University of Tartu
Institute of Computer Science
Computer Science Curriculum

Geitrud Pank

Lab package: Mobile application security testing

Bachelor's Thesis (9 ECTS)

Supervisors: Dietmar Pfahl, PhD
Hina Anwar, PhD

Tartu 2022

Lab Package: Mobile application security testing

Abstract:

This thesis aims to create lab materials for the University of Tartu's course "Software testing" (LTAT.05.006) about mobile application security testing. This thesis gives background information on the course, on security testing in general and specifically mobile application security testing, describes the materials and lab execution using the materials, states, analyses the student feedback form and lab supervisors' feedback, and suggests future improvements. The lab took place in the spring semester of 2022.

Keywords:

Software testing, security testing, mobile application, lab package

CERCS:

P170 Computer science, numerical analysis, systems, control

Praktikumimaterjal: Mobiilirakenduse turvatestimine

Lühikokkuvõte:

Käesoleva bakalaureusetöö eesmärgiks on luua õppematerjali Tartu Ülikooli kursusele “Tarkvara testimine” (LTAT.05.006) mobiilirakenduse turvalisuse testimise teemal. Töö seletab lahti kursuse, üldiselt turvalisuse testimise ja täpsemalt mobiilirakenduse turvatestimise, kirjeldab loodud materjale ja praktikumi nende materjalide põhjal, esitab ja analüüsib tudengite ja praktikumijuhendajate tagasiside tulemusi ning soovitab materjalide sisu parandusi. Õppematerjal võeti kasutusele 2022. aasta kevadsemestril.

Võtmesõnad:

Tarkvara testimine, turvatestimine, mobiilirakendus, praktikumimaterjal

CERCS:

P170 Arvutiteadus, arvutusmeetodid, süsteemid, juhtimine

Table of Contents

Introduction

Software testing is an essential part of the software development process, which is used to check whether the software meets all of the standard requirements. The University of Tartu has a course named "Software Testing" (LTAT.05.006), which gives students an overview of different testing methods and techniques.

Security testing is a testing method used to spot vulnerabilities in a development, for example, a mobile application. Eliminating the possibility of security vulnerabilities before the launch of an application is vital for the security of user data and the company's reputation. For this reason, it is essential to teach students the most common vulnerabilities and how to develop a secure application.

This thesis aims to create a lab package for an existing lecture on security testing in the course. In the previous years, guest lecturer Kristiina Rahkema gave a lecture on the security testing of mobile applications, but there was no lab package on the topic. The lab package contains separate instructions for students and lab supervisors, lab slides for supervisors, homework tasks, and a grading scheme. The created homework teaches students the basics of security testing and the most common security vulnerabilities found in a mobile application. In addition, gives them an overview of Android Studio and how to emulate mobile applications.

This thesis is composed of four chapters. The first chapter gives background information about the "Software Testing" course and software testing, specifically security testing of mobile applications. The lab design chapter introduces the homework materials, tools and tasks, and the grading schema. The third chapter gives an overview of the execution of the lab. Chapter four contains the feedback of students and teaching assistants, the analysis of the feedback, and immediate and future improvement propositions.

1 Background information

This chapter gives an overview of the information needed to create a lab package on mobile application security testing in the "Software Testing" course.

1.1 Course information

The lab is designed to fit into an existing 6 ECTS course named "Software testing" [1] (LTAT.05.006) at the University of Tartu. The target group is second-year computer science bachelor students as it is compulsory to take Software Engineering as a prerequisite. In addition to introducing a variety of different testing methods and strategies, the course gives an overview of test planning and documentation, defect estimation, and the role of testing in the software development process.

The course covers 11 topics as of the year 2022:

1. Debugging
2. Basic Black-Box Testing
3. Combinatorial Testing
4. Basic White-Box Testing
5. Random Testing
6. Automated Web Application Testing
7. Visual GUI Testing
8. Security Testing
9. Mutation Testing
10. Static Code Analysis
11. Document Inspection and Defect Prediction

Course's assessment consists of 11 homework assignments, weekly quizzes, and the final exam, where homeworks are worth 60%, weekly quizzes 10%, and the final exam 30% of the final grade. The student has to get at least 50% of the possible marks from labs and the final exam to pass the course.

