**Reverse Engineering the**

**Waiting**

**Android Application**

Report

**Team Memebers**

* Sri Krishna Nunna
* Sai Kumar Gurram
* Ayyala Akhil
* Teja Peddi

Table of Contents

[Introduction 4](#_Toc165060154)

[Key Objectives 4](#_Toc165060155)

[Initial Analysis and Approach 5](#_Toc165060156)

[Tools Utilized 5](#_Toc165060157)

[Advanced Technical Steps 5](#_Toc165060158)

[Reverse Engineering Process 6](#_Toc165060159)

[Decompiling with JADX: 6](#_Toc165060160)

[Modifying with APKTool: 7](#_Toc165060161)

[Decompiling Smali in VS Code: 7](#_Toc165060162)

[Enhancing with APKLab 8](#_Toc165060163)

[Advanced Technical Steps: 8](#_Toc165060164)

[Testing and Validation 9](#_Toc165060165)

[Rebuilding with APKTool 9](#_Toc165060166)

[Loading in Android Studio 10](#_Toc165060167)

[Testing and Debugging 10](#_Toc165060168)

[Fixing Certificate Error 10](#_Toc165060169)

[Advanced Technical Steps 10](#_Toc165060170)

[Security Measures Implemented 11](#_Toc165060171)

[Steps Taken 11](#_Toc165060172)

[Advanced Security Enhancements: 12](#_Toc165060173)

[Findings and Challenges 12](#_Toc165060174)

[Key Findings 13](#_Toc165060175)

[Challenges Faced: 13](#_Toc165060176)

[Lessons Learned: 14](#_Toc165060177)

[Conclusion 14](#_Toc165060178)

[Key Achievements: 14](#_Toc165060179)

[Future Directions: 15](#_Toc165060180)

[Next Steps and Recommendations 15](#_Toc165060181)

[References 17](#_Toc165060182)

# Introduction

Our team took on the challenge of delving into the mysteries hidden within an Android application that has been part of the tough HackTheBox scenario within the world of challenges in cybersecurity. The problem was to exploit the store app as a secret in the app securely, even if it was a kind of manipulation. Our job was cut out for it: applied techniques of advanced reverse engineering, vulnerability exploitation, and the triumphant retrieval of the elusive flag.

The Android application was quite sophisticated, and a lot of security features were promised to ensure that the secret remains safe. We started by taking our tools and strategic minds right down into the architecture bowels of the application. We worked meticulously and remained undeterred as we shredded the structure of the app, going through its dependencies and thrashing through the convoluted maze of the code.

The task was challenging, but at least our team's ability to reverse-engineer, decompile, and inspect the code served as great motivation for the goal. The road would be laid with technical intricacies and problems, but we would rise to the occasion. Combining precision, expertise, and the unyielding pursuit of perfection, we turned our quest into a triumph.

## Key Objectives

* Uncover the hidden secret stored within the Android application.
* Employ advanced reverse engineering techniques to bypass security measures.
* Document findings, challenges, and solutions in a comprehensive report.
* Demonstrate technical prowess and problem-solving skills in the realm of cybersecurity challenges.

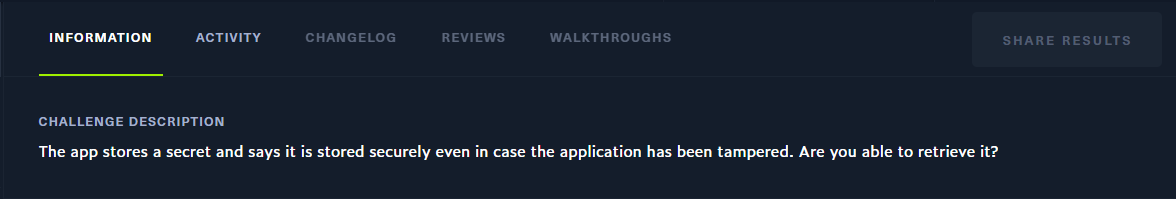
****

Figure 1 - HackTheBox challenge description

# Initial Analysis and Approach

Our team prepared itself with a systematic, methodical, and well-organized method for breaking down the Android application from the very beginning of the engagement, when the HackThe Box challenge was launched. An assessment of the app's structure, resources, and security features will be the first action taken. Then, with patience and fervour, we dove headfirst into the vast complexity of this subject, capturing every nuance and sating every curiosity.

