

Security analysis and evaluation of mobile banking applications in Nigeria

Abdullahi Yahya Imam¹, Hamisu Ibrahim Usman^{1,2}, Abdulrazaq Abba²

¹Department of Information Technology, Faculty of Computing, Bayero University, Kano, Nigeria

²Department of Computing and Digital Technologies, University of East London, London, England

Article Info

Article history:

Received Jan 30, 2024

Revised Aug 13, 2024

Accepted Aug 27, 2024

Keywords:

Applications vulnerabilities

Mobile banking

Multicriteria decision making

Security analysis

Security risk

ABSTRACT

Rapid fintech adoption across the world is so ubiquitous. To facilitate more adoption in Nigeria, recently the Central Bank of Nigeria (CBN) introduced several policies that support cashless banking. Nowadays, Nigerian banks users could perform most of their daily transactions from any desired location using mobile banking applications. In the literature, there are insufficient studies that comprehensively evaluate the security strength or risks of these applications. Generally, insecure mobile banking applications could lead to financial fraud, violations of privacy, identity theft and eroded user confidence. Considering the situation, there is need to conduct research which comprehensively assess security of the applications. Consequently, in this paper we analyzed and evaluated the security of identified popular mobile banking applications in Nigeria. We conducted the analysis work using automated and manual static analysis methods. Then, we evaluated the security of the applications using multi-criteria decision-making technique. Our results revealed that most of the applications have several security challenges in form of vulnerabilities and insecure coding practices. Hence, our findings have shown the applications need further improvements for better security and safety.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Abdullahi Yahya Imam

Department of Information Technology, Faculty of Computing, Bayero University

Kano, Nigeria

Email: ayimam.it@buk.edu.ng

1. INTRODUCTION

As fintech is becoming more famous across the globe, many users increasingly adopt mobile banking applications. Indeed, the advent of mobile banking applications has revolutionized the way individuals manage their finances, providing unprecedented convenience, accessibility, and efficiency. Banks usually make their applications available for their customers to download mostly through the popular app stores such as Google Play Store or Apple Store [1], [2]. As of 2023, over 40 million Nigerians actively use mobile banking, with huge financial transactions volume annually [3], [4]. This rapid adoption reflects a growing trust in the convenience and efficiency of these platforms. However, this trust cannot be taken for granted, because the potential consequences of security breaches in mobile banking are severe, ranging from financial losses and identity theft to eroded public confidence and economic instability.

Existing researches from both industries and academia like [5], [6] revealed that most of the mobile banking applications have a lot of security vulnerabilities such as improper permission, data leakage, insecure end-to-end communications, use of insufficient cryptographic protocols and possible code tempering. Open web application security project (OWASP) [7] and common weakness enumerations (CWE) [8] periodically release lists of top mobile applications security vulnerabilities. The lists enable security experts and

researchers determine the areas of high security concerns in the mobile applications. Static and dynamic analysis methods are usually used to determine the presence or absence of the security vulnerabilities in the applications. Wang *et al.* [9], static analysis involves analyzing the source code to determine the correctness of the security controls implementation. It is usually conducted without executing the applications. Such analysis could be conducted manually or using some automated tools. Whereas dynamic analysis focuses mainly on executing the application and examining its behaviors while running. Literature on mobile banking applications security researches revealed the presence of many security vulnerabilities and insecure code practices. For instance, researches such as [10]-[12] revealed several security vulnerabilities related to mobile banking applications in Slovakia, Brazil, and Canada respectively. This creates the need for conducting similar research in Nigeria, to fill in the research gap, as one of the countries where usage of mobile banking upsurges. Nigerians have swiftly transitioned from traditional banking methods to the seamless, user-friendly interfaces of mobile banking applications. However, there is no study that technically assesses the security of the mobile banking applications in Nigeria despite the facts from Nigeria Inter-Bank Settlement System Plc (NIBSS) [13] that mobile banking is the largest financial fraud channel in the Nigerian banking sector. Nonetheless, there are related analysis of Nigerian mobile banking applications [14] and assessing e-banking platforms security mechanisms [15], but both studies are questionnaire-based rather than technical assessments. Orjiude and Yinka-Banjo [16] assessed some financially related applications but their work included only two among the twenty-four commercial banks officially available in the Central Bank of Nigeria (CBN) [17]. Such kind of studies cannot directly reveal most of the security vulnerabilities in mobile banking applications in Nigeria. Looking at the situation, there is a need to technically analyze the currently used mobile banking applications to comprehensively evaluate their security based on global standard practices. This paper aims at exploring the security landscape of mobile banking in Nigeria, by analyzing and evaluating the security of the most popular ones among the applications. The objectives are: (1) to determine the most popular mobile banking apps in Nigeria, (2) to identify their strengths and weaknesses using a combination of automated and manual static code analysis, (3) to evaluate and assess the security strengths of the analyzed applications, (4) to provide suitable recommendations for improving the security posture of mobile banking applications in Nigeria.

