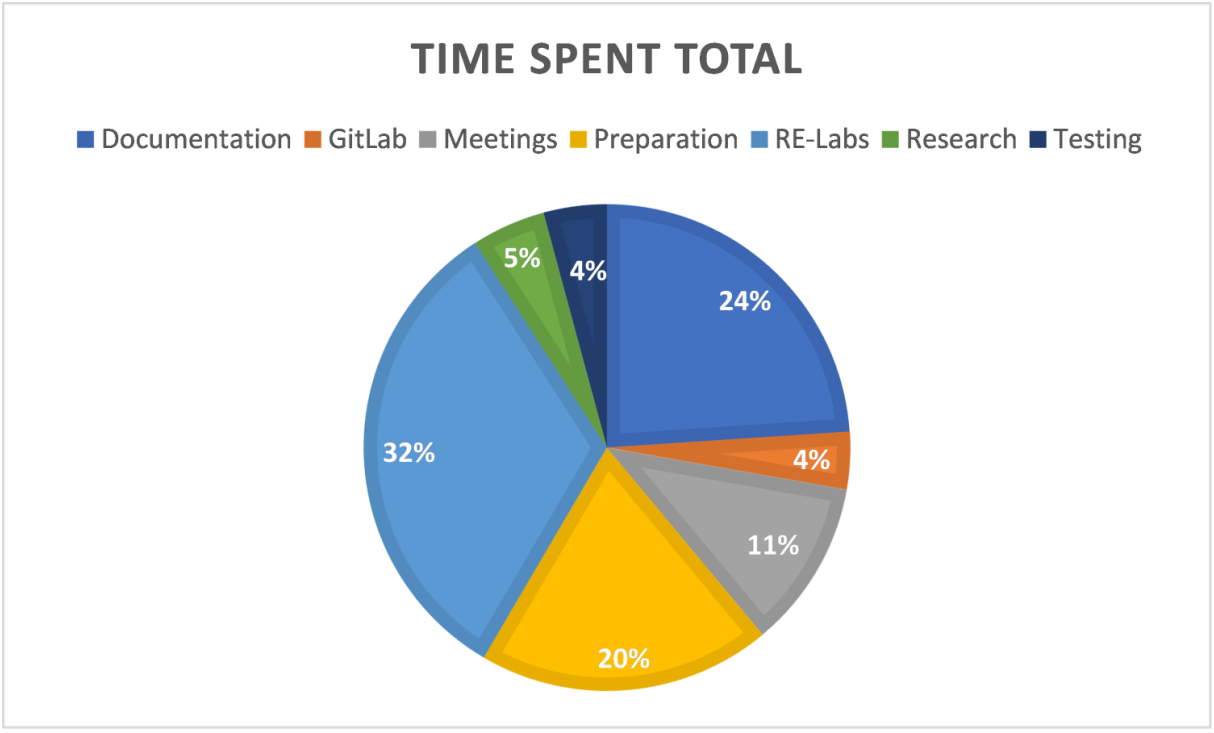

Figure 4.4: Our time spent per category

4.4 Personal Rapports

4.4.1 Gianluca Nenz

Thanks to Ivan, we had a great start into this topic by solving sample challenges and thus evaluating our skill level. At first, we had to adjust our workflow since it differs from the one we have learned in the "SEProject" previously. Once everything settled we quickly got a good overview of the problem domain and started iteratively working on our labs which worked well. I really liked this approach and would love to do it again this way. For the next time though, I would plan some more ahead or have enough side projects ready to switch to if I finished the lab faster than anticipated. Thanks to this, I would always have enough to work on and my working hours would be distributed more evenly over the time of the project. Finding willing testing participants turned out to be more difficult than we first thought because most students do not have enough spare time to test everything.
Thanks to our friends we were able to get everything tested, but maybe we can somehow integrate testing as part of a lecture at OST or somewhere else in the future.
All in all I am really happy with the outcome of our "Semesterarbeit". We have calculated enough time to finish up our labs and the documentation in time without the need of several night shifts the days before hand in. This could not have been made this smooth without the great teamwork we had. Thanks to everyone for putting effort into making good, beginner-friendly labs.
I am thankful to have had the opportunity to bring reverse engineering into more students lives and I am looking forward to digging deeper in future projects.