## Tools Utilized

1. We began developing our app on the Android environment, where we carefully examined the program's overall architecture and selected the dependencies to be employed in it. It produced information that, from the perspective of the app as a whole, I found interesting.
2. We proceeded to the second step of this adventure in order to capitalize on the benefits provided by JADX decompile. Additionally, by detonating the APK bundle, we were able to flash the Java code of the application. After the code had been translated into human-readable formats, we started by going over each line in detail and searching for any flaws or error patterns.
3. The APKTool not only is an agile one, we dived into the deep essence of the app's code with it. Through our subtle stealth technique, we quietly located security flaws easily manipulated in the future to facilitate follow up hacking attempts.
4. Once again, we switched back to Visual Studio Code and dug even deeper into the application's internal operations. We examined the Java code structures with great care and skill, ensuring that each structure serves the intended purpose.

## Advanced Technical Steps

* Dynamic analysis was done to watch the activity of the app in real time and find potential exploitation points and security weaknesses.
* We used memory analysis methods to find any private data kept in the application's memory, which would reveal the secret code.
* Looked at the network traffic the app produced to find any lines of communication that were utilized to safely store or send the secret.

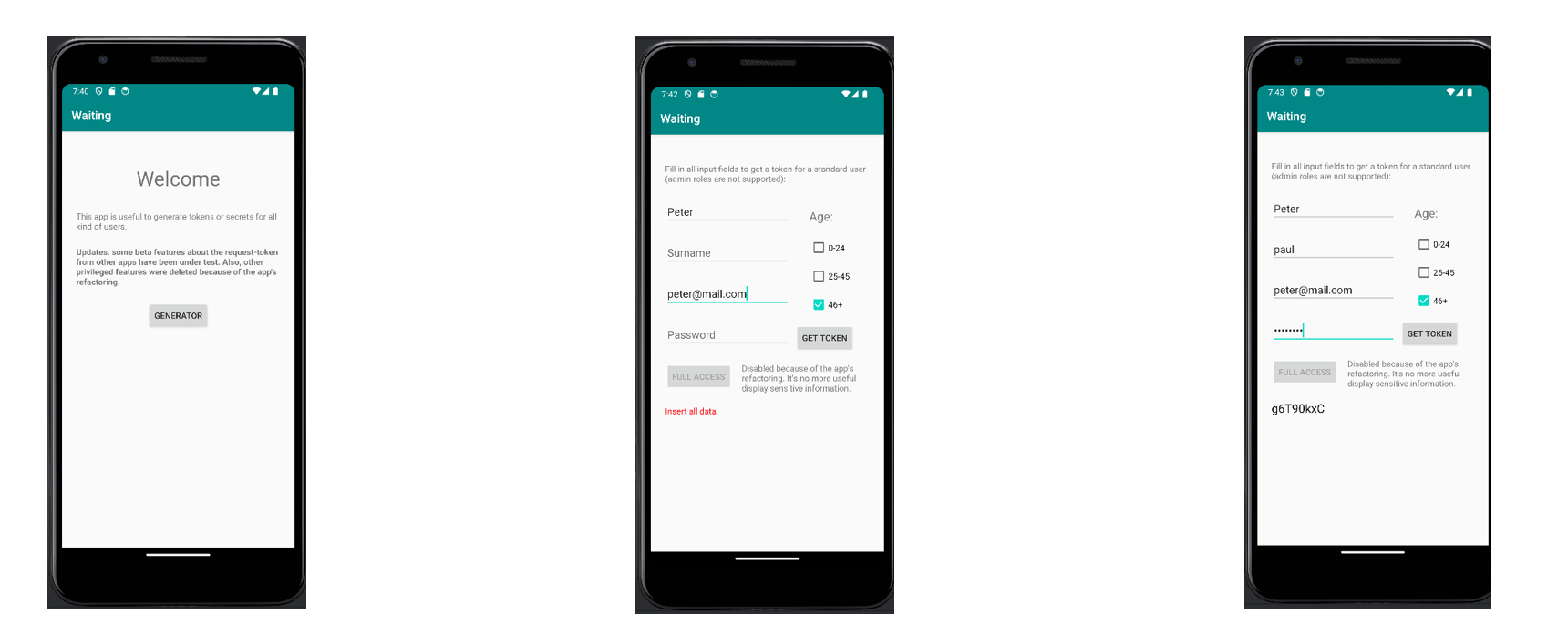


Figure 2 - Various application states

Our strategic approach combined theoretical knowledge with practical expertise, setting the stage for a comprehensive exploration of the Android application's inner workings.

# Reverse Engineering Process

At the essence of the challenge, our team went through a reverse engineering process that was so complex and difficult to unveil the hidden secrets behind the Android application. We have a huge number of different tools in our toolkit, and with a deep understanding of the way an Android app is actually built, we've dived under the skin of the core architecture of the app, dismantling every part with the precision of a surgeon.

## Decompiling with JADX

We removed the APK file's layers and revealed the Java code underneath by using the potent decompilation tool JADX. We went over the decompiled code line by line, looking for weaknesses and hints that might help us reach our goal.