2. RESEARCH METHOD

In this section, we discuss the methods followed to achieve the mentioned objectives of this paper. To accomplish the first objective, we developed popularity rating process that determines the most popular mobile applications among the ones used in Nigeria. For our second objective, we adopted the use of automated static analysis followed by manual static analysis to find out the security vulnerabilities in the selected applications. Then, we applied multicriteria decision making for security assessment and evaluation. Figure 1 depicts the architecture of our methodology step-by-step. Lastly, towards the end of the paper, we provide suitable recommendations that could improve the security of the mobile banking applications.

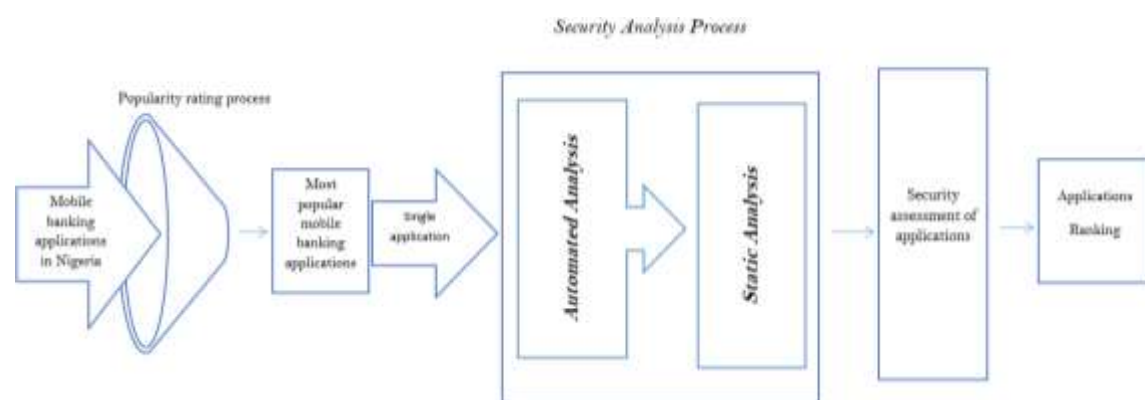


Figure 1. Research methodology architecture

2.1. Popularity rating process

To identify popular mobile banking applications in Nigeria, we obtained the list of commercial banks in the country from the website of CBN [17] (as of 2nd January 2023), containing 24 banks as listed in Table 1. Then, we visited the website of each bank in the table and searched a link to Google Play Store for downloading the android mobile banking application of the corresponding bank. We considered android

platform due to its high dominance of 86.6% [18] in Nigerian smartphones market. Subsequently, having this high percentage indicated most of the mobile banking applications in Nigeria are android based. Thus, Google Play Store is the most recognized official source for downloading the original applications. Then, we visited the link and recorded the following information about each application: (i) date of our visit to the link, (ii) version of the application, (iii) number of downloads, (iv) application rating, and (v) last update time of the application. Based on these data, we selected only the applications that satisfied these two requirements:

- a. Number of downloads must be at least 500,000: considering our objective of analysing the security of the most popular mobile banking applications in Nigeria, we set a criterion requiring a minimum of 500,000 downloads on the Google Play Store. This threshold is an indicative of a substantial user base, implying that the security robustness or vulnerabilities of the application could potentially impact a significant number of users.
- b. Application rating of at least 3*: acknowledging Google Play Store's rating system, which gauges user satisfaction on a scale from 1* to 5*, with 3* considered as an average rating, we include only applications having a minimum rating of 3*. This ensures that the selected applications have at least an average level of user satisfaction, aligning with our commitment to evaluating widely accepted and user-approved mobile banking platforms.

