ISSN 2356-5462
https://journals.telkomuniversity.ac.id/ijoict
Intl. Journal on ICT
Vol. 11, No. 1, Jun 2025. pp. 36-49
doi.org/10.21108/ijoict.v11i1.9094

Benchmarking Mobile Apps Security in Universities: An OWASP Mobile Top 10 Framework Perspective

Fajar Maulana Kadir[1]*, Muhamad Irsan[2], Aji Gautama Putrada[3]

[1,2,3] *School of Computing, Telkom University*
Bandung, Indonesia

* mooreph@student.telkomuniversity.ac.id

Abstract

Leading universities in Indonesia such as Telkom University (Tel-U), Bandung Institute of Technology (ITB), University of Indonesia (UI), and Gadjah Mada University (UGM) have developed mobile-based academic information systems that improve the accessibility of campus services, where sensitive information such as personal data, access credentials, and educational information is stored and managed through mobile applications. The current gap is a lack of understanding of the specific vulnerability profile of campus mobile applications and how these vulnerabilities can affect the security of educational institution data. This study conducted a comparative analysis of vulnerabilities in campus mobile applications using the OWASP (Open Web Application Security Project) Mobile Top 10 2024 as the testing framework, selected due to its widely global recognition. This study employed three static application security testing (SAST) tools: AndroBugs, Mobile Security Framework (MobSF), and QARK (Quick Android Review Kit). There are ten vulnerability categories in OWASP Mobile Top 10 2024, being M1, M2, M3, M4, M5, M6, M7, M8, M9, and M10. The test results show that MySIX ITB and WeAreUI have the most vulnerabilities compared to the other two campuses, with MySIX ITB being the most vulnerable application, with vulnerabilities in five OWASP categories. Additionally, the two most frequently occurring categories were M6 (Inadequate Privacy Control) and M8 (Security Misconfiguration), indicating issues in the university mobile application architecture. Finally, there was knowledge that all four mobile applications had resistance to M2 category (Inadequate Supply Chain Security), indicating that the use of third-party libraries is safe.

Keywords: *OWASP, Mobile Security, Vulnerability Assessment, Mobile Application, Static Analysis Tool*

I. INTRODUCTION

The global digital transformation development has changed the way universities in Indonesia provide services to academics [1]. This can be seen from various leading universities such as the Bandung Institute of Technology (ITB), University of Indonesia (UI), Gadjah Mada University (UGM), and Telkom University, which have developed mobile-based academic information systems that increase the accessibility of campus services. Sensitive information such as personal data, credential access, and educational information is stored and managed through these mobile applications [2].

On the other hand, several studies highlight the problems in campus mobile applications. Titiakarawongse *et al.* [3] stated that increasing campus mobile application use also increases security risks in handling sensitive

Received on 09 May 2025. Revised on 19 Jun 2025. Accepted and Published on 11 Jul 2025.

data. Utama *et al.* [4] stated that most higher education institutions do not have standard procedures for evaluating the security of their mobile applications. The current gap is the significant difference in vulnerability among several campus applications and how this difference affects data security in the education sector, which is a problem.

There are various frameworks that can test vulnerabilities in mobile applications, including campus mobile applications. Among these frameworks, OWASP (Open Web Application Security Project) offers a variety of more specific approaches, such as the OWASP Mobile Top 10 2024, which is specifically designed to identify and address vulnerability risks in mobile applications. The advantages of the OWASP Mobile Top 10 lie in its broad vulnerability risk coverage, update routine, and broad support from the global community [5], thus making it an appropriate choice as a campus mobile application security test. In several studies conducted on campus mobile applications, such as that conducted by Dhingra *et al.* [6] , it was stated that OWASP Mobile Top 10 is a globally recognized benchmark for classifying mobile application security vulnerabilities. In another study, Wicaksana *et al.* [7] noted that this framework has a systematic methodology for identifying and assessing mobile application vulnerabilities, particularly in an educational context. Although many studies have highlighted the global importance of the OWASP Mobile Top 10 2024, its application in the education sector remains insufficient, especially in Indonesia. Therefore, this research presents an opportunity to utilize the OWASP Mobile Top 10 2024 as a framework that can serve as a benchmark for classifying mobile application vulnerabilities in campus environments.

This study uses OWASP Mobile Top 10 2024 to conduct a comparative analysis of mobile campuses between universities, from public to private. In its implementation, tools are needed to assess vulnerabilities. This study will use three static application security testing (SAST): Androbugs, Mobile Security Framework (MobSF), and Quick Android Review Kit (QARK) as tools for vulnerability assessment. These three tools were chosen because of their ability to detect various types of vulnerabilities covered in the OWASP Mobile Top 10 2024. By comparing vulnerability analysis results on different campus mobile applications, this study aims to identify common vulnerability patterns and recommend improvements based on the OWASP Mobile Top 10 2024 security standards. There are ten vulnerability categories in OWASP Mobile Top 10 2024, being M1, M2, M3, M4, M5, M6, M7, M8, M9, and M10. The scope of this study only limited the uses of assessment of mobile applications in Android platform only. The iOS applications is not included due architectural differences between Android and iOS. Furthermore, this study only focuses on static analysis and does not include dynamic analysis.

