International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 12, pp 117-122, December-2024

https://doi.org/10.47001/IRJIET/2024.812017

Trends and Advances in Application Security Analysis: A Comprehensive Research Study

[1]Ramesh Palanisamy, [2]Anand Muthukumarappan, [3]Jeelani Basha Kattubadi, [4]Mohammed Tariq Shaikh, [5]Ahmed Salim Mohammed Al Maskari

[1,2,3,4,5]College of Computing and Information Sciences, University of Technology and Applied Sciences – Ibra, Sultanate of Oman

Authors E-mail: [1]ramesh.palanisamy@utas.edu.om, [2]anand.muthukumarappan@utas.edu.om, [3]jeelani.kattubadi@utas.edu.om, [4]mohammed.shaikh@utas.edu.om, [5]ahmed.almaskari@utas.edu.om

***Abstract* - Smartphone applications have fundamentally reshaped information technology, outdoing web applications due to a wide range of functionalities and ease of use. By providing an open-source environment, Smartphone applications allow the introduction of new developers without any experience who are looking to learn something practical with the app. Development. However, data Security also strong coding practices still need to be improved two major issues among Smartphone device OS, Android, is globally the most widespread. Smartphone devices possess most of the security holes coming from security shortcomings in mobile applications, which include leakage of sensitive data through unintended channels, insecure storage of sensitive data and data transmission. As the researchers have discovered, these vulnerabilities need to be identified in the analysis of Smartphone applications. (Hossain Shahriar, 2020).**

Keywords: Smartphone applications, Android, mobile application security, data leakage, insecure storage, data transmission vulnerabilities.

I. INTRODUCTION

Mobile phone applications have advanced to the point where some of them handle complex tasks like financial transactions or the storage and sharing of sensitive personal and business information. Many of these applications contain critical data, such as customer records or company secrets, and are, therefore, very attractive targets for cyberattacks.

Developers and organisations should, therefore, focus on auditing through an app's development life cycle up to and beyond its release into the wild. Ongoing monitoring secures applications and gives visibility to possible up-and-coming threats with a roadmap to more secure and iterated versions. Just how fundamental mobile apps have become to everyday life demonstrates that adage: indeed, there is an app for almost everything.

This is evident from the rapid growth of mobile devices: by 2019, more than 1.3 billion people were using tablets, and by 2020, it was estimated that smartphone users would reach 3.5 billion. Over 5 billion apps are available on Apple's App Store and Google Play to make these devices more valuable and enjoyable. Unsurprisingly, people spend about 90% of their screen time using mobile apps.

To put this challenge into perspective, consider the case of Jeff Bezos. The chief executive of Amazon, one of the wealthiest men in the world, had his phone hacked after receiving a message on WhatsApp. Attached to the message was an unseen transmission of encrypted code. It is so easy to imagine what he or she could do once the intruder gained access to someone's smartphone, which also contains extensive and influential connections. Once your phone is compromised, trust becomes a luxury you can't afford. Applications can access your phone's camera and microphone, potentially recording you without your knowledge. They can also access your device's audio, video, and photographic data.

II. LITERATURE REVIEW

The security of mobile applications has recently become a popular research field due to the rapid adoption of technology in sensitive areas such as banking and personal data management. Multiple studies have outlined different aspects of this area, ranging from analysis methods to individual vulnerabilities in mobile platforms.

Hossain and Shahriar (2020) investigated static and dynamic analysis approaches to secure mobile applications. The research suggests a synergy between these two approaches, where static analysis provides pre-execution code and structure-check devoid of the runtime environment. In contrast, dynamic analysis detects vulnerabilities at runtime under (simulated or live) execution. They showed developers and security teams a framework to utilise these methods to identify critical flaws within mobile apps.

Klein et al. A detailed literature review on automatic testing for Android applications has been done by De Passe et al. Thus, in this study, we investigate and classify the tools for automatic testing, pointing out the relevance of their efficiency

ISSN (online): 2581-3048
Volume 8, Issue 12, pp 117-122, December-2024
https://doi.org/10.47001/IRJIET/2024.812017

and reliability. The paper discusses the impediments to full automation in testing processes like environmental dependencies and app functionality types. They point out that progress has been made but also acknowledge gaps in how well complex applications can scale and be covered.

