Studienarbeit: Reverse Engineering Lab

Studiengang Informatik
OST - Ostschweizer Fachhochschule
Campus Rapperswil Jon

Semester: Autumn 2022

Autors: Gianluca Nenz
Ronny Mueller
Thomas Kleb

Project Advisor: Ivan Buetler
Release: E-Prints
Version: Monday 17th October, 2022

School of Computer Science
OST Eastern Switzerland University of Applied Sciences

Abstract

Contents

Abstract **i**

1 Project Idea **1**
1.1 Problem Domain 1
1.2 Learning Concepts 1

2 Management Summary **3**

3 Technical Report **4**

4 Project Documentation **5**
4.1 Project Plan 5
4.2 Risk Analysis 10
4.3 Project Monitoring 11
4.4 Personal Rapports 11

5 Meetings **12**
5.1 06-10-22 12

Directory **13**
5.2 Glossary 13
5.3 References 13
5.4 Table Directory 14
5.5 Illustration Directory 15

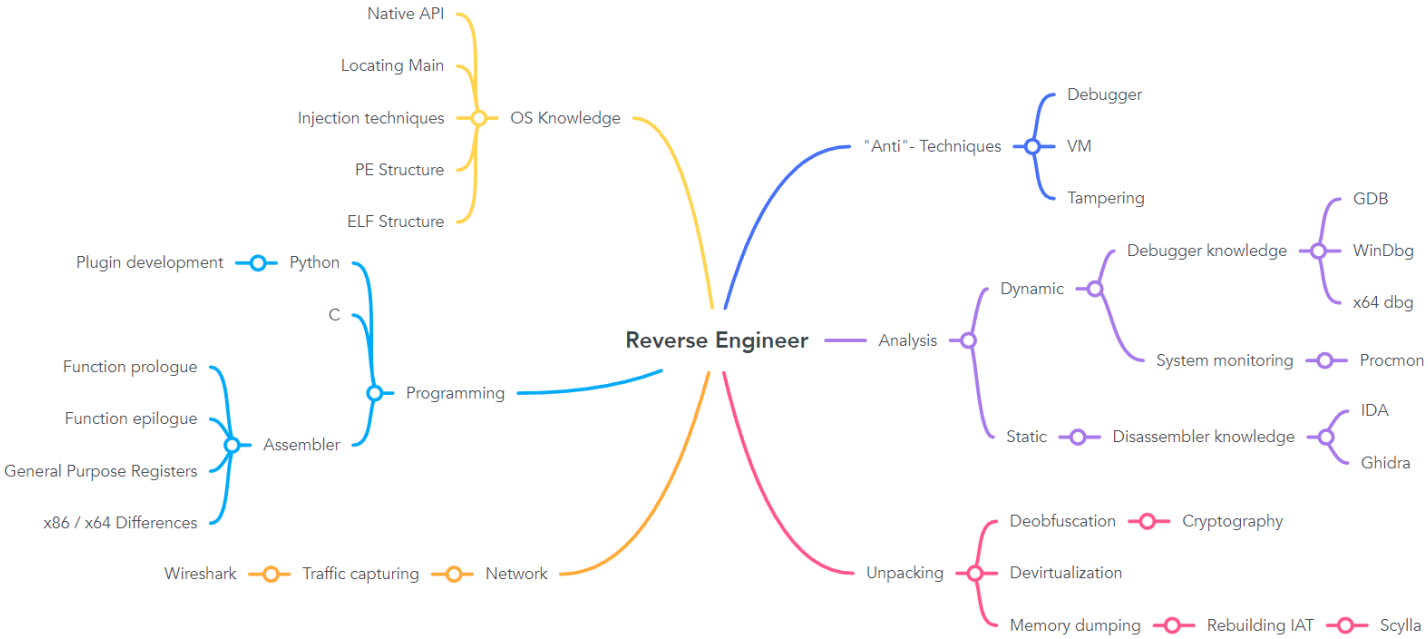
Appendix **16**
5.6 Eigenständigkeitserklärung 16
5.7 Nutzungsrechte 17
5.8 Danksagung 17

Chapter 1

Project Idea

1.1 Problem Domain

We defined in our opinion the most important domains which a Reverse Engineer has to have knowledge of.



1.2 Learning Concepts

Based on the Problem Domain from the last section we decided the most basic domains were programming, analysis and OS knowledge. So we decided that we are going to create the most labs and the first ones about these topics and then the later introduce the other domains in later Labs. We want to focus on Linux and Windows.

We also decided that we can expect for Students to already know about C, Python and some basic knowledge about Assembler because everyone has to have had BSYS which teaches about Assembler and C. Automation with python is also a module which now is in every sample curriculum at the OST.