1.2 Software security

Software security [2] is the process of developing and testing software to make sure it continues to function correctly under malicious attacks and exploits. The defensive mechanisms are often added to the software system in the later stages of the development process. This can contribute to new security vulnerabilities emerging into various stages of software development lifecycles. It is vital to target the vulnerabilities by addressing security concerns from the start to prevent this.

1.3 Security testing

Data safety and security [3] are a big concern for developers because a vulnerable application allows third parties to access the user's personal information. Security testing is often fundamentally different from traditional testing because it emphasises what an application should not do rather than what it should do. In addition, users do not usually try to search out software bugs intelligently, but they do try to search for security vulnerabilities. If they succeed, they can cause problems for other users, who may be adversely affected.

1.4 Mobile application testing

Mobile application testing [3] is necessary to develop a bug-free application as every device needs to be tested for its functionality, usability, and performance. With mobile applications, the major challenge is the diversity of devices with different features and capabilities. Other challenges include extra hardware elements creating bugs to software or hardware, making sure the usability of the application is unambiguously understandable for every user, and operating systems having different versions for different types of devices.

1.5 Mobile application security risks

Security testing is an integral part of a mobile application testing process [4] due to devices having access to networks with different security levels. Security test activities are performed to identify potential security vulnerabilities within the system. It is crucial to address the vulnerabilities before deploying the application, as security vulnerabilities can drastically impact a company's reputation and expenses.

OWASP [5] creates a list of the year's top 10 most common mobile application security risks. The year 2021 list includes broken access control, cryptographic failures, SQL injection, insecure design, security misconfiguration, vulnerable and outdated components, identification and authentication failures, software and data integrity failures, security logging and monitoring failures, and server-side request forgery.

2 Lab design

This chapter gives an overview of all the tools used in the homework, homework tasks, materials made for students and lab supervisors, grading schema, and lab schedule.

2.1 Test application

Finding a suitable test application was the first task when creating a lab package. To find the best app to use for the security testing of mobile applications homework, two different applications were tried, DVIA and DIVA.

2.1.1 DVIA mobile application

DVIA (Damn Vulnerable iOS App) [7] is an iOS mobile application created for students and mobile security enthusiasts to test their iOS testing skills in a legal environment. DVIA has both Swift and Objective-C versions; the current version is written in Swift. The app is structured as a list of exercises that cover a variety of different vulnerabilities. The list includes local data storage, jailbreak detection, excessive permissions, runtime manipulation, phishing, broken cryptography, and other vulnerabilities. The creator of this app updates the app code regularly, and it is possible to propose fixes and new additions to the list of exercises.

2.1.2 DIVA mobile application

DIVA (Damn Insecure and Vulnerable App) [8] is an intentionally vulnerable Android application. It aims to teach students, software developers, and testers how to spot security vulnerabilities, often resulting from inadequate code writing skills. The app was created in 2016 when the author Aseem Jakhar came up with the idea to develop an Android version of the iOS DVIA app.

DIVA is written mainly in Java and covers some of the most common Android security vulnerabilities. The app has 13 different exercises, and it covers five topics: insecure logging, hardcoding issues, insecure data storage, input validation issues, and access control issues. It is possible to contribute to the app by emailing the creator with details of vulnerabilities they could implement in future versions.

2.1.3 Comparison of DVIA and DIVA

Both applications are structured as a list of exercises with different security vulnerabilities. The DVIA application would be a better choice in terms of content as it covers 15 security risks, ten more than the DIVA app. The problem with DVIA is that iOS applications can only be emulated in Xcode for free, which only works on the macOS operating system. It is written in Swift, which the students might not be familiar with. The DIVA application is written in Java and can be emulated in Android Studio, supported on all operating systems. When considering these arguments, the better choice is the DIVA app, as it can be used on all operating systems and is written in a programming language the students have already learned in earlier courses.