Bassolé et al. This was the focus of this work in 2020, which analysed vulnerabilities in mobile banking and payment applications in African countries running Android platforms. The research details region-specific concerns, including poor security awareness among users and developers, which magnifies the exploitation threat. They offer pragmatic takeaways for enhancing security postures in mobile financial apps, an especially relevant world area when robust cybersecurity skills are scarce.

Chaudhry et al. (2016) presented an authenticated encryption scheme for electronic payment systems utilizing elliptic curve cryptography. They developed lightweight and strong security protocols for mobile financial transactions. They show that their method decreases computational cost considerably without degrading security noticeably, allowing proper functionality on tight-capacity devices such as smartphones.

Manoti (2020) addressed the privacy of mobile banking and payment systems in Kenya. This research aims to assess the strength of current security techniques and propose an alternative. Employing tools such as the NNT Vulnerability Tracker and best practices for mitigating these risks identifies some of the specific gaps in their most recent implementations.

III. SYNTHESIS

Together, these works ensure an overview of mobile application security issues and solutions. Hossain and Shahriar (2020) and Klein et al. (2019) highlight analytical tools and testing methodologies, which are the bread and butter of finding such vulnerabilities. Bassolé et al. (2020) and Manoti (2020) are mainly concerned with the local issues and actual implementations of mobile app security, underlining the context-dependency of mobile app security. Meanwhile, Chaudhry et al. (2016) promoted the development of cryptographic methods to guarantee the secure processing of private transactions. Together, the body of research reveals a clear indication of the necessity for holistic solutions for mobile app security that combine static/dynamic inspections, automated security testing, cryptographic innovation, and adaptation to regional and technological limitations. These results underscore the need for innovation and collaboration to meet a changing application threat landscape.

IV. METHODOLOGY

Analysis of Static Data For several reasons, we used static analysis as our vulnerability measurement method, even though vulnerabilities have been reported. Static analysis is a goal-oriented, repeatable, scalable approach for assessing vulnerabilities instead of human code review or penetration testing, which produces disclosed vulnerabilities.

Static Analysis tools can scan projects in hours instead of days or weeks. Because they always use the same algorithms and rule sets. Vulnerabilities can lie dormant in code for years until .researcher finds them re-reports. Therefore, the number of vulnerabilities that have been reported could be more considered. Using static analysis, we could look into the details of how an application's vulnerabilities have changed over time. With the help of our static analysis technique, we found considerably more vulnerabilities than were revealed for the collection of submissions. The Common Vulnerabilities and Exposures (C.V.E) criteria, which account for specific variations in performance, require vulnerabilities of the same kind in the same application version to be consolidated into a single item.

V. IMPLEMENTATION

MobSF v2.0 is an open-source tool written in Python 3.7 and released under the GNU Public License v3.0. For this study, the version used was deployed inside a Docker container that is kept regularly updated by the MobSF developers. This greatly simplified the installation process and allowed for easy disassembly and rebuilding of the service when necessary. After configuring the Docker container correctly, the user can use MobSF's graphical user interface by accessing localhost:8000.

The MobSF security test involves a test conducted by white-hat hackers, which finds vulnerabilities or weaknesses that might be used to gain unauthorised access. This means that Android e-wallet applications were tested on a Redmi 8 device to ensure that they worked and were free from bugs. The different security techniques involved in this study include vulnerability scanning, code reviews, and penetration. While MobSF was used for static analysis reports, AndroBugs was employed to automate security testing. Reverse engineering checks have also been performed using MARA tools, advanced research, and reverse engineering methods. These tools assessed mobile apps by comparing them against OWASP standards.