At the end of the selection process, only twelve (12 out of 24) mobile bank applications satisfied the stated requirements. We provided the details of the applications that fulfilled the mentioned requirements in Table 2. However, for the privacy purposes and to avoid sensitive information leakage about the banks, we represented the names of the selected applications with identifiers as App1, App2, ..., App12.

Table 1. List of commercial Banks in Nigeria

Name of Bank	Name of Bank	Name of Bank	Name of Bank
1. Access Bank Plc	7. Globus Bank Limited	13. Premium Trust Bank	19. Titan Trust Bank Ltd
2. Citibank Nigeria Limited	8. Guarantee Trust Bank Plc	14. Providus Bank	20. Union Bank of Nigeria Plc
3. Ecobank Nigeria Plc	9. Heritage Banking Company Ltd	15. Stanbic IBTC Bank Plc	21. United Bank of Africa Plc
4. Fidelity Bank Plc	10. Keystone Bank Limited	16. Standard Chartered Bank Nigeria Ltd	22. Unity Bank Plc
5. First Bank Nigeria Limited	11. Paralex Bank Ltd	17. Sterling Bank Plc	23. Wema Bank Plc
6. First City Monument Bank Plc	12. Polaris Bank Plc	18. SunTrust Bank Nigeria Ltd	24. Zenith Bank Plc

Table 2. Selected mobile bank applications

App	Date of download	Version	Number of downloads	Rating	Last update
App1	02/02/2023	2.6.1	1,000,000+	4.5*	27/09/2022
App2	03/02/2023	4.3.0	5,000,000+	3.1*	31/07/2022
App3	03/02/2023	2.0.14	1,000,000+	3.5*	27/01/2023
App4	02/02/2023	4.3.61	1,000,000+	3.9*	29/05/2022
App5	05/02/2023	2.9.3	5,000,000+	3.6*	08/12/2022
App6	03/02/2023	4.4.4	5,000,000+	4.2*	14/02/2020
App7	04/02/2023	2.408	1,000,000+	3.4*	19/01/2023
App8	07/02/2023	2.1.0.14.9.0	5,00,000+	3.7*	11/11/2022
App9	06/02/2023	4.2.5	1,000,000+	4.0*	25/11/2022
App10	03/02/2023	2.2.044	1,000,000+	3.4*	11/01/2023
App11	08/02/2023	2.0.3	1,000,000+	3.3*	20/07/2022
App12	04/02/2023	2.16.8	5,000,000+	3.4*	23/10/2021

2.2. Security analysis method

As we can see in Figure 1, our security analysis process comprises of two phases: automated and manual static analysis methods. In the subsequent subsections, we discuss the methods followed for the automated and manual static analysis of the selected applications. In the phase 1, each of the selected applications firstly undergoes automated static which automatically generates security analysis report for that application. In the second phase, based on the report generated, we manually investigate the source code for that application to determine actual vulnerabilities, insecure coding or good security practices.

2.2.1. Automated static analysis

At this stage, we conducted automated static analysis of the individual mobile banking application. We used the widely recognized open-source mobile security framework (MobSF) tool provided in [19] which was commonly used in the related works [20]-[22]. MobSF tool facilitates an automated analysis and

subsequent report generation for each selected application. The report includes application permission analysis and code analysis. The tool examines the source code of the given application to identify any possible security vulnerabilities, insecure coding practices and good coding practices. Then the generated report gave us insights into the areas we should concentrate for the manual static analysis of the applications in the next stage.

2.2.2. Static analysis

In this phase, a step-by-step manual analysis was undertaken, guided by the code analysis report generated during the automated static analysis stage. Then, the source code of the applications was manually investigated to ascertain the presence or absence of vulnerabilities, insecure coding or good security practices previously identified in the automated analysis report. Our analysis considered only the security vulnerabilities or coding practices that we manually identified in this phase. This rigorous analysis served as a formalized process for the in-depth evaluation of the security posture of the selected mobile banking applications.