2.2 Android Studio

The fact that students can run the application on an emulator, which can be used to test the functionality and usability of applications, and DIVA is an Android application is the main reason Android Studio [9] is best suited for students to use. Android Studio is the official IDE (Integrated Development Environment) for the Android platform and is used to make most of the apps Android users use daily. It is developed by Google, and it supports Java and Kotlin programming languages. It is possible to run the application in two ways: on an emulator or a physical device connected to the computer. The user will then also be able to 'debug' the program as it runs and get feedback explaining crashes etc., so that users can more quickly solve the problem.

2.3 Preparing DIVA for use

An email was written to the creator of the DIVA application to ensure the application could be used for the homework. The creator allowed it as long as his name was stated in the read.me file in the application code.

The DIVA app had outdated Gradle dependencies, and it did not build successfully when following the guide written by the author of the application. The code needed updating in three files: *build.gradle*, *gradle.properties*, and *gradle-wrapper.properties*. In *build.gradle*, the repositories needed changing, jCenter was switched to mavenCentral. In *gradle-wrapper.properties*, the distribution URL was changed to Gradle version 7.2 before the app was using Gradle version 2.4. In *gradle.properties*, a line allows the application to

use deprecated NDK, which was commented out as NDK is not supported by Google Play Services starting from November 2022 [10].

The application was uploaded to GitHub (see appendix I) to make the setup more straightforward for students. As there were a number of changes made in the code, it was decided that students will get the code that does not need modifying before setup.

2.4 Homework tasks

The tasks were chosen after completing all exercises in the app, analysing the importance of each one, and matching the exercises with the correct OWASP security risks. Afterwards, all unsuitable exercises were eliminated that were either not covered in the lectures or were too difficult for the given time frame of the homework. Ultimately, five tasks were chosen that are expected to be feasible to complete by all students. Some exercises have elements that may not be familiar to students; therefore, hints were added to the description of those tasks.

The first task is about insecure logging. The student needs to enter a random credit card number and find where and how the credit card number is logged. The main idea of this task is to show how it is important to make sure the application doesn’t log any sensitive information as other applications may gain access and read logs to steal sensitive data.

The second task is about hardcoding issues. In order to get the correct output after entering a vendor key, the student needs to find what is hardcoded in the app and where. This task aims to remind students that if they need to hardcode something into the app, they must ensure it doesn't let third parties access sensitive information, such as passwords.

The third and fourth tasks are about insecure data storage. Both ask the student to enter a random 3rd service user name and password and find out where/how the credentials are stored and the vulnerable code. The purpose of these tasks is to show how easy it could be to get access to users' sensitive information if developers don't think about encrypting user data and assume third parties will not have access to a mobile's file system and application files.

The last task is about input validation issues. They need to try to access all user data in the app by entering a malicious search. With this task, the students learn how important it is to make sure data is correctly validated as input validation prevents improperly formed data from entering an information system.

2.5 Grading

The level of difficulty is not the same for all tasks, but it does not vary enough to give different amounts of points for tasks. In addition, this ensures students of all skill levels will still be able to get points from the homework. Every homework is set to give a maximum of 9 points, and there are five tasks in this homework, where every task gives 1.8 points.

The grading criteria are as follows:

- Up to 0.7 points for finding the bug, where 0.3 points are for the screenshot of the output and 0.4 points are for the explanation of the bug.
- Up to 0.5 points for the OWASP security risk, where 0.2 points are for the choice of the security risk and 0.3 points are for the justification.
- Up to 0.3 points for the screenshot of the faulty line of code.
- Up to 0.3 points for the explanation of how to fix the vulnerable line of code.