Topic	Description
Refresher	Give the students some little refreshing on the key topics (Assembly)
Introduction to RE #1	Explain analysis approaches (Dynamic / Static) and install tools
Introduction to RE #2	Given a simple C file, students compile it and try to find a key (Static) (Find Main function)
Introduction to RE #3	Given a simple C file, students compile it and try to find a key (Dynamic) (learn GDB / x64)
First RE attempts	Given simple files compiled in several languages (PY, C#, C++) get flag
First keygen	Not only finding out the password but writing a keygen for the program
Harder CrackMes	Introduce new native API funcs / techniques like stack strings
Injection techniques	Explain some injection techniques
Dump memory	Explain how to dump memory off a given executable which uses a previously explained injection technique
"Anti"-Techniques	Introduce "Anti"-Techniques and provide program for students to bypass

Chapter 2

Management Summary

Chapter 3

Technical Report

Chapter 4

Project Documentation

4.1 Project Plan

4.1.1 Project Overview

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 hosted on the OST server.

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.

4.1.2 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.

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

Planning and Project Management

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

4.1.3 Organisation

Participants

The "Studienarbeit"-Team consists of three students: Gianluca Nenz, Ronny Mueller and Thomas Kleb. 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 cyber security 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 distributed 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	Ronny Mueller	Thomas Kleb
Meetings	Work 1	Protocols
Work 2	Work 2	Documentation
Work 3	Work 3	Work 3
Work 4	Work 4	Work 4

Table 4.2: Work Distribution per Student

4.1.4 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 comming project. After having a talk with the advisor it was decided to first find out the level of knowledge each student has to make it easier for the advisor to plan.

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 Learning-concepts are defined
5	14.10.2022	25.10.2022	Lab Conepts 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	25.10.2022	01.11.2022	—-
7	01.11.2022	08.11.2022	—-
8	08.11.2022	15.11.2022	—-
9	15.11.2022	22.11.2022	—-
10	22.11.2022	29.11.2022	—-
11	29.11.2022	06.12.2022	—-

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 labs 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	Buffer
13	13.12.2022	20.12.2022	—-
14	20.12.2022	23.12.2022	—-

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 solves the given exercises to find the level of RE knowledge.
M2 - Defining Problem Domains and Learnconcepts	13.10.2022	Problem Domains are defined, first Lernconcepts 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

Time Tracking

For time tracking the team has decided on using GitLabs integrated time tracking.

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.

Meetings

The team has meetings each tuesday to elaborate problems and check up on the progress. This meetings are also used to distributed the work load and the different parts of the sprint.
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.

CI/CD

Projectmanagment

The whole project will use a GitLab repository. To make sure no confusion happens a multirepo principle is used where one repository is for the documentation and protocols only and other are for code, information gathered, etc. Each student works on a branch and before pushing to the main branch has another student look into the code / text written.

4.1.5 Testing

Procedure

To be sure that our defined Labs have a high value to future students we wanted to define some Students who solve the Lab and give Feedback to us about their experiences. To have some sort of comparable Feedback we created a Google Forms which contains many Questions about their experiences completing our lab and what they think about it. After getting their Feedback we are going to tweak our Labs with their opinions in mind.

Test Subjects

Our Test Subjects are like us in their 5th Semester at the Ost in Rapperswil-Jona. They also study Computer Science.

Feedback

The feedback we got was the following:

Impact

Based on the Feedback in the last chapter we decided to tweak some Labs to better suit a beginner. The changes we made will be described in detail in the following Chapters.

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

General Risks

- Finding Testing Participants (severity: medium, probability: high)
  Mitigations: Early looking for backup person
  Actions taken: Found backup person
  New probability: low

- Being able to create reverseable Programs with additional difficulties. (severity: very high, probability: medium)
  Mitigations: being able to ask Ivan
  Actions taken: Asked for possible help
  New probability: low

- Irreparable corruption of git server. (severity: very high, probability: low)
  Mitigations: Weekly off-site git server backups
  Actions taken: Repository mirrored to GitHub
  New severity: low

- Lost work due to un-pushed work. (severity: low, probability: high)
  Mitigations: Frequent reminders to push changes by Scrum Master / Team

License Complications

- License Problems with Ghidra. (severity: high, probability: very low)
  Mitigations: No mitigations needed because it's completely Open Source.

- License Problems with IDA. (severity: high, probability: low)
  Mitigations: Providing a previously free version

4.3 Project Monitoring

Overview

Milestones

Time Tracking

4.4 Personal Rapports

Gianluca Nenz

Ronny Mueller

Thomas Kleb

Chapter 5

Meetings

5.1 06-10-22

Directory

5.2 Glossary

5.3 References

5.4 Table Directory

4.1 Time Investments 5
4.2 Work Distribution per Student 6
4.3 RUP: Inception Phase Planning 7
4.4 RUP: Elaboration Phase Planning 7
4.5 RUP: Construction Phase Planning 8
4.6 RUP: Transition Phase Planning 8

5.5 Illustration Directory

Appendix

5.6 Eigenständigkeitserklärung

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	Date
Ronny Mueller	Date
Thomas Kleb	Date

5.7 Nutzungsrechte

5.8 Danksagung