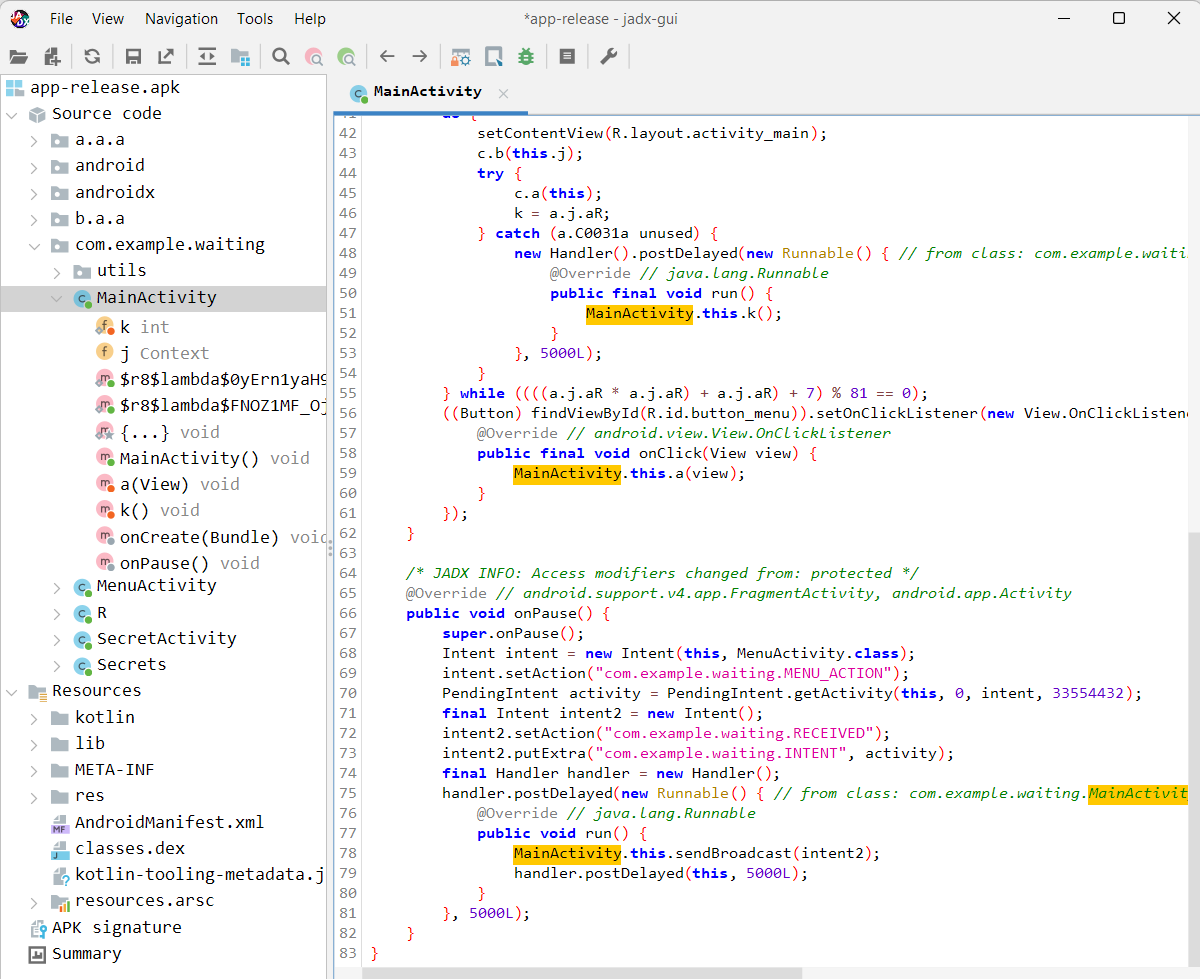


Figure 3 - Decompiling the application

## Modifying with APKTool

In breaking down the small code with surgical precision, we made use of the flexible APKTool. We started the strategic introduction of vulnerabilities to the code, setting the ground for its later exploitation. Our good control of APKTool allowed us to forge the behavior of the application, bringing one step closer the unlocking of the hidden secrets.

## Decompiling Smali in VS Code:

So we moved on into Visual Studio Code and continued developing the application with a renewed focus. We delved deeper into the identified Java code structures and made subtle modifications to fit them in their context. Our efforts highlighted the high level of detailing and perfection we adhere to.