Finding the defect gives the highest amount of points. The explanation of the bug is more important than the screenshot, because this way, the lab supervisor will see if the student has actually understood that it is a security risk. The OWASP question was added to get the students to read through the risks and find out if they understand and can explain how a certain vulnerability is a security risk. That is why the justification gives a higher amount of points than the choice of the risk. Stating the faulty line of code and explaining why it is a vulnerability gives the same amount of points because stating the line of code and suggesting a fix are the same in volume.

2.6 Lab materials

The homework instruction file includes an introduction to mobile application security testing, a tool setup guide, five homework tasks, a grading schema, and a sample report. For each task, the student needs to report the input and, in some cases, the output on the emulator, the correct OWASP security risk that correlates to the found bug, screenshot of the faulty line of code, and explain why they think it is a security risk and how to fix it. Students complete the homework in pairs and the expected total time spent on it, as is for all homeworks in the course, is 10 hours.

Separate instructions were created for students and lab supervisors (see appendix I). The file for supervisors contains answers and error solutions for the tasks in addition to everything that is in the student homework document. In addition, a model solution was created for the lab supervisors to use as a reference when grading.

2.7 Lab schedule

The lab session has three main parts:

- Introduction to security testing of mobile applications.
- Briefly go through the homework tasks and explain the main idea of every task.
- Set up the tools and successfully run the DIVA application.

The setup will more likely take up most of the lab as the students need to clone the app, download Android Studio, build and sync Gradle files, set up the emulator, and lastly, build and run the DIVA app. If the supervisor finishes before 90 minutes, the students have the option to start their homework in the lab session.

3 Lab execution

The purpose of this chapter is to report how the mobile security testing labs were executed. The labs took place on April 19 and 20 and were carried out in 6 groups. One lab was attended on April 19, and the lab execution was observed. Interviews were conducted with the other lab supervisors the week after lab execution to get an overview of all other labs as well.

3.1 Observations made in a lab

There were 13 students present in the lab. The lab started with the lab supervisor giving an overview of the topic and briefly going through the homework tasks, which took approximately 15 minutes. Next, the students began to set up the application. The setup was estimated to take about 45 minutes, after which they could start working on their homework. A couple of students got the application working under 30 minutes, but it took the estimated time for most. The emulator opened in Android Studio for most students, and the lab supervisor showed them how to open it in a separate window. The reason was that it is not resizable on the platform, and the text in the application is too small to read on a laptop. After the setup, two students left the lab and others started doing the homework assignment.

Students asked some questions as well, most of them were about the setup. The students who started with the homework were asking about the first task. They asked about the fix, whether the correct solution would be to delete the line of code logging the credit card number, hide the number completely or leave some numbers to be seen in the log.

One problem was not waiting for the emulator to finish building and trying to already click on applications in the emulator. When the students closed the emulator, clicked “Run” again and waited until the emulator was running correctly, everything worked. Another thing students with macOS needed to keep in mind was accessibility issues. There was one student in the lab with a MacBook and he had to separately allow Android Studio to have access to files and folders in the computer. In addition, the student with a MacBook could not see the emulator at first. The problem was that the emulator runs on a separate platform by default in macOS and the student had Android Studio on full screen. The full screen blocked the student from seeing the emulator as it was running in the background.

3.2 Observations made by supervisors

According to the lab supervisors, other labs had 12 to 15 students as well, with the exception of one lab where three students attended. For all labs, the introduction took about 15 minutes, after which students started to set up the application. One supervisor created her own slides to use in the lab, where she rearranged the order of the lab activities. She advised the students to start downloading Android Studio before going through the topic presentation, which shortened the setup time by 10 minutes. Students asked additional questions about mobile application development and Android Studio, as most had not used it beforehand. Two lab supervisors completed two of the tasks in the lab with students, which left the students with three tasks to complete on their own.

Some students had errors when building the application and needed assistance with fixing them. One problem was using an older version of Android Studio, but it was stated in the document that the most recent version is required. The build was successful after updating the platform and deleting the cache. Other problems were related to either missing a step in the instructions or needing to change a configuration in Android Studio. Lab supervisors had gone through the setup; therefore, they had encountered some problems themselves and knew the solutions to errors.