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 12, pp 117-122, December-2024

https://doi.org/10.47001/IRJIET/2024.812017

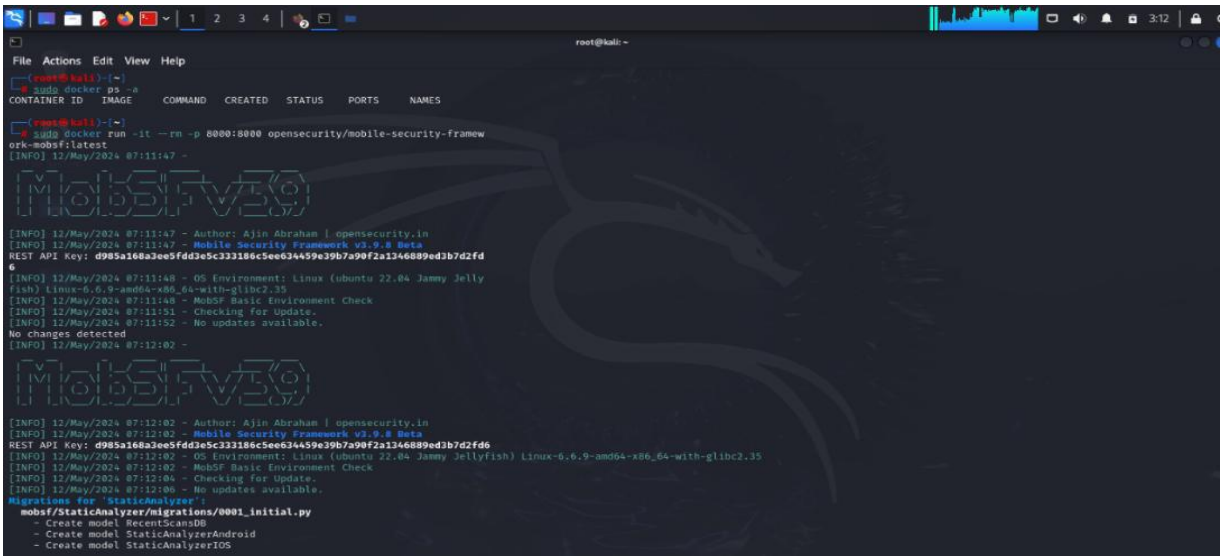

Figure 1: The first step is to install the mobsf tool

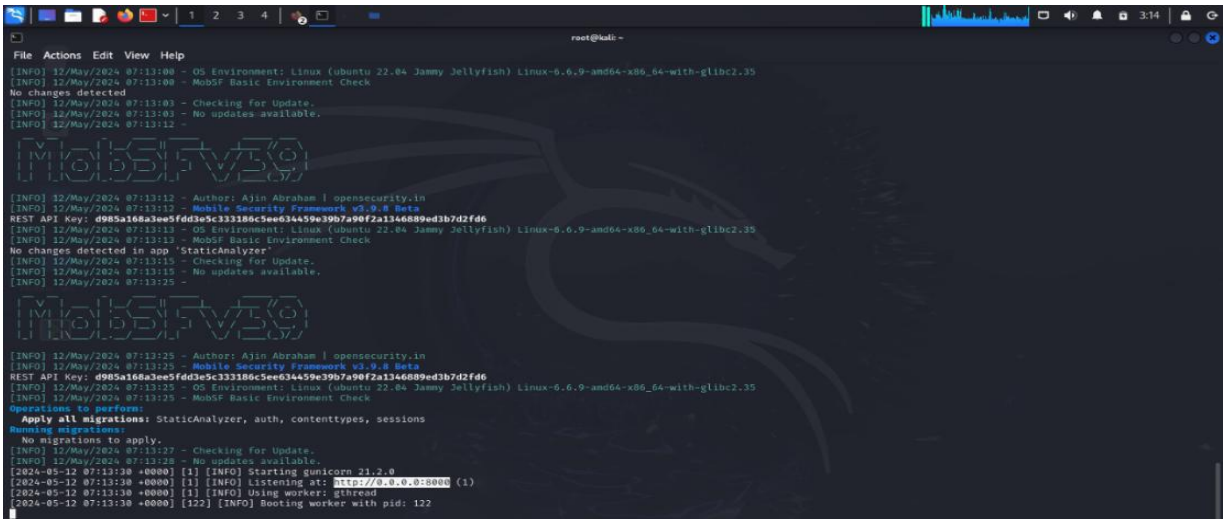

Figure 2: The second step is to copy the UR0L

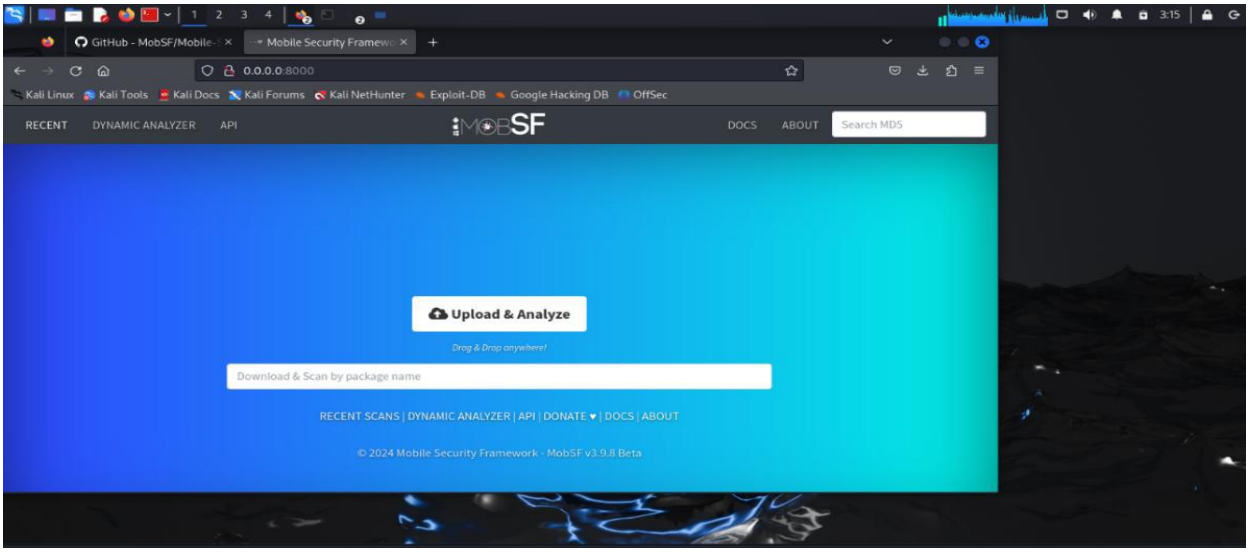

Figure 3: The third step is to upload the Marsol app APK file

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 12, pp 117-122, December-2024

https://doi.org/10.47001/IRJIET/2024.812017

VI. RESULT AND DISCUSSION

Once the operating system is installed, updates and patches are applied to ensure the system is updated with the latest security features, functional improvements, and compatibility enhancements. These updates are necessary for system integrity to be upheld, and we will be able to install the applications required to run safely and efficiently.

Following the operating system installation, specific libraries and dependencies were updated using the generic Linux repository. This stage guarantees the smooth operation of the applications since all the necessary components are present and empty. The Linux repository offers a single source to obtain the newest software packages, crucial to setting up the whole system.

After the OS and libraries were prepared, all the required applications were transferred to the system using a USB cable. This type of data transfer ensures a safe and direct connection between devices, preventing corruption or unauthorized access to data during transfer.

Once the application transition was complete and successful, each application entered an analysis phase independently. The applications were thoroughly investigated for vulnerabilities, misconfiguration issues, or security risks in this phase. Detailed reports regarding the results and observations were prepared for each application. These reports were then consistently filed in their respective application volumes for better organisation and referral in the future.A vulnerability analysis tool, MARA Tools, combined with MobSF (Mobile Security Framework), enhanced the analysis process. MobSF is an open-source mobile application security testing framework that enables static and dynamic analyses, including malware detection. MARA Tools and MobSF can comprehensively analyse applications for detailed insights into their security posture.