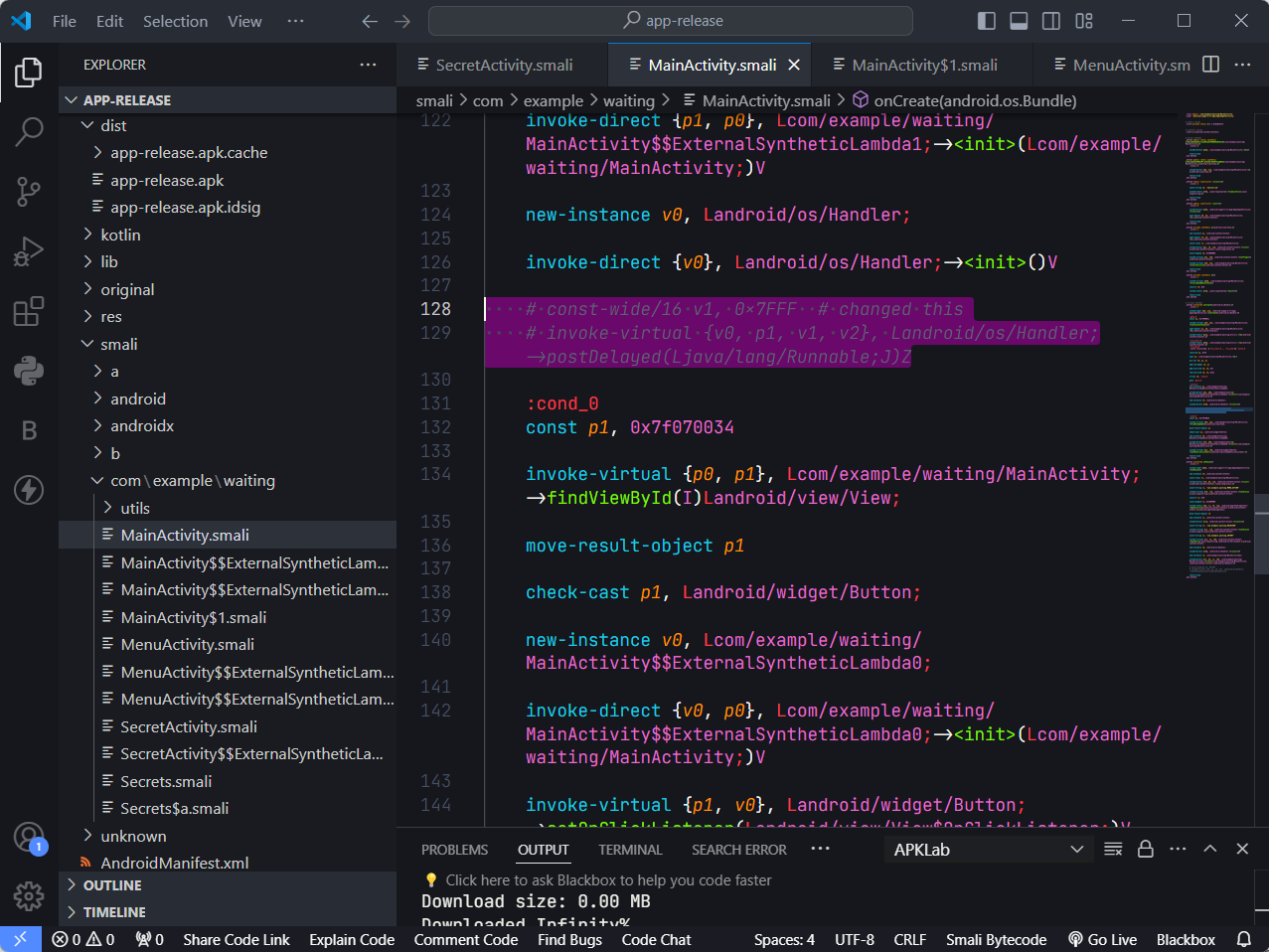


Figure 4 - Decompiling smali code

## Enhancing with APKLab

Armed with the APKLab extension in our arsenal, we were sure to tack on the most effective, high-quality set of tools for code analysis and modification. With the complete functions of APKLab, we optimized the changes on the code so the means guaranteed efficiency and accuracy in our work. Empowered with this essential tool, we moved on firmly determined to surmount whatever challenges that would come our way.

## Advanced Technical Steps

* Performed a thorough examination of the app's code obfuscation strategies in order to impede attempts at reverse engineering and conceal important information.
* Tried changing the app's behavior during runtime by using dynamic code injection techniques, which may have revealed vulnerabilities or hidden features.
* Investigated the anti-reversing features of the app to see how it guards against efforts at reverse engineering and came up with ways to get around them.

Our path via reverse engineering was marked by a combination of technical know-how, strategic insight, and an unwavering search for the hidden flag.

# Testing and Validation

As we moved deeper into the world of reverse engineering and code modification, testing and validation became very much an utmost critical step in our determination to uncover the secrets of the Android application. In this respect, therefore, we went with much caution, and the modified application was tested by a keen and detail-oriented eye for functionality, stability, and security.