4 Evaluation

This chapter gives an overview of the feedback given by students and lab supervisors, an analysis of the feedback, and improvements already made and planned to be made.

4.1 Student feedback

The following week from lab execution, lab supervisors gave the students 5 to 10 minutes to fill out an online feedback form (see appendix II). Around 60 students attended the lab, and 23 of them filled out the questionnaire. The form had one question, seven statements, and an optional feedback field, where the students could leave additional comments about the lab package. Students had the option to either answer anonymously or state their names. The form contained the following:

1. The goals of the homework were clearly defined and communicated
2. The instructions for the homework were clear and helpful
3. The structure of the homework tasks was clear and helpful
4. How long did the setup take for you?
5. The tools used in the homework were appropriate and useful
6. Compared to the previous homework, this assignment was more difficult
7. What I learned while completing the homework is relevant for working in the software industry
8. Overall, the homework was interesting and useful

Students could rate the statements from 1 to 5, 1 meaning they disagree entirely and five that they completely agree with the statement. The question about setup time duration had four options that the students could choose from: up to 30 minutes, 30 minutes to 1 hour, 1 hour to 2 hours, and more than 2 hours. The summary is visible in Figure 1.

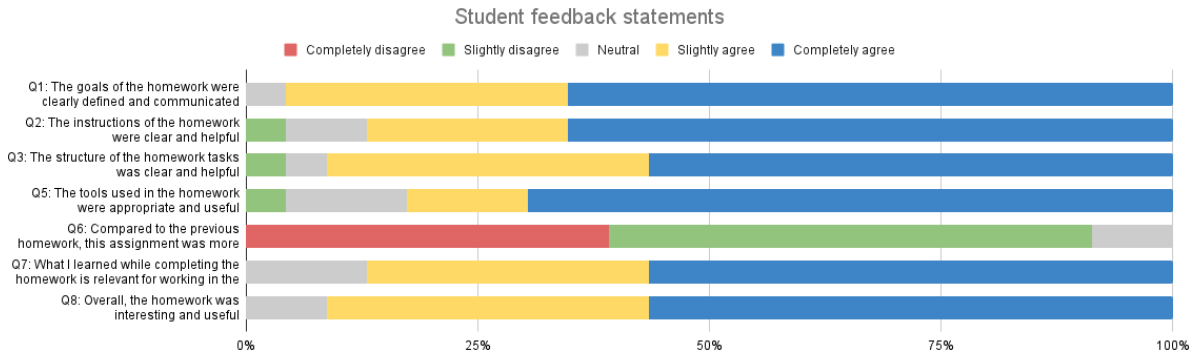

Figure 1. Statistics of statements in the student feedback form.

Over 95% of the students agreed that the homework goals were clearly defined and communicated. One student stayed neutral, and no one disagreed with the statement. 19 students out of 23 found the instructions of the homework to be clear and helpful, two were neutral, and one slightly differed. 90% of the students agreed the structure of the homework tasks was clear and helpful. In conclusion, on average, about 90% of the students thought the homework document was clearly defined, and the instructions and structure were appropriate and helpful.

There were 16 students who agreed that the tools used in the homework were appropriate for this lab, three stayed neutral, and one slightly disagreed. This was asked to know if the students liked the DIVA application and testing in Android Studio.

Students were also asked about the difficulty of tasks compared to previous homeworks. There were 39% of students who completely disagreed that it was more difficult than previous homeworks, 52% slightly disagreed, and 9% stayed neutral. This means the homework was easier than previous ones, which was expected, but because it was not known how long the setup would take for an average student, the task list was limited to five.

Precisely 87% agreed with the statement about the homework being relevant in the software industry, and 13% stayed neutral. In addition, they were asked to state whether the homework was interesting and useful, which 91% said it was.