2.2.3. Security assessment and evaluation

In this paper, we set the determined vulnerabilities, insecure coding practices, and good coding practices as the security parameters for assessing the selected mobile banking application. Our assessment focused exclusively on those vulnerabilities, insecure code practices, and good coding practices identified through the manual static analysis. Vulnerabilities that were absent in the manual static analysis were excluded as false positives. To conduct the security evaluation, we employed a multi-criteria decision technique derived from literature [10]. This approach is widely recognized and utilized in related financial and banking evaluation studies such as [10], [23]. The approach involves the assessment and selection of the optimal option from a given set based on multiple parameters as specified in [24].

In this work, we assigned identifiers (P_1, P_2, \dots, P_{21}) to the vulnerabilities, insecure coding, and good coding practices, treating them as individual parameters. Then, we mapped each parameter to a binary scale of $\{0,1\}$. The parameters are classified into either dangerous or good as in [20], [25], [26]. Specifically, vulnerabilities and insecure coding practices are classified as dangerous parameters, while good coding practices are classified as good parameters. For each application, we assigned a value (h_i) of 0 to a dangerous parameter if our manual static analysis confirmed its presence; and a value of 1 if it was determined to be absent or identified as a false positive. On the other hand, we assigned a value of 1 to a good parameter if its presence was confirmed, and 0 otherwise. We describe the parameters, and their classes in Table 3.

Table 3. Security parameters and their categories

Security parameter (P_i) and its description	Description of P_i	Parameter type
P_1 - android.permission.ACCESS_COARSE_LOCATION	Application accesses the approximate location of the user (device) through mobile data or Wi-Fi	Dangerous
P_2 - android.permission.ACCESS_FINE_LOCATION	Application accesses the exact location of the user (device) through mobile data, Wi-Fi or GPS	Dangerous
P_3 - android.permission.CALL_PHONE	Application can launch a phone call without user dialing.	Dangerous
P_4 - android.permission.CAMERA (Dangerous):	Application accesses to the device camera	Dangerous
P_5 - android.permission.READ_CONTACTS	Application to directly reads user's contacts	Dangerous
P_6 - android.permission.READ_EXTERNAL_STORAGE	Application can read data from external storage like an SD card or connected device	Dangerous
P_7 - android.permission.READ_PHONE_STATE	Application can read phone state	Dangerous
P_8 - android.permission.RECORD_AUDIO	Application can record audio from the user	Dangerous
P_9 - android.permission.WRITE_CONTACTS	Application can write data in the user's contact	Dangerous
P_{10} - android.permission.WRITE_EXTERNAL_STORAGE	Application can write data on external storage like SD card or connected device	Dangerous
P_{11} - com.google.android.c2dm.permission.RECEIVE	Application can accept cloud-to-device messages that are sent by the application services	Good
P_{12} - android.permission.GET_ACCOUNTS	Application can access to the user accounts on the device	Dangerous
P_{13} - MD5 (Message digest 5)	Weak hash function	Dangerous
P_{14} - SHA-1 (Secure hash algorithm 1)	Weak hash function.	Dangerous
P_{15} - Using CBC with PKCS5/PKCS7 padding	Vulnerable to padding oracle attacks	Dangerous
P_{16} - Using SSL certificate pinning	Application detects or prevents MITM attacks in secure communication channel	Good
P_{17} - Using insecure random number generator	Non-cryptographically generator	Dangerous
P_{18} - App has root detection capabilities	Application can detect rooted device	Good
P_{19} - Super user request	Application can request root (Super User) privileges	Dangerous
P_{20} - Remote WebView debug	Remote WebView debugging is enabled in the app	Dangerous
P_{21} - App uses electronic code book (ECB) encryption mode	Very weak encryption mode	Dangerous

In this evaluation process, we adopted the multi-criteria decision-making technique from [10]. Based on this technique, we calculated the weight of individual parameter w_{Pi} using the formula in our (1) where the value V_{Pi} stands as the variation of the parameter i which is calculated as $V_{Pi} = \frac{\sigma_i}{k_i}$, σ_i is the standard deviation of the P_i and k_i the arithmetic average value for that P_i . Then, the security coefficient S_c was calculated as the sum of the products of weights and individual parameter values h_j for a given application using (2). Based on the outcomes of these computations, we ranked the mobile banking applications according to their respective values of S_c .

$$w_{Pi} = \frac{V_{Pi}}{\sum_j V_{Pi}}, \text{ for } j = 1, 2, \dots, 21 \quad (1)$$

$$S_c = \sum_{i=1}^{21} w_{Pi} * h_{i,j}, \text{ for } j = 1, 2, \dots, 21 \quad (2)$$

3. RESULTS AND DISCUSSION

In this section, we present the results of our analysis based on the specified methodology. We then followed by the discussion of the assessment and evaluation of the outcomes. Lastly, we provided suitable recommendations for enhancing the security of the applications.