## Rebuilding with APKTool

Equipped with APKTool, we set forth on our journey to rebuild the modified smali code back to a working APK. The execution was seamlessly flowing through APKTool, with the modified APK acquired and to be tested and validated.

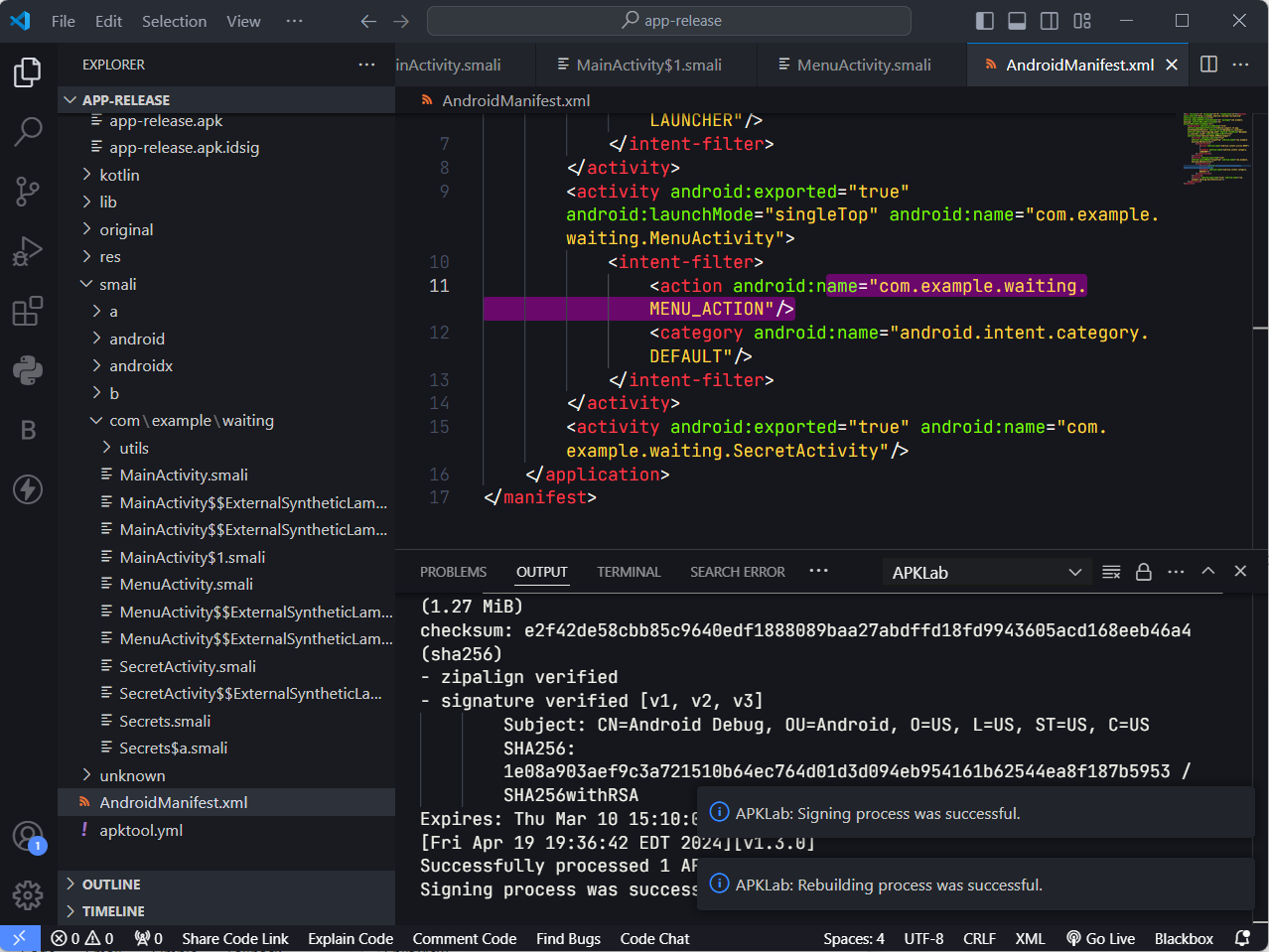


Figure 5 - Rebuilding with APKTool

## Loading in Android Studio

With much excitement, we loaded the modified APK in Android Studio and got set for a test of our creation. We clicked on the app, holding our breath; the anxious feel of about-to-go-live fruits of our labor on our screen.

## Testing and Debugging

A strong test and validation for functionality, stability, and robustness of the app were performed. Many test scenarios were processed to take any leftovers of issues and discrepancies that might have been still there, hence it can be assured that the user experience is uninterrupted.