4.4.2 Ronny Mueller

Our process of working on the challenges in an iterating fashion seemed to work great and I would do it like that again. The hours invested each week varied by too much and I think we should somehow look for a more well balanced time investment. I also that even though we got some testing participants we should aim to test the exercises inside an exercise session at the OST. So the students don't have to sacrifice their leisure time to test our challenges.
The first week of the project I was still having one week left of mandatory military service. Because of this I want to thank everyone involved for their consideration.
I am happy with the final challenges and think that they cover good basics of reverse engineering and is a good entry point for students in the same position as we were. I am very grateful towards Gianluca Nenz and Thomas Kleb for their hard work and dedication they put into this project.

4.4.3 Thomas Kleb

Reverse Engineering for me was a relatively new subject but as I am highly interested in all things cybersecurity I was glad, that we got accepted for this project. The start of it was a bit stressful for me since we never had to do a project this size without many rules and guidelines, we were totally free.
Once we had our first meetings with Ivan everything got clearer and we knew how to structure this project. These meetings really helped me organize and I am extremely thankful that we had such professional guidance from our advisor. Working on the different labs and finding problems to teach each student was my highlight and i am looking forward to do it again in my Bachelor Thesis. Our team was well organized and the different tasks were distributed fairly thanks to multiple meetings. What I didnt plan for was the downtime between the creation of the labs. Which is why I want to plan more precisely in the future to have more work done in between. The problems created during the building of the labs like docker not working as intended, LateX not doing what I want or even finding different test subjects, were a lesson to be taught at the end. For my next work I want to take what I have learned from this and improve on it, especially the preparation since we had to do some prepare work which could have been done beforehand.
At the end I am glad to have had the chance working on such a project and I am thankful for all the help we got from our advisor. Overall I enjoyed working in this team and on the project and can't wait to do it again next semester.

Acknowledgement

This work was made possible by OST in Rapperswil-Jona. We thank them for the possibility to be able to work on such a project.

We thank Ivan Bütler for proposing such an interesting topic and for advising us throughout the process of this work. His insight and expertise greatly helped us to steer the complex path of reverse engineering. Also we are thankful towards Compass Security who made it possible to publish our challenges on their platform.

We would also like to show our gratitude towards our testers, who sacrificed their leisure time to ensure the quality of our challenges. This is no small feat because they also had to work on their own projects. Their insights and feedback was integral to the quality of our product.

List of Figures

List of Tables

Appendix

Meetings

Nr	Phase	Date	Description	Duration [min]
1	Elaboration	06.10.2022	Coordinate the project, documentation and ideas	90
2	Elaboration	13.10.2022	Present the problemdomain with learning concepts and define the project plan	60
3	Elaboration	20.10.2022	Lab concept drafts, GANTT diagram	80
4	Construction	27.10.2022	Think about the exploitation aspect and add it to mindmap; POC for lab 2 and 3 and started testing	86
5	Construction	03.11.2022	POC for lab running with docker , finish lab 5	80
6	Construction	10.11.2022	Finish labs and create a POC for pwntools lab	120
7	Construction	17.11.2022	Fix Pwntools lab, Introduction labs, testing	60
8	Construction	25.11.2022	Concept for patching Lab, testing, refresher lab	90
9	Construction	01.12.2022	Patching lab fixing, testing, setting up forms for feedback	60
10	Transition	13.12.2022	Discussing documentation	120

Table 4.16: Meetings held with advisor

Elaboration Phase

Meeting 1: 06. Okt.

Deliverables: Draft the problemdomains together with learningconcepts and the projectplan.
Discussed Topics:

- Documentation
- Goal of the project: Hacking Lab integration for cybersecurity classes.
  - Teacher needs required knowhow
  - Labs built for a student attending the third year
- Brainstorming for problemdomains
  - Learningconcepts need to be defined for teacher and for lab construction
  - RE tools
  - Required knowhow for a reverse engineer
- Mini projectplan brainstorming:
  - Analysis of problemdomains
  - Setting up learningconcepts
  - What is important?
  - Build lab concepts
  - Implement with tests

Decisions:

- Idea of the project: Hacking-Lab integration (modulintegration)

Duration: 1h 30min

Meeting 2: 13. Okt.