To the best of our knowledge, there has never been a study that comprehensively compares vulnerability analysis of mobile applications with the OWASP Mobile Top 10 2024 in Indonesia. Previous studies have focused more on one application or one institution, without comparing between universities and without using a multi-tool approach in vulnerability detection. In addition, most existing studies are conducted in other sectors such as banking, health, or e-commerce, with characteristics and security needs different from those of the academic environment. Therefore, the following list is the contribution of this study:

1. A research methodology that strengthens the knowledge that AndroBugs, MobSF, and QARK are reliable for testing M2 (Inadequate Supply Chain), M6 (Inadequate Privacy Controls), and M8 (Security Misconfiguration) in the OWASP Mobile Top 10 2024 category.
2. A knowledge that QARK is a SAST tool that produces different results in detecting vulnerabilities compared to AndroBugs and MobSF.
3. Knowledge that vulnerabilities M6 and M8 in OWASP Mobile Top 10 2024 categories, or Inadequate Privacy Controls and Security Misconfiguration, respectively, require the most attention in campus mobile apps, especially from four Indonesian campuses: Telkom University, ITB, UI, and UGM.

The remainder of this paper has a systematic structure: Section II discusses recent research related to this topic; Section III explains the research methodology; Section IV presents the results and discusses them; and Section V concludes the research findings and provides suggestions for further research.

II. LITERATURE REVIEW

Several researchers have made significant progress in understanding mobile application vulnerabilities using the OWASP Mobile Top 10 2024. Titiakarawongse *et al.* [3] conducted a study that combined the OWASP Mobile Top 10 framework in the context of mobile application security, where researchers compared vulnerability analysis on banking applications in Thailand and non-Thailand. The researchers said using the OWASP Mobile Top 10 can detect vulnerabilities in these applications. Utama *et al.* [4] studied academic information system vulnerabilities using the OWASP Framework to reveal the complexity of digital security threats in the university environment. The study conducted at 36 universities with information systems from XYZ Company showed that digital security vulnerabilities are not just technical risks, but strategic challenges that require a comprehensive approach. Table I summarizes our discussion about the related works and highlights our research contributions.

TABLE I
SUMMARY OF RELATED WORKS IN MOBILE APPLICATION SECURITY ANALYSIS

Reference	Paper Highlight	OWASP Mobile Top 10	AndroBugs	MobSF	QARK	University Case Study
Titiakarawongse *et al.* [3]	OWASP Mobile Top 10 Framework with AndroBugs, MobSF, and QARK on Banking	✓	✓	✓	✓	✗
Utama & Nurhadi [4]	PTES and OWASP on academic information systems	✓	✗	✗	✗	✗
Kusreynada & Barkah [8]	Static and dynamic analysis of MobSF on Mobile JKN	✗	✗	✓	✗	✗
Almohaini *et al.* [9]	Dynamic analysis of MobSF for ransomware detection	✗	✗	✓	✗	✗
Mohsen *et al.* [11]	Development and enhancement of AndroBugs framework	✗	✓	✗	✗	✗
Al-Delayel *et al.* [13]	Security Assessment on m-banking in Qatar using QARK and MobSF	✗	✗	✓	✓	✗
Proposed Method	OWASP Mobile Top 10 Framework with AndroBugs, MobSF, and QARK at University	✓	✓	✓	✓	✓

Several researchers have made significant progress in understanding mobile application vulnerabilities using the MobSF tool. Research conducted by Kusreynada *et al.* [8] revealed that MobSF can show the ability to identify various vulnerabilities.. Then Almohaini *et al.* [9] conducted a ransomware detection study, which showed that MobSF can achieve 100% detection accuracy in dynamic analysis, confirming its position as a powerful security analysis tool. Research by Flood *et al.* [10] showed that static analysis performed by MobSF

also reinforces that MobSF can reveal crucial information such as dangerous permissions or the importance of security of sensitive information, providing a comprehensive picture of potential security vulnerabilities in mobile applications. Research by Kohli *et al.* [11] on COVID-19 tracking applications further strengthens the importance of mobile security testing.

Several researchers have made significant progress in understanding mobile application vulnerabilities using the AndroBugs tool. Mohsen *et al.* [12] study on AndroBugs revealed significant challenges in mobile application security, with a focus on upgrading AndroBugs in order to be up to date with the vulnerabilities of mobile applications that are increasingly challenging and constantly evolving.

Several researchers have made significant progress in understanding mobile applications vulnerabilities using QARK tool. Al-Delayel *et al.* [13] study on QARK revealed m-banking in Qatar show many poor security practices and major security issues which further reinforces the need for mobile app security testing, and raised awareness of security posture will have real impact on m-banking, and user confidence.