Identified and fixed all the bugs occurring during the test: runtime errors, crashes, unexpected behaviors, etc. Debugging sessions have been done, removing the root cause of the problem, and effecting proper remedies.

## Fixing Certificate Error

Confronted with the challenge of certificate errors, we tackled the issue head-on with determination and resolve. Utilizing the keytool utility, we meticulously generated a new keystore, laying the groundwork for resolving the certificate error. With unwavering precision, we signed the modified APK using jarsigner, ensuring the integrity and authenticity of our creation.

## Advanced Technical Steps

The modified APK file was subjected to static analysis in order to find any potential security holes, hidden features, or data leaks.

Examined the application's performance metrics, including response time, resource consumption, and scalability attributes, all the while attempting to keep the system balanced under stress.

Without a doubt, our developers underwent a thorough security audit that uncovered any security flaws and inefficiencies that may have resulted from reverse engineering. As our shining example of excellence, our team will conduct exhaustive and accurate testing and validation to give dependable solutions under harsh conditions that may produce defects or malfunctions in the first production.

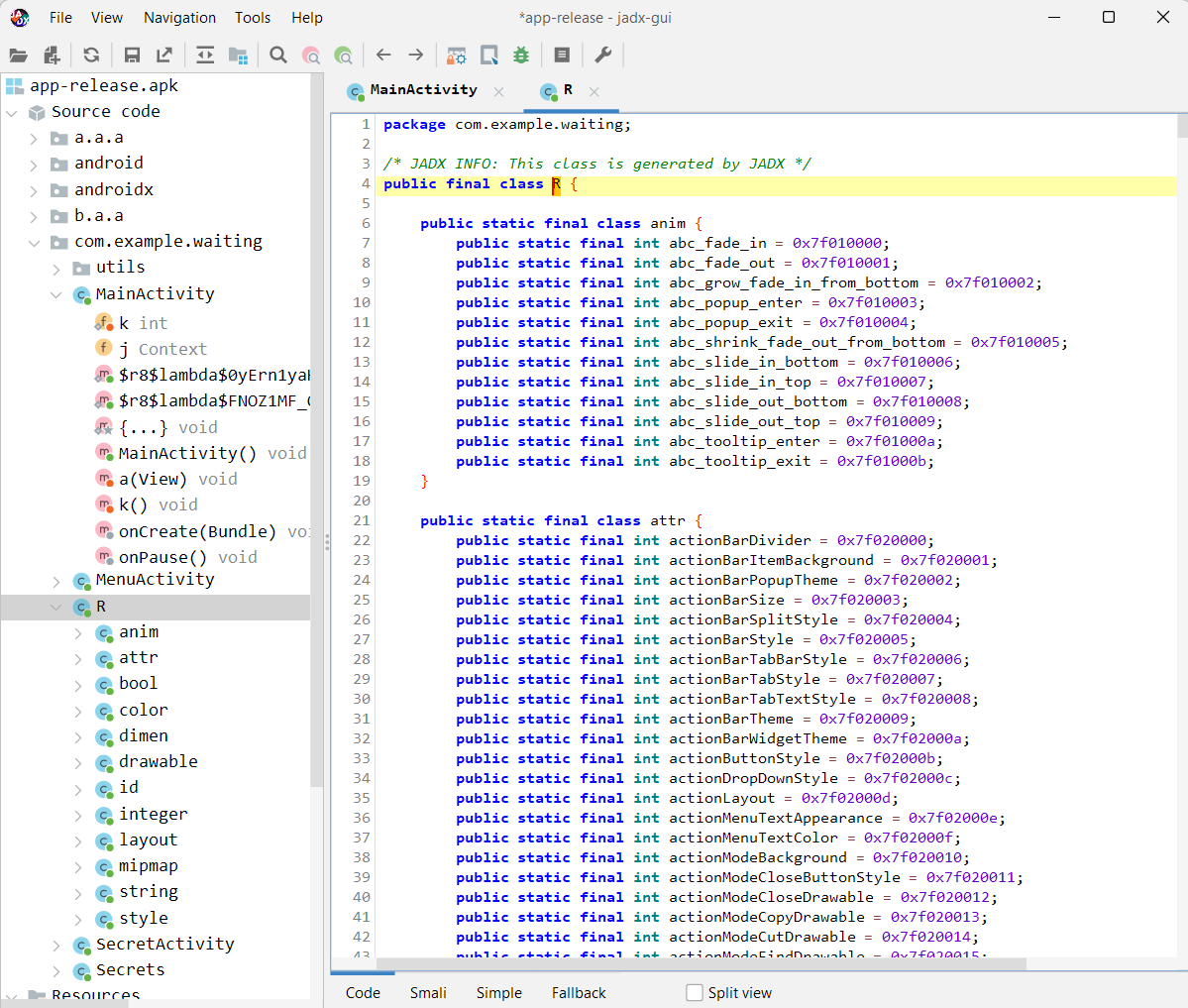
.