A question was asked about setup time duration, see Figure 2. It was added to determine whether it would be possible to add a task to the lab in the form of an exercise from the application. The setup took up to an hour for 71% of students, one hour to two hours for 17%,

and more than two hours for 12% of students. The majority got the application running in an hour, which means a lab task should be added as most students still had at least 45 minutes left in the lab.

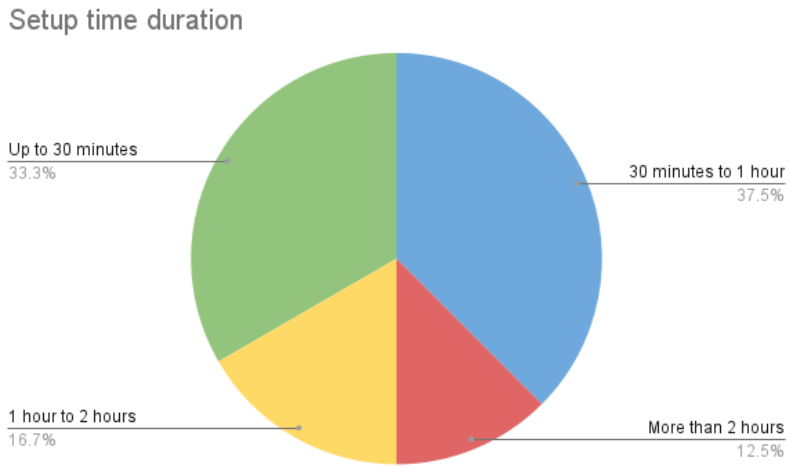

Figure 2. Summary of setup time duration.

There were eight responses in the additional feedback field. Two comments were about using SQLite; they had trouble figuring out how to use it properly and said there should be more instructions on how to use it in the document. At the same time, one student added that it wasn't such a big problem because it was entirely possible to search for instructions online. Most of the comments were about matching the security vulnerability with the correct OWASP security risk. Some said that explaining why they think a specific risk is the correct one was difficult and not always straightforward. Two students wrote that they liked the homework very much and would like to see more lab packages of such type.

4.2 Lab supervisor feedback

Every lab supervisor was interviewed for 10 to 15 minutes to get feedback on the lab package. They were asked a series of questions about the materials, whether the instructions were clear, the tools helpful, and whether tasks were suitable for students. They all agreed that there are no significant changes to be made but commented on some aspects that need improving.

One assistant advised that troubleshooting should be added to the homework, where common errors and solutions could be copied. Also, in task 1, there should be a hint added that the correct answer to fixing the bug is not hiding the credit card number completely or not logging the number at all. This was something that many of the students asked from the lab supervisor and answered wrong in the report. Another assistant said to go over the document again and fix some misleading wordings in the instructions.

In addition, they said that a lab exercise should be added, as many students got the application running within an hour and had the last 45 minutes to already start with the homework. Two assistants completed the first two tasks in the lab, which was not expected as the instructions did not guide them to do so. This created an unfair situation because, in some labs, the assistant did not complete any tasks. Next year, it is advised for the lab supervisors to decide together beforehand how much they will finish in the labs.

When talking about the usefulness of the lab, one assistant spoke about how students were very interested in Android Studio and emulating an app. They asked many additional questions about app development and how to create an app themselves using Android Studio. The assistant stated that in the future, there should also be a lab package more about mobile application testing in general as well because students were very interested in it.

4.3 Discussion

Both students and lab supervisors gave valuable and constructive feedback on the lab package. Some improvements have already been made, and some should be made before next year.

4.3.1 Feedback analysis

In general, both lab supervisors and students gave primarily positive feedback. The supervisors made valuable, constructive comments about the materials, which will help to improve the lab. Negative feedback about the OWASP question was expected because there were multiple correct answers for a couple of tasks which confused some students. The lab package is counted as a success as there were no drastic problems with the execution, and a couple of students and lab supervisors even specifically stated how much they liked the homework

4.3.2 Immediate improvements

A student pointed out that in task 5, the output does not show all three users; there were three dots shown on the screen instead of the users. Lab supervisors figured out that the problem was with the system image said to be used in the instructions. In reality, the students should use system image R as that shows the asked output of task 5, but the instructions required system image S to be used. System image S does not show all three users, and therefore students could not take a screenshot of the required output. The instructions were corrected by lab supervisors during the week when students were completing the homework.