Deliverables: Build some lab concepts (deliverables and overall subject), GANTT diagramm
Discussed Topics:

- Problemdomain mindmap
- Learning concepts
- What a Hacking-Lab course needs
  - Deliverables defined
  - Show which skills are trained (mindmap / MITRA style table)
  - Exercises created as markdown for ease of use
  - Get inspiration from already established courses

Decisions:

- Add passive and active labs for networks.
- Write protocol of sprint meetings for own use but not needed in documentation.
- Create Hacking-Lab content on markdown files

Duration: 1h

Meeting 3: 20. Okt.

Deliverables:

- Students:
  - Think about the exploitation aspect and add it to mindmap.
  - POC for lab 2 and 3 and started testing.
  - Update GANTT chart.
- Advisor:
  - Sample courses for RE.
  - Unlock ECSC Hacking-Labs.

Discussed Topics:

- Online GANTT chart instead of excel chart
- Refresher and lab 1 (introduction #1)
  - Create at the end to know what is necessary for the labs
  - Use binaries from the already created labs
- Server part is important (analyzing and exploiting remote); socat discussed
- Generally try to add exploits at the end
- RE is used to find weaknesses and exploit them
- Markdown for the Hacking-Lab consists of: Main page / step pages and a solution page

Decisions:

- Timetracking: Only if the used tool (clockify) allows it, add estimated time
- Plan labs while creating others to maximize the efficency
- Start construction phase now (21.10.2022)
- Add exploits to the labs (remote server using docker)
- Start construction with lab 2 and 3 to ensure the refresher and lab 1 are done correctly

Duration: 1h 20min

Construction Phase

Meeting 4: 27. Okt.

Deliverables:

- Students:
  - POC for lab running inside a docker with downloadable binary and generated flag.
  - POC for RE-Lab with different compiler using public code.
- Advisor:
  - Update Hacking-Lab CD with packet for IDA.
  - Upload introduction labs 2 and 3 to Hacking-Lab.

Discussed Topics:

- S6 Overlay
- Use Hacking-Lab GitHub repo for references

Decisions:

- Windows: bootstrap script for windows (choco)
- Linux: Hacking-Lab CD packet for IDA
- Use source code of example exercises for ideas

Duration: 1h 30min

Meeting 5: 03. Nov.

Deliverables:

- Students:
  - Finish remote-exploit lab
  - Check out sent labs and Links
  - Finish RE-attempts lab
  - POC für Python PWN lab
  - IDA installation guide
- Advisor:
  - Upload finished labs to Hacking-Lab

Discussed Topics:

- How a Hacking-Lab lab is built (Sections and Steps) and how the "Competitive"-mode works
- Adding security questions as second section
- Add to the labs what the student should hand in (Flag, Writeup, multiple-choice or a combination of the three)
- Update of the two labs (remote and RE-attempts)
- Hacking-Lab generator ("generator-hl-challenge" on GitHub)

Decisions:

- Add difficulty to RE-attempts lab
- Don't use pyinstaller for packing since it makes the reversing annoying
- Create a PWN library lab
- Send PDF a day before meeting because of CICD
- Add docker-compose to all docker folders
- Choose mode to run the lab in (Competitive, Training, Optional steps)
- Use section, text and step for markdown

Duration: 1h 20min

Meeting 6: 10. Nov.

Deliverables:

- Students:
  - Fix pwntool lab
  - Introduction labs for IDA, GDB and x64dbg
  - Test the newly uploaded labs
- Advisor:
  - Migrate challenges from OST to demo tenant

Discussed Topics:

- Demo tenant for easier access and testing
- Hacking-Lab tutorial with editor and resource pages
- Reversing is used to understand code and explotiation is needed too.
- Lab ideas (IDA scripting, virus lab, more client-Server labs)

Decisions:

- Use demo tenant for labs before uploading to OST tenant

Duration: 2h

Meeting 7: 17. Nov.

Deliverables:

- Students:
  - Testing labs
  - GANTT-diagram
  - Refresher lab
  - Concept for the last labs
- Advisor:
  - Fix Modes and settings for labs

Discussed Topics:

- IDA installation on Hacking-Lab CD
- How to structure the Labs to upload it easily
- Other tenant for easier updating
- Pwntools lab problems with docker
- Refresher lab with coding example
- More ideas for last lab: PWN iter (Bruteforcing), DLL virus, patching lab, reverse shell

Decisions:

- Introduction structure with different steps
- Read mail regarding documentation structure
- Use given structure for the documentation
- Create a crypto lab
- Concepts for the last lab

Duration: 2h

Meeting 8: 25. Nov.