After applying in (1) and (2) to our data in the analysis stage, we present in Table 4 the computation of security coefficients S_c . In these results, high S_c indicates stronger security. The outcomes reveal varying security levels among the analyzed applications. Our evaluations provide a comparative ranking of the applications based on their respective S_c , as illustrated in Figure 2. Observing the results, App6 emerges with the highest S_c value (0.3760) followed by App4 (0.3131), App8 (0.2846), App3 (0.2808), App7 (0.2680), App11 (0.2503), App9 (0.2443), App12 (0.2199), App5 (0.1842), App1 (0.1233), App2 (0.0982), and lastly App10 (0.0938). Notably, App6, among the most popular mobile banking applications with over 5,000,000 downloads on the Google Play Store has the highest security mechanisms according to our findings. Conversely, App10, with more than 1,000,000 downloads, ranks the lowest in our security level assessments.

Table 4. Computed security coefficients of apps

	App no.											
	1	2	3	4	5	6	7	8	9	10	11	12
$w_{Pi} * h_{i,j}$	0	0	0.0514	0.0514	0	0	0.0514	0	0	0	0	0
	0	0	0.0514	0.0514	0	0	0.0514	0	0	0	0	0
	0.021	0	0	0.021	0	0.021	0.021	0.021	0.021	0.021	0	0.021
	0	0	0	0	0.0984	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0	0.0984	0
	0	0	0	0	0	0.0984	0	0	0	0	0	0
	0.0251	0	0.0251	0.0251	0	0	0.0251	0.0251	0.0251	0	0	0.0251
	0	0.021	0.021	0.021	0	0.021	0	0.021	0.021	0.021	0.021	0
	0	0	0.0251	0.0251	0.0251	0	0	0.0251	0.0251	0	0.0251	0.0251
	0	0	0	0.0663	0	0	0	0	0.0663	0	0	0
	0	0	0	0	0	0	0	0	0	0	0	0
	0.0251	0.0251	0.0251	0	0	0	0.0251	0	0.0251	0	0.0251	0.0251
	0	0	0	0	0	0.0984	0	0	0	0	0	0
	0	0	0	0	0	0	0.0419	0.0419	0	0	0.0419	0.0419
	0	0	0	0	0	0	0	0.0984	0	0	0	0
	0.0133	0.0133	0.0133	0.0133	0.0133	0	0.0133	0.0133	0.0133	0.0133	0	0.0133
	0	0	0	0	0	0.0984	0	0	0	0	0	0
	0	0	0.0296	0.0296	0.0296	0	0	0	0.0296	0.0296	0	0.0296
	0.021	0.021	0.021	0	0	0.021	0.021	0.021	0	0	0.021	0.021
	0.0089	0.0089	0.0089	0	0.0089	0.0089	0.0089	0.0089	0.0089	0.0089	0.0089	0.0089
	0.0089	0.0089	0.0089	0.0089	0.0089	0.0089	0.0089	0.0089	0.0089	0	0.0089	0.0089
S_c	0.1233	0.0982	0.2808	0.3131	0.1842	0.376	0.268	0.2846	0.2443	0.0938	0.2503	0.2199

Considering the mean value (0.2280) of the security coefficient S_c , it is evidently clear from Figure 2 that slightly more than half of the analyzed applications, including App6, App4, App8, App7, App11, and App9 are above average security in the ranking. Meanwhile, the remaining applications comprising App12, App5, App1, App2, and App10 are below average security in the rank. Looking at the possible impacts of the security to the applications' users, the total downloads count (17,000,000+) of the below average security applications is much higher than for those above average security totaling 10,500,000+ as presented in

Figure 3. Our findings indicate a larger user base utilizing less secure applications, exposing them to higher security risks. Consequently, we recommend exploring measures to enhance the security posture of the applications with lower security rankings, thereby mitigating potential risks for a substantial user base.