There were also some more minor changes suggested in wording during the week when students completed the homework and the following week when lab supervisors were grading the assignment. It is stated in the grading schema that 0.3 points are given for the screenshot of the output and 0.4 points for the explanation of the bug. In task 2, task 3, and task 4, it is not stated in the report requirements to explain why the found bug is a vulnerability. This caused confusion amongst supervisors as some students reported everything that was required in the grading criteria, others what was said in the report requirements under each task. Lab supervisors decided to give 0.7 points for the screenshot and not take off points for missing explanations for these tasks.

4.3.3 Future improvements

There are some aspects of the lab package that should be improved. The most important is to add a lab task which could be the first task in the homework about logging. Lab supervisors stated that this task was the easiest and could be switched out to be a lab task and create a new task for the homework instead. As it came out in the student feedback as well, the homework is easier than previous ones, meaning the new task could be made to be more complex.

Another issue was something that came up in a lab. In the instructions about emulator configuration, it is said that the student has to click "Next" to continue to the next page, but it is missing in one place. This caused some confusion because the student did not know where to configure the next part of the instructions. The instructions need to be analysed again to check if there is anything more that needs to be explained more thoroughly.

Lab supervisors said that the setup took a long time because of Android Studio's downloading time. The lab execution order should be changed, and students should start downloading at the beginning of the lab. The downloading time is 10 minutes on average, which means the assistant could finish introducing the lab, and Android Studio would be downloaded in the meantime.

Something that should be checked every year before the start of the course is the setup guide. It is possible that something in the code becomes outdated in a year or some dependencies need to be changed. It is advised to make a zip file of the application code and homework instructions, because during the 2022 spring semester it is in GitHub. There is also the possibility of combining everything needed to run the DIVA application into a Docker container.

Conclusion

The aim of this thesis was to create a lab package for the "Software Testing" (LTAT.05.006) course at the University of Tartu. The materials were made for an existing lecture on the security testing of mobile applications. Mobile application security testing is a testing technique to assure the application is not at risk for security vulnerabilities such as data leakage, insecure logging, etc.

The result of this thesis was the lab materials created for the course. The package contained homework instructions for both students and lab supervisors, lab slides, and a model solution for lab supervisors. Feedback was collected from students and lab supervisors.

Student feedback was collected from 23 students with an online questionnaire where they were asked to rate statements about the lab package. About 90% of students agreed that the homework was beneficial, goals and instructions were clear, and what they learnt from the homework is useful in the software industry. Also, they had the option to leave additional comments about the homework. A couple of students said they liked the homework a lot, and some wrote about the problems they encountered while completing the homework.

Lab supervisors were interviewed, and they were asked about the execution of the lab and feedback on the materials. They all said there are no significant changes needed but pointed out that there should definitely be a lab task for next year and make a couple of steps clearer in the instructions. Some minor corrections were made in the materials during the week of the lab package by lab supervisors. In addition, using the constructive comments from students and lab supervisors, future improvements were also suggested.

References

[1] "Software testing" course information for the 2022 spring semester. https://ois2.ut.ee/#/courses/LTAT.05.006/version/95ce8971-8536-d390-672d-9f8010e335b7/details (02.04.2022)

[2] Nabil M. Mohammed, Mahmood Niazi, Mohammad Alshayeb, Sajjad Mahmood, "Exploring software security approaches in software development lifecycle: A systematic mapping study", Computer Standards & Interfaces, Volume 50, 2017, pp. 107-115, https://doi.org/10.1016/j.csi.2016.10.001.