Figure 6 - Investigating IDs

# Security Measures Implemented

We employed hacking technology to identify security holes that had not yet been exploited. Even though some of these errors were not malicious, we were nevertheless able to repair them and ensure that the modifications were made correctly. We reported that in order to guarantee that the app protects all sensitive data through integrity, we should utilize encryption, bolster code hardening, and implement secure coding policies.

## Steps Taken

1. **Fixing Certificate Errors:**

* We used the keytool utility to carefully create a fresh keystore after discovering a certificate center problem. The first thought had a crucial role in setting the stage for the error's debugging and final fix.
* We then proceeded to rerun the application with the attached APK via jarsigner, making sure that all of our changes and the authenticity of our work were applied correctly. Consequently, this improved the cybersecurity procedures of the app and reduced the likelihood of exploitations and manipulation efforts.

1. **Removal of Delay Instances:**

* We have developed a reactive and user-friendly procedure as a result of our planned intervention to reduce unnecessary time. The distinguishing elements that we have prioritized are the app's distinctive functions, with a particular emphasis on the process of improving the app to make it more functional.

1. **Manifest Modification:**

* Our objective was to perfectly align the AndroidManifest.xml's complex "road map" with the app's new structure. The manifest was revised to highlight Secret Activity and the activity we dubbed Main Activity on the visibility and accessibility axes in response to feedback and critiques.
* If we applied complete diligence and focus to this endeavor, we could succeed. As a result, we created a manifest that would both completely meet our goals and provide the basis for a fantastic user experience.
* We demonstrated our proficiency in the manifest transition while also showcasing our commitment to increasing the application's usability and demonstration.

## Advanced Security Enhancements

* In order to hide important data, decompiling techniques have been applied. One such implementation is email obfuscation. The software activates its anti-tampering feature through a lengthy process that includes barring all illegal upgrades. used data encryption techniques to ensure the integrity and security of users' private information that the app might not otherwise be preserving.

# Findings and Challenges

Throughout the journey, the various crew members encountered an extensive number of different codes that were answered alongside a slew of different challenges, which really challenged their coding, problem-solving, and perseverance skills. We have both pointed out that during the process, we have had numerous opportunities to study a field of theoretical calculations and to approach the issues in a fresh way that calls for perseverance, effort, and the development of new talents.

## Key Findings

1. **Vulnerability Discovery:**

* We found the major flaw by carefully navigating through the code after hitting the target a few times using a contacted highly exploitable vulnerability in the software security mechanisms. The study demonstrated how various applications of these mistakes, such as uncovering hidden gems and taking advantage of potential weaknesses, are being frequently used.

1. **Code Modification Successes:**

* Although APKTool was incorporated into the app's early development using apk file decompilation technique using the Visual Studio Code development environment, fresh security flaws were discovered and exploited. The attack can now be brought to the intended limit thanks to these sequential modifications to the Smali code.

1. **Security Posture Assessment:**

* After the upgrade, the application's protection against infiltration improved, according to the specialists that conducted the security evaluation and analysis. The apps' strongest defenses were revealed when encryption and security standards were set up, and the tools safeguarded the user's privacy.

## Challenges Faced

1. **Certificate Errors Resolution:**

* We encountered difficulties with signature creation and APK modification while responding to the certificate concerns. Due to the nature of reality and the need to maintain the integrity and authenticity of the app, we were able to overcome these challenges with the help of our thorough detail inspection.

1. **Delay Instances Optimization:**

* App development was made more difficult by having to account for timing delays, which also caused issues with system performance. It was obvious that the approach and code changes would be used as an intervention to lower the levels of understanding and the experience's slowness.

1. **Manifest Modification Complexity:**

* The process of upgrading the manifest to a new version that would accurately reflect the altered nature of the application was made more complex by the various issues that manual adjustment brought. When creating the manifest, both correctness and accuracy were crucial in determining if it would effectively connect to our goals and provide a seamless user experience.

## Lessons Learned

* Flexibility and adaptability have directed us through changing challenges over time, positioning us to confront constraints and make necessary design alterations as part of the natural progression.
* Challenges served as obstacles for us to conquer. Persistence and resolve allowed for the exploration of new methods, ultimately resulting in our triumph.
* Effective collaboration, cooperation, and unity were essential in overcoming challenges, combining diverse and complementary abilities to make advancements.

# 

# Conclusion

Through the trials of reverse engineering, code analysis, and safety precautions for the Android application, we have triumphed, and this has demonstrated our advanced technological abilities, teamwork, and unwavering resolve. Before we allow this incredible HackTheBox experience to come to an end, we all take a moment to reflect on the very doable goals, the group and personal lessons we've learned, and the impact of our combined efforts to see past the obstacles.