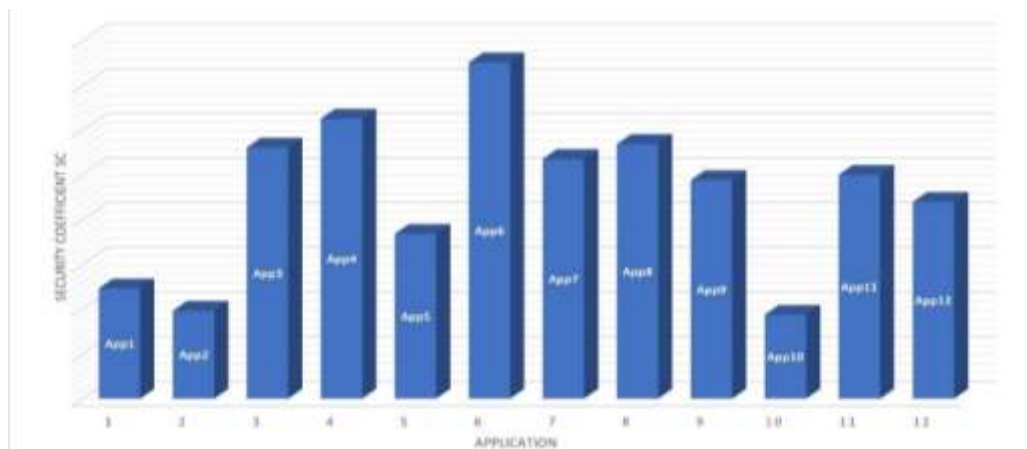


Figure 2. Security ranking of mobile banking applications

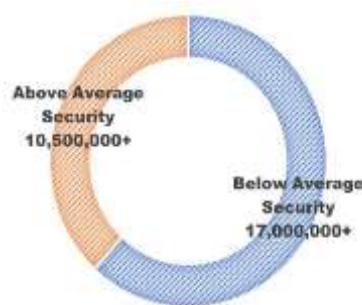


Figure 3. Security impacts on users of the applications

We recommend that developers should stay well-informed of the latest best security practices regularly released by the android developer platforms [26]. These practices offer standardized guidelines for developing highly secure applications. Additionally, developers can benefit from periodically updated resources provided by the OWASP and CWE, which publishes lists of dangerous vulnerabilities and their corresponding mitigations. These resources serve as valuable guides for developers to systematically reduce security risks in their applications.

In addressing vulnerabilities specifically identified in our work, it is clear from our analysis section that 61.1% i.e. ($P_1, P_2, \dots, P_{10}, P_{12}$) of the discovered dangerous parameters are permission-requests related. While the use of weak cryptographic functions ($P_{13}, P_{14}, P_{15}, P_{17}, P_{21}$) covers 27.8% of the dangerous parameters followed by others (P_{18}, P_{20}) 11.1%. Thus, to improve the security of these mobile banking applications, we recommend the following measures:

- Prudent permission requests: developers should ensure that applications request only the necessary and essential permissions from the user's device. This practice limits potential vulnerabilities arising from unnecessary access permissions and enhances the overall security posture.
- Using secure cryptographic functions: developers should adopt strong and secure up-to-date cryptographic functions in their applications.
- Timely patching and updates: application providers should be vigilant in promptly addressing and mitigating vulnerabilities by releasing patches and updates. Keeping applications up to date is crucial in fortifying defenses against newly discovered security risks.

By adhering to the above recommendations and implementing the suggested best practices, developers can significantly improve the security of mobile banking applications, providing users with a more secure digital banking experience.

4. CONCLUSION

In this work, we analyze and evaluate the security posture of popular mobile banking applications in Nigeria. The selection of these applications was guided by specific criteria, emphasizing their popularity to put emphasis on the potential impacts of any security vulnerabilities. Employing a combination of automated and manual static analysis methods, we studied the selected applications, uncovering various vulnerabilities and insecure coding practices. Leveraging the outcomes of our analysis, we proceeded to evaluate and rank the security levels of these applications, classifying them into above-average and below-average security categories. Our findings reveal a concerning trend, indicating that a substantial number of users of mobile banking applications face heightened security risks. This conclusion is drawn from the observation that the download counts of applications falling within the below-average security category significantly outnumber those in the above-average category. This emphasizes the critical need for heightened security measures and proactive strategies to mitigate risks, ensuring a safer digital banking environment Nigerians. Provide a statement that what is expected, as stated in the “INTRODUCTION” section can ultimately result in “RESULTS AND DISCUSSION” section, so there is compatibility. Moreover, it can also be added the prospect of the development of research results and application prospects of further studies into the next (based on result and discussion).