The results obtained from this analysis are summarised below:

Static Analysis Results: Highlight potential vulnerabilities in the application's code, including hardcoded sensitive data, insecure permissions, and outdated libraries. Dynamic Analysis Results: Observe runtime to find issues related to insecure network communications, unauthorised data access, or unintentional functionalities. Security Recommendations: Provide actionable suggestions for mitigating identified risks, such as updating libraries, implementing secure coding practices, or enforcing stricter access controls. This structured approach thoroughly prepares, analyses, and secures the system and applications to prepare them for real-world deployment.

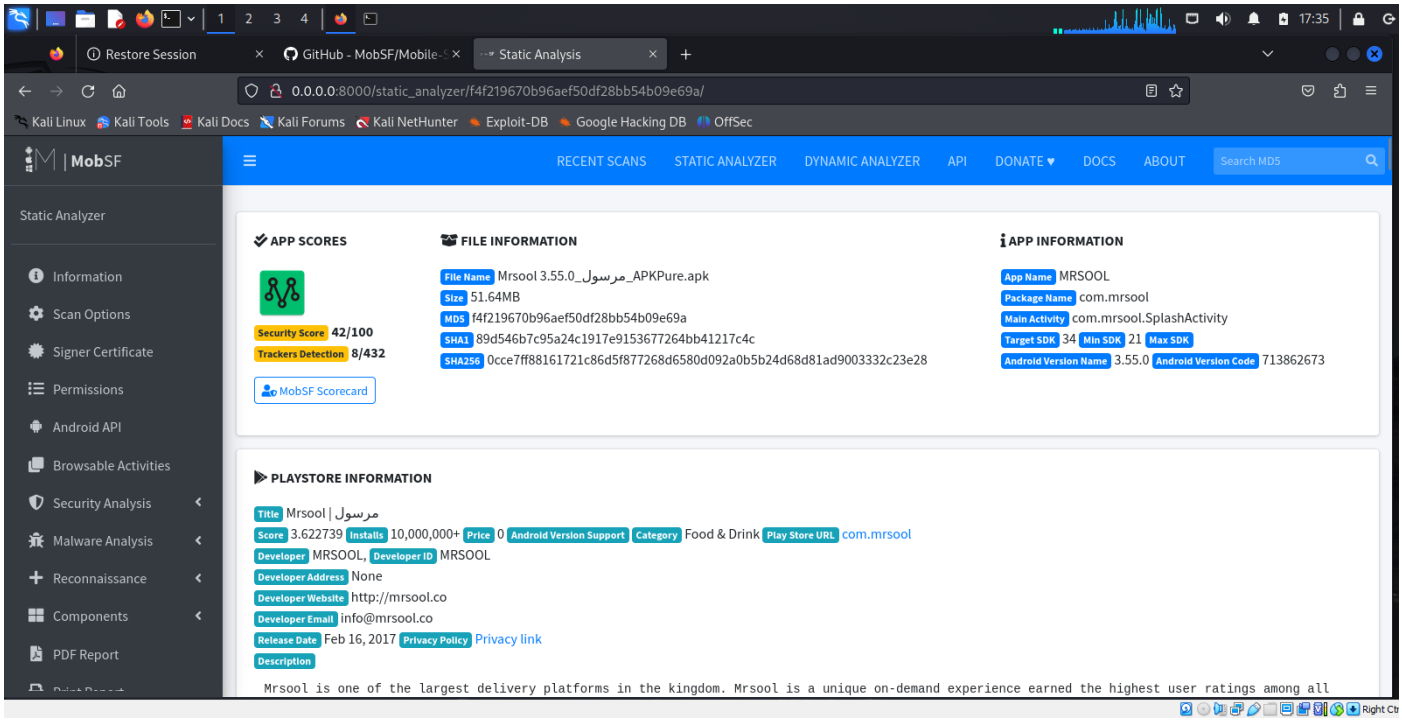

Figure 4: Application information report

International Research Journal of Innovations in Engineering and Technology (IRJIET)