[3] Michael, C.C., van Wyk, K, Radosevich, W. https://www.cisa.gov/uscert/bsi/articles/best-practices/security-testing/risk-based-and-functional-security-testing (28.04.2022)

[4] B. Kirubakaran and V. Karthikeyani, "Mobile application testing — Challenges and solution approach through automation," *2013 International Conference on Pattern Recognition, Informatics and Mobile Engineering*, 2013, pp. 79-84, doi: 10.1109/ICPRIME.2013.6496451, https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6496451&isnumber=6496435.

[5] Mobile App Security Testing Guidelines. https://www.softwaretestinghelp.com/mobile-app-security-testing-guide/ (08.04.2022)

[6] OWASP Top 10 Web Application Security Risks. https://owasp.org/www-project-top-ten/ (06.04.2022)

[7] Prateek147. DVIA-v2 app. https://github.com/prateek147/DVIA-v2 (10.04.2022)

[8] Jakhar, A. DIVA app. https://github.com/payatu/diva-android (10.04.2022)

[9] Mullis, A. Android Studio tutorial. https://www.androidauthority.com/android-studio-tutorial-beginners-637572/ (10.04.2021)

[10] November 2022 deprecated camera methods removal for ARCore SDK. https://developers.google.com/ar/develop/ndk-camera-deprecation#:~:text=In%20November%202022%2C%20Google%20Play,Google%20Play%20Services%20for%20AR. (30.04.2022)

Appendix

I. Lab Materials

Student instructions

Student homework instructions PDF file

https://courses.cs.ut.ee/2022/SWT2022/spring/uploads/Main/SWT2022L8-Instr-V1.01.pdf

Lab supervisor instructions

Both homework instructions and the model solution contain the correct answers to the homework tasks. In addition, the homework instructions include step-by-step guides of the correct solutions.

- Homework instructions PDF file
- Model solution report PDF file
- Lab slides

For confidentiality reasons, the lab supervisor's materials are not public but can be made available on request.

Homework source code

https://github.com/geitrudpank/DIVA-app

II. Feedback Form

Feedback to Homework 8 - Security testing of mobile applications

*Required

1. Name (optional)

2. Q1: The goals of the homework were clearly defined and communicated *

Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

3. Q2: The instructions of the homework were clear and helpful *

Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

4. Q3: The structure of the homework tasks was clear and helpful *

Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

5. Q4: How long did the set-up take for you? *

Mark only one oval.

- ◯ Up to 30 minutes
- ◯ 30 minutes to 1 hour
- ◯ 1 hour to 2 hours
- ◯ More than 2 hours

6. Q5: The tools used in the homework were appropriate and useful *

Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

7. Q6: Compared to the previous homework, this assignment was more difficult *

Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

8. Q7: What I learned while completing the homework is relevant for working in the software industry *

Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

9. Q8: Overall, the homework was interesting and useful *

Mark only one oval.

	1	2	3	4	5	
Completely disagree	◯	◯	◯	◯	◯	Completely agree

10. Here you can add additional feedback

III. Licence

Non-exclusive licence to reproduce thesis and make thesis public

I, Geitrud Pank,

1. grant the University of Tartu a free permit (non-exclusive licence) to reproduce, for the purpose of preservation, including for adding to the DSpace digital archives until the expiry of the term of copyright, my thesis

   Lab package: Mobile application security testing,

   supervised by Dietmar Pfahl and Hina Anwar.

2. I grant the University of Tartu a permit to make the thesis specified in point 1 available to the public via the web environment of the University of Tartu, including via the DSpace digital archives, under the Creative Commons licence CC BY NC ND 4.0, which allows, by giving appropriate credit to the author, to reproduce, distribute the work and communicate it to the public, and prohibits the creation of derivative works and any commercial use of the work until the expiry of the term of copyright.

3. I am aware of the fact that the author retains the rights specified in points 1 and 2.

4. I confirm that granting the non-exclusive licence does not infringe other persons' intellectual property rights or rights arising from the personal data protection legislation.

Geitrud Pank

10/05/2022