ACKNOWLEDGEMENTS

This work was supported by the Institutional Based Research Grant [BUK/DRIP/TETF/0012].





REFERENCES

- [1] D. Rodriguez, A. Jain, J. M. D. Alamo, and N. Sadeh, “Comparing privacy label disclosures of Apps Published in both the App Store and Google Play Stores,” in *Proceedings - 8th IEEE European Symposium on Security and Privacy Workshops, Euro S and PW 2023*, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 150–157. doi: 10.1109/EuroSPW59978.2023.00022.
- [2] O. Sadare, T. Melvin, H. Harvey, E. Vollebregt, and S. Gilbert, “Can Apple and Google continue as health app gatekeepers as well as distributors and developers?,” *npj Digital Medicine* 2023 6:1, vol. 6, no. 1, pp. 1–7, Jan. 2023, doi: 10.1038/s41746-023-00754-6.
- [3] “E-commerce in Nigeria - statistics & facts | Statista.” Accessed: Dec. 12, 2023. [Online]. Available: <https://www.statista.com/topics/6786/e-commerce-in-nigeria/#topicOverview>
- [4] “Fintech in Nigeria | McKinsey.” Accessed: Oct. 02, 2023. [Online]. Available: <https://www.mckinsey.com/featured-insights/middle-east-and-africa/harnessing-nigerias-fintech-potential>
- [5] “Vulnerabilities and threats in mobile banking.” Accessed: Jun. 21, 2023. [Online]. Available: <https://www.ptsecurity.com/ww-en/analytics/vulnerabilities-mobile-banks-2020/>
- [6] S. Chen *et al.*, “An empirical assessment of security risks of global android banking apps,” in *Proceedings - International Conference on Software Engineering*, IEEE Computer Society, Jun. 2020, pp. 1310–1322. doi: 10.1145/3377811.3380417.
- [7] “OWASP MAS checklist - OWASP mobile application security.” Accessed: Jun. 14, 2023. [Online]. Available: <https://mas.owasp.org/checklists/>
- [8] “CWE - CWE top 25 most dangerous software weaknesses.” Accessed: Jul. 20, 2023. [Online]. Available: <https://cwe.mitre.org/top25/>
- [9] Y. Wang *et al.*, “Identifying vulnerabilities of SSL/TLS certificate verification in Android apps with static and dynamic analysis,” *Journal of Systems and Software*, vol. 167, Sep. 2020, doi: 10.1016/j.jss.2020.110609.
- [10] J. Bucko, “Security of smart banking applications in Slovakia,” *Journal of Theoretical and Applied Electronic Commerce Research*, vol. 12, no. 1, pp. 42–52, Jan. 2017, doi: 10.4067/S0718-18762017000100004.
- [11] M. Botacin, A. Kalysch, and A. Grégio, “The internet banking [in]security spiral: past, present, and future of online banking protection mechanisms based on a Brazilian case study,” in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Aug. 2019. doi: 10.1145/3339252.3340103.
- [12] R. Kaur, Y. Li, J. Iqbal, H. Gonzalez, and N. Stakhanova, “A security assessment of HCE-NFC enabled e-wallet banking android apps,” in *Proceedings - International Computer Software and Applications Conference*, IEEE Computer Society, Jun. 2018, pp. 492–497. doi: 10.1109/COMPSAC.2018.10282.
- [13] “Navigating the ember months: cybersecurity measures for Nigerians and the Banking Ecosystem - NIBSS.” Accessed: Dec. 26, 2023. [Online]. Available: <https://nibss-plc.com.ng/navigating-the-ember-months-cybersecurity-measures-for-nigerians-and-the-banking-ecosystem/>
- [14] B. S. Omotosho, “Analysing user experience of mobile banking applications in Nigeria: a text mining approach,” *Central Bank of Nigeria Journal of Applied Statistics*, vol. 12, no. No. 1, pp. 77–108, Aug. 2021, doi: 10.33429/cjas.12121.4/6.
- [15] O. J. Ayangbekun, O. F. Bankole, and B. A. Saka, “Analysis of security mechanisms in Nigeria E-banking platform,” *International Journal of Electrical and Computer Engineering*, vol. 4, no. 6, pp. 837–847, Dec. 2014, doi: 10.11591/ijece.v4i6.6857.
- [16] K. E. Orjiude and C. O. Yinka-Banjo, “A multilateral privacy impact analysis method for android applications,” *Annals of Science and Technology*, vol. 7, no. 2, pp. 1–20, Dec. 2022, doi: 10.2478/ast-2022-0005.
- [17] “Central Bank of Nigeria | Commercial Banks.” Accessed: Nov. 13, 2022. [Online]. Available: <https://www.cbn.gov.ng/Supervision/Inst-DM.asp>
- [18] “Mobile OS share in Nigeria 2018-2023 | Statista.” Accessed: Dec. 23, 2023. [Online]. Available: <https://www.statista.com/statistics/1063846/market-share-held-by-mobile-operating-systems-in-nigeria/>
- [19] “Mobile security framework (MobSF).” Accessed: May 21, 2023. [Online]. Available: <https://github.com/MobSF/Mobile-Security-Framework-MobSF>
- [20] E. Chatzoglou, G. Kambourakis, and V. Kouliaridis, “A multi-tier security analysis of official car management apps for android,” *Future Internet*, vol. 13, no. 3, pp. 1–35, Mar. 2021, doi: 10.3390/fi13030058.