Deliverables:

- Students:
  - Patching lab
  - Testing
  - Mail for pwntools lab
- Advisor:
  - Upload pwntools lab

Discussed Topics:

- Pwntools lab
- Cryptographic lab works and is uploaded
- Assembly refresher is finished and uploaded
- Patching lab concept as favorite
- Calculator which needs to be patched to works
- Use website to upload patching file and display output

Decisions:

- Patching lab
- Start first testing
- Update documentation

Duration: 2h

Meeting 9: 1.Dez.

Deliverables:

- Students:
  - Patching Lab finished
  - Feedback Forms setup and integrated

Discussed Topics:

- Patching lab status
- Testing setup and forms
- Overview of all labs
- Documentation status
- Next meetings

Decisions:

- Only one more meeting (13. dez.)
- All labs are ready and can get tested
- Finalizing documentation until next meeting
- Use structure given by OST for the documentation

Duration: 1h

Transition Phase

Meeting 10: 13.Dez.

Deliverables:

- Students:
  - -

Discussed Topics:

- What belongs into Management-Summary and how to write it
- What to cover in technical report

Decisions:

- Management-Summary should be written in a manner that someone with zero knowledge would understand our project
- Technical report can be written in a manner that professionals should be able to understand it

Duration: 30min

Bibliography

[1] S. Morgan. "Cybersecurity jobs report: 3.5 million openings in 2025." (2021), [Online]. Available: https://cybersecurityventures.com/jobs/. (accessed: 04.11.2022).

[2] OST. "Studienschwerpunkte." (2022), [Online]. Available: https://www.ost.ch/de/studium/informatik/bachelor-informatik/studieninhalt-und-aufbau/studienschwerpunkte. (accessed: 04.11.2022).

[3] Clockify. "Time tracking tool." (2022), [Online]. Available: https://clockify.me/. (accessed: 09.11.2022).

Eigenständigkeitserklärung

Erklärung
Wir erklären hiermit,

- dass wir die vorliegende Arbeit selbst und ohne fremde Hilfe durchgeführt haben, ausser derjenigen, welche explizit in der Aufgabenstellung erwähnt ist oder mit de Betreuer schriftlich vereinbart wurde,
- dass wir sämtliche verwendeten Quellen erwähnt und gemäss gängigen wissenschaftlichen Zitierregeln korrekt angegeben haben.
- dass wir keine duch Copyright geschützten Materialien (z.B. Bilder) in dieser Arbeit in unerlaubter Weise genutzt haben.

Gianluca Nenz
OST Student

Ronny Mueller
OST Student

Thomas Kleb
OST Student

Dated: Dec. 16, 2022

Nutzungsrechte

Vereinbarung

1. Gegenstand der Vereinbarung

Mit dieser Vereinbarung werden die Rechte über die Verwendung und die Weiterentwicklung der Ergebnisse der Semesterarbeit "Reverse Engineering Lab" im HS 2022 der FH OST von Thomas Kleb, Ronny Müller und Gianluca Nenz unter der Betreuung von Ivan Bütler geregelt.

2. Urheberrecht

Die Urheberrechte der im Rahmen der Semesterarbeit entwickelten Teile des "Reverse Engineering Lab" stehen den Studenten der Semesterarbeit zu.

3. Abgrenzung

Die Urheberrechte der von Compass Security und Hacking-Lab eingebrachten Vorleistungen in Form von Beispielen aus dem Hacking-Lab verbleiben bei Compass Security und Hacking-Lab.

4. Verwendung

Die Ergebnisse der Arbeit dürfen sowohl von den Studenten der Semesterarbeit, durch den Betreuer der FH OST wie auch von der Compass Security Holding oder einer ihrer Compass Tochterfirmen (Compass Schweiz, Compass Deutschland, Compass Cyber Defense, Hacking-Lab AG) uneingeschränkt nach Abschluss der Arbeit verwendet und weiterentwickelt werden.

Gianluca Nenz
OST Student

Ronny Mueller
OST Student

Thomas Kleb
OST Student

Ivan Buetler
Advisor

Dated: Dec. 16, 2022