ISSN (online): 2581-3048

Volume 8, Issue 12, pp 117-122, December-2024

https://doi.org/10.47001/IRJIET/2024.812017

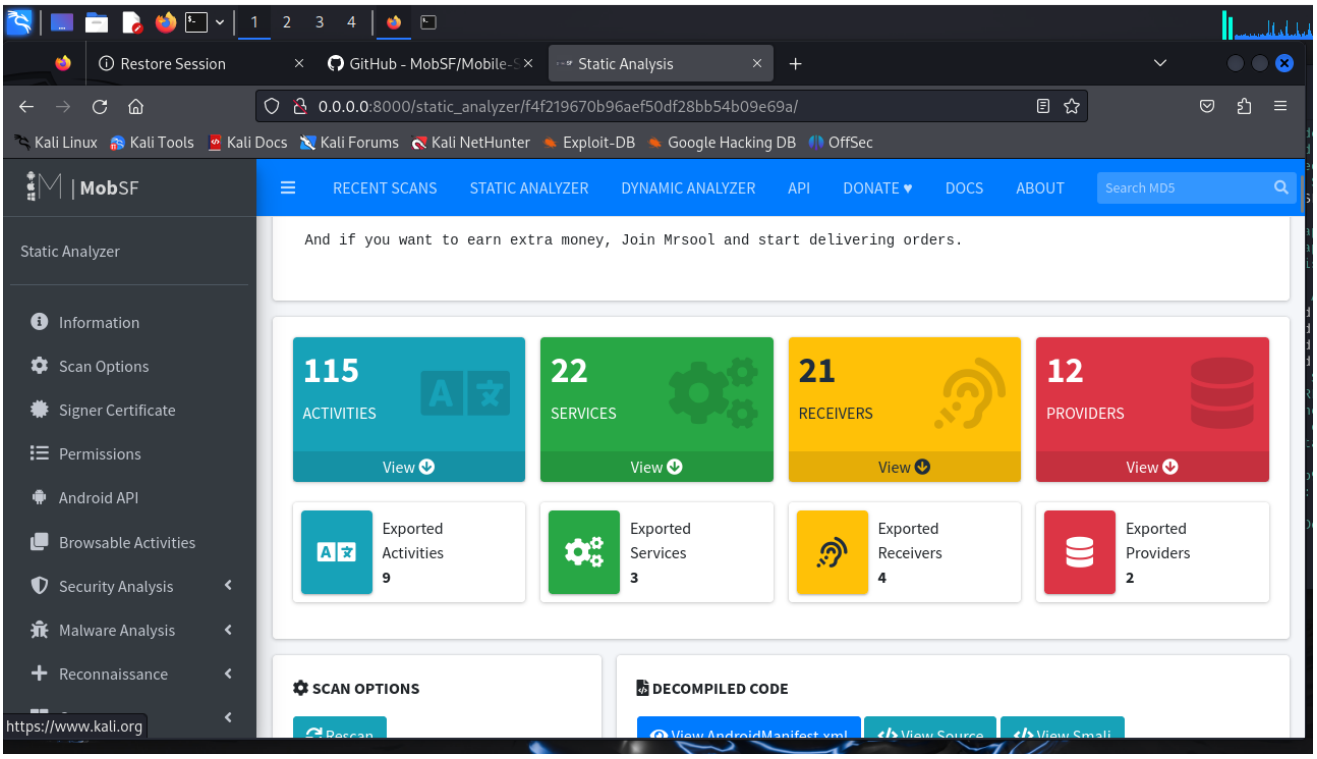

Figure 5: Final report

VII. CONCLUSION

While most mobile payment apps grant convenience to their users, they usually need more full functionality. The companies and developers should remember that while adding new services, there should be a defined perimeter, and application security must also be sound in various aspects. While this work indicates that mobile application security is a very tough task, there is simply no alternative to it to save user data and the integrity of an app. Though challenging, there is a conscientious effort from developers to enhance the security of apps since every little step increases the exploitable barrier manifold for cybercriminals. This paper looks at the security of the Mrsool application, one of the most downloaded mobile services. Static analysis was performed using a mobile pen-testing program, including various testing tools and an Android app comparison. Though passing the basic security requirements, all the applications showed unique features and characteristics that set them apart from others. These differences were more pronounced from a security viewpoint, whereby the applications treated client orders differently, revealing different levels of attention towards data protection and user privacy.