- [21] R. Chandra *et al.*, “Mobile application security in the health care and finance sector with static analysis (Tools: MobSF),” Institute of Electrical and Electronics Engineers (IEEE), Dec. 2023, pp. 1–7. doi: 10.1109/ict60153.2023.10374057.
- [22] G. Lamalva and S. Schmeelk, “MobSF: mobile health care android applications through the lens of open source static analysis,” in *2020 IEEE MIT Undergraduate Research Technology Conference, URTC 2020*, Institute of Electrical and Electronics Engineers Inc., 2020. doi: 10.1109/URTC51696.2020.9668870.
- [23] P. K. Roy and K. Shaw, “A credit scoring model for SMEs using AHP and TOPSIS,” *International Journal of Finance and Economics*, vol. 28, no. 1, pp. 372–391, Jan. 2023, doi: 10.1002/ijfe.2425.
- [24] E. Triantaphyllou, “Multi-criteria decision making methods,” 2000. doi: 10.1007/978-1-4757-3157-6_2.
- [25] I. M. Almomani and A. Al Khayer, “A comprehensive analysis of the android permissions system,” *IEEE Access*, vol. 8, pp. 216671–216688, 2020, doi: 10.1109/ACCESS.2020.3041432.
- [26] “Security guidelines | App quality | Android Developers.” Accessed: Jul. 23, 2023. [Online]. Available: <https://developer.android.com/privacy-and-security/security-tips>

BIOGRAPHIES OF AUTHORS







Abdullahi Yahya Imam     obtained master's degree in information security and cyber forensics from the school of information technology, SRM University India, and his Bachelor's degree in Computer Science from Bayero University Kano, Nigeria. He is currently a lecturer in the Department of Information Technology, Bayero University Kano, Nigeria. His research interests focus on applied cryptography, cybersecurity, and mobile applications security. He can be contacted at email: ayimam.it@buk.edu.ng.



Hamisu Ibrahim Usman     received his master's degree in information security and cyber forensics from SRM University, India in 2015, and his first degree in Computer Science from Bayero University Kano in 2011. Currently a lecturer at the Department of Information Technology Faculty of Computing Bayero University, Kano Nigeria. His research interests focus on cybersecurity, incident response, and machine learning. He can be contacted at email: hiusman.it@buk.edu.ng.



Abdulrazaq Abba     received his Ph.D. degree from the Department of Computer Science, University of York, UK in 2021, his M.Sc. from the University of Leicester, UK in 2014, and his first degree from Bayero University Kano in 2011. Currently a lecturer at the Department of Computer Science and Digital Technologies at the University of East London. His research interests focus on software security engineering, applications of AI, specifically machine learning, and learning technologies. He can be contacted at email: ahagnmj@gmail.com.