Security Testing Methods for Android: An InsecureBankv2 Case Study

Hoàng Thanh Tú
Thai Nguyen University of Information and Communication Technology
Thai Nguyen University, Thai Nguyen, Vietnam
student508@ictu.edu.vn

***Abstract*—This thesis reviews information security and penetration-testing methods for Android applications, then applies selected checks to the intentionally vulnerable InsecureBankv2 application. The author uses static analysis with MobSF and practical inspection with Android debugging and a web proxy, documenting configuration, test steps, and resulting screenshots. The source discusses exposed application components, locally stored or logged information, network traffic, and a reported SQL-injection finding. It states that 15 of 22 planned testing factors were exercised in the demonstration and reports a low MobSF security score for the sample application. These observations concern a training target; they do not establish the vulnerability rate of ordinary Android applications. The report provides a concrete example of how static and dynamic checks can be combined, while acknowledging that the study is limited to one application and a subset of the planned factors. Future work proposes testing other platforms and more complex applications. The contribution is a documented educational security assessment, with its results bounded by the chosen test target and tools.**

***Index Terms*—Android security, penetration testing, InsecureBankv2, MobSF, static analysis, dynamic analysis**

I. Introduction

Mobile applications hold data that may be exposed through insecure storage, component configuration, logs, or network communication. The thesis introduces Android application security and penetration testing as a way to inspect those risks. It selects InsecureBankv2, a deliberately insecure training application, for the practical demonstration. The work aims to build and document a testing workflow, rather than to make claims about the security of unrelated real banking software [1]. Earlier ICTU student work on related designs is included here for context [3], [4]. Related published studies provide context for the subject and methods [5], [6], [7], [8], [9], [10]. These contextual works were selected for this English summary; they are not asserted to appear in the original thesis bibliography.

II. Methods

The author reviews Android security concepts, then combines static inspection with dynamic observation of the InsecureBankv2 sample. MobSF is used to analyze the application package and summarize findings. Android debugging tools inspect components, local data, and logs; a proxy is used to observe application traffic during testing. The report organizes planned checks around storage, exposed activities and providers, certificate handling, and network behavior. Screenshots record the tool outputs and selected test results [1].

III. Results

The source reports completion of 15 of 22 planned testing factors on the sample application. MobSF output and subsequent checks identify weak handling of sensitive information and exposed application behavior; the report also discusses a SQL-injection finding. Screenshots show the analysis process and examples of data accessible during tests. These findings are specific to InsecureBankv2. The thesis does not report independent replication or a systematic study across a representative collection of Android apps [1].

IV. Discussion

The case study illustrates how multiple tools can reveal different classes of Android security weaknesses, but a vulnerable training app is selected precisely to contain faults. Its findings should therefore be interpreted as demonstrations of testing methods. The author notes that some planned factors were not completed and proposes more complex targets and additional tools for future work. Testing authorization and controlled scope would be essential before applying the same methods to production applications [1].

V. Conclusion

The thesis documents a static and dynamic security-testing exercise on InsecureBankv2 and records the observed weaknesses. It supports an educational workflow for examining Android application components, storage, and traffic. The evidence is limited to one deliberately vulnerable target and does not measure general Android security. Further work should broaden the test set and complete the planned checks under an authorized evaluation scope [1].

Supervisor: Vũ Việt Dũng, Thai Nguyen University of Information and Communication Technology (vvdung@ictu.edu.vn).

References

[1] Hoàng Thanh Tú, ”Security Testing Methods for Android: An InsecureBankv2 Case Study,” internal undergraduate thesis (original in Vietnamese), Thai Nguyen University of Information and Communication Technology, 2024.

[2] OWASP, ”Android application security resources,” https://owasp.org/, listed in the source thesis.

[3] ICTU Student 43, ”Software Testing and Application of Designing an Automated Testing Tool,” internal undergraduate thesis, Thai Nguyen University of Information and Communication Technology, 2008.
[4] ICTU Student 170, ”A Study on Software Testing Techniques and Website Application Development,” internal undergraduate thesis, Thai Nguyen University of Information and Communication Technology, 2010.
[5] Hee-Hoon Cho et al., ”A Study on the Security Vulnerability of Android,” Asia-pacific Journal of Multimedia Services Convergent with Art, Humanities, and Sociology, 2015. doi:10.14257/ajmahs.2015.12.04.
[6] Okolie C.C, ”Penetration Testing for Android Smartphones,” IOSR Journal of Computer Engineering, 2013. doi:10.9790/0661-143104109.
[7] Shivi Garg et al., ”Android security assessment: A review, taxonomy and research gap study,” Computers & Security, 2021. doi:10.1016/j.cose.2020.102087.
[8] Deco Aprilliansyah et al., ”Penetration Testing with OWASP Mobile for Android Security Optimization,” Journal of Cyber Health and Computer, 2023. doi:10.64163/jochac.v1i1.3.
[9] Sébastien Salva et al., ”APSET, an Android aPplication SEcurity Testing tool for detecting intent-based vulnerabilities,” International Journal on Software Tools for Technology Transfer, 2014. doi:10.1007/s10009-014-0303-8.
[10] William Enck et al., ”Understanding Android Security,” IEEE Security & Privacy Magazine, 2009. doi:10.1109/msp.2009.26.