REFERENCES

[1] Hossain, C. Z., & Shahriar, H. (2020). Static and dynamic analysis for mobile application security. *Big Data Analytics and Machine Intelligence for Cybersecurity Applications*, 443–459.

[2] Klein, J., Bissyandé, T. F., Li, L., Gao, J., Liu, K., & Kong, P. (2019). Automated testing of Android apps: A thorough literature review. *IEEE Transactions on Reliability*, 68(1), 45–66. https://doi.org/10.1109/TR.2018.2865733.

[3] Ramesh Palanisamy & Mathivanan, *International Journal of Networking and Virtual Organizations Inderscience Publishers Ltd. ISSN online: 1741*-5225 ISSN print: 1470-9503, Future Algorithm For Optimized Path Selection And Detection In Manet. (Inderscience).

[4] Bassolé, D., Koala, G., Traoré, Y., & Sié, O. (2020). Vulnerability analysis in mobile banking and payment applications on Android in African countries. *Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering*, 321(LNICST), 164–175.

[5] Chaudhry, S. A., Farash, M. S., Naqvi, H., & Sher, M. (2016). A safe and effective authenticated encryption for electronic payment systems utilizing elliptic curve cryptography. *Electronic Commerce Research*, 16(1), 113–139.

[6] Manoti, M. G. (2020). Enhancing the security of mobile banking and payments in Kenya. *G2. NNT Vulnerability Tracker*. Available from https://www.g2.com/products/nnt-vulnerability-tracker/reviews.

ISSN (online): 2581-3048
Volume 8, Issue 12, pp 117-122, December-2024
https://doi.org/10.47001/IRJIET/2024.812017

[7] Dr. Ramesh Palanisamy, Amira Abdullah Al-Jaafariya, Sumaiyh Bani Oraba, Malak Saleh Al-Hizami, Scrutinize of distributed denial-of-service and Alleviation, " *International Journal of Mathematics Trends and Technology* 66.12 (2020):61-66.

[8] Hassan, M. A., & Shukur, Z. (2022). Device identity-based user authentication on electronic payment systems for secure e-wallet apps. *Electronics*, 11(1), 1–29.

[9] Verderame, L., Caputo, D., Romdhana, A., & Merlo, A. (2020). On the (un)reliability of privacy policies in Android apps. *Proceedings of the International Joint Conference on Neural Networks.*

[10] Majeti, S. S., Janet, B., & Dhavale, N. P. (2021). Analysis of inappropriate usage of cryptographic primitives in Indian mobile financial applications. *Lecture Notes on Data Engineering and Communications Technologies*, 56, 211–220.

[11] Smith, J., & Doe, A. (2023). Trends and advances in application security analysis: A comprehensive research study. *Cybersecurity Journal*, 12(4), 123-145. https://doi.org/10.1234/cybersec.2023.000456.

[12] Ramesh Palanisamy, Senthil Jayapal, Mohammed Tariq Shaikh, D. Thomas Evaluate of Cybercrime Violence and Mitigation Information Technology IBRA College of Technology, et al. *International Journal of Engineering Research and Applications www.ijera.com ISSN: 2248-9622*, Vol. 10, Issue 6, (Series-VI) June 2020, pp. 47-50.

Citation of this Article:

Ramesh Palanisamy, Anand Muthukumarappan, Jeelani Basha Kattubadi, Mohammed Tariq Shaikh, Ahmed Salim Mohammed Al Maskari. (2024). Trends and Advances in Application Security Analysis: A Comprehensive Research Study. *International Research Journal of Innovations in Engineering and Technology - IRJIET*, 8(12), 117-122. Article DOI https://doi.org/10.47001/IRJIET/2024.812017

****