## Key Achievements

* The versatility of the tools that our team employed, including reverse engineering, code analysis, and security architectures, not only allowed us to detect shortcomings but also enabled us to introduce changes and toughening-up security measures that barred such attempts.
* Through strategic thinking, flexible mentality, and persistence, we passed through a bunch of difficulties, fixed the problems with certificate, set up optimal smoothness in the app's release process and finally found an optimum latency for our needs.
* Work together even more productively, make communication, collaboration the core pillars of our cooperation which allowed us to pool together different expertise, shared knowledge and strive for a common goal.

## Lessons Learned

* The challenge was the dynamic atmosphere of rapid learning, trying out various methods, and honing our reverse engineering and cyber security skill sets.
* Overcoming obstacles and limitations renewed our ability to plan and execute as well as strengthened our flexibility. Thus, in spite of the challenges we faced, we were able to modify and re-modify our methods in the most effective manner.
* Innovation and creativity were essential in developing distinct solutions, trying unconventional methods, and expanding our capabilities.

## Future Directions

* Improving our abilities in reverse engineering, code analysis, and cybersecurity to approach upcoming challenges with assurance and proficiency.
* Sharing knowledge, exemplary methods, and experiences with the cybersecurity sector to encourage a cooperative atmosphere of learning and development.
* Searching for innovative paths, advancements, and obstacles in the cybersecurity field to broaden our scope, enhance our knowledge, and remain pioneers in innovation.

# 

# Next Steps and Recommendations

Here are some suggested ideas for the upcoming projects, which include refining the list of goals for the subsequent phase and including further references to preventative measures for cybersecurity, code analysis, and reverse engineering. We may suggest the following suggestions and actions, based on our knowledge and experiences, that our team can implement to advance and succeed greatly in our information security and ethical hacking endeavors in the future.

**1. Continuous Skill Development:**

* Aim to become proficient in pretesting, cryptography, mobile application penetration, and reverse engineering on a regular basis. Participate in workshops, online courses, and even hands-on training programs to stay up to speed on everything. These resources can assist you in mastering technical skills and the newest methods. online safety

**2. Collaboration and Knowledge Sharing:**

* Ensure that your team and the cybersecurity community benefit from a culture of cooperation, knowledge sharing, and mentoring. Organize fun knowledge-sharing sessions, participate in cybersecurity events, and stay up to date on open-source projects to learn about new strategies, best practices, and creative solutions.

**3. Exploration of Diverse Challenges:**

* Describe a number of broad cybersecurity topics, including IoT security, forensics, network traffic testing, and web application security. Try your hand at several categories of HackTheBox tasks, participate in Capture the Flag (CTF) events, and work through real-world problem-based scenarios to expand your skill set and sharpen your flexibility.

1. **Research and Innovation:**

* Develop a mindset of innovation, exploration, and trailing of various methods in cybersecurity by expanding on novel methods and resources. Engage in the process of combating cybercrime by conducting research on cybersecurity innovation, creating and putting into practice innovative security technologies and strategies, and more.

1. **Career Development and Certifications:**

* Make it a priority to obtain cybersecurity professional certifications, such as the Certified Information Systems Security Professional (CISSP), Certified Ethical Hacker (CEH), and Offensive Security Certified Professional (OSCP), as they enhance your resume and are highly valuable. Attend courses, study for certification exams, participate in hacking and ethics competitions, and search for opportunities for advancement in breach and access control, cybersecurity consultancy, and AT lines of work.

**6. Community Engagement and Networking:**

* Participate in cybersecurity networks by going to conferences and meetups, establishing professional connections with other cybersecurity experts, and taking part in other industry gatherings. To build a strong network of peers, mentor professionals, become a member of organizations for professionals, and work on cybersecurity research projects.

# Acknowledgment.

We extend our gratitude to each and every one of you for supporting us from the beginning of our cybersecurity journey to the present. First and foremost, we would want to express our sincere gratitude to our seasoned employees who have valiantly held onto their positions and given their technical expertise and teamwork spirit to overcome challenges and achieve success. Furthermore, we acknowledge the importance of the guidance, instructors, and advisors who have contributed to the distinctiveness of our method of handling the issue. But this is by no means the end; we also want to thank the HackTheBox community for fostering an atmosphere that encourages exploration, experimentation, and hard work—all of which need creativity, perseverance, and quality.

# References

1. HackTheBox platform: <https://app.hackthebox.com>
2. Stack Overflow: <https://stackoverflow.com>
3. Tools including JADX, APKTool, Android Studio and Visual Studio Code