III. RESEARCH METHOD

This study uses a comprehensive methodology that combines static application security testing (SAST) to identify vulnerabilities in selected campus mobile applications. As summarized in Fig. 1, the process illustrates the flow of the research methodology used in the vulnerability analysis of mobile applications. The process begins with selecting four different applications—My Tel-U, MySIX ITB, WeAreUI, and Simaster UGM—then continues with preparing three static security analysis testing tools: AndroBugs, MobSF, and QARK. Each application is analyzed one by one using the three tools to identify potential vulnerabilities. If vulnerabilities are found, the results are recorded as a report, which is then classified based on the OWASP Mobile Top 10 2024 categories. This procedure is repeated until all applications are analyzed. After that, all classified vulnerability data is collected for statistical analysis. This stage aims to identify each application's pattern and security level. With this approach, the study focuses on detection and a comprehensive evaluation of the security posture of mobile applications in the academic environment. Fig. 1 shows the flowchart of the research methodology that we carried out.

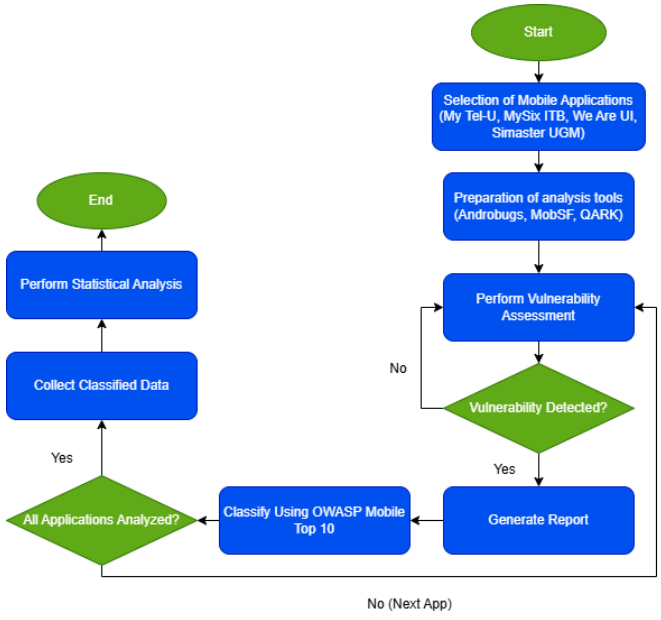

Fig. 1. Flowchart of Research Methodology

A. Selection of Mobile Application and SAST Tools

Chiboora *et al.* [14] divided the risk level based on the number of downloads of the application on the Google Play Store; the greater the number of downloads, the greater the vulnerability. The division is as follows: Applications with downloads above 50,000 are highly risky and are classified as top tier. Applications with download numbers between 5,000 and 50,000 have a medium risk and are classified as middle tier. Applications with download numbers between 500 and 5,000 have a low risk and are classified as low tier. This study used several university applications, which are classified as top and middle tier, namely: Telkom University and UGM applications for the top tier, then UI and ITB applications for the middle tier. Table II presents complete details of the campus mobile applications selected in this study.

TABLE II
CAMPUS MOBILE APPS USED

Applications	University	Versions	Size (MB)	Type	Downloads
MyTel-U	Telkom University	2.0.0	77.00	Private	50K+
MySIX ITB	ITB	2.0.0	24.24	Public	10K+
We Are UI	UI	1.1.12	23.45	Public	5K+
Simaster UGM	UGM	3.4.4	10.60	Public	100K+

SAST is an important methodology for checking the vulnerability of an application by analyzing the source code for vulnerabilities before the source code is executed [15]. The methodology used is important for how developers can improve application security from the SAST results. SAST can perform vulnerability assessment analysis directly without seeing the source code of the sample application, simplifying the assessment process [16]. The three SAST tools that this study proposes for testing are AndroBugs, MobSF, and QARK [17]. The use of multiple tools facilitates this research because the tools mentioned have unique and complementary benefits.

1. AndroBugs
   AndroBugs is a static security analysis framework for Android applications developed to identify potential security vulnerabilities in application code [18]. This tool performs automatic scans in the application code to look for weaknesses/vulnerabilities and creates a report of the vulnerabilities found [12].
2. Mobile Security Framework (MobSF)
   Mobile Security Framework or commonly known as MobSF is a SAST Framework that can perform malware analysis, security assessment, and pen-testing if possible automatically by performing static analysis and dynamic analysis for Android and iOS mobile applications [19].
3. Quick Android Review Kit (QARK)
   Quick Quick Android Review Kit or commonly known as QARK is a SAST framework developed by LinkedIn in an open source to identify vulnerabilities of android applications in APK or source code. [20].

As Table III shows, these three tools have complementary characteristics and capabilities in the mobile application security analysis process.

TABLE III
CHARACTERISTICS AND CAPABILITIES OF THE MOBILE APPLICATION SECURITY ANALYSIS TOOLS

Aspects	AndroBugs	MobSF	QARK
Analysis Type	Static	Static & Dynamic	Static
Input Format	APK	APK & IPA	APK & Source Code
Analysis Speed	Fast	Medium	Medium
Report Details	Medium	High	High
Analysis Focus	Component & Configuration	Comprehensive	Source Code & Component

Some tools, such as MobSF, provide a code export feature that allows manual application code review in a more readable format [21]. This approach integrates automated analysis with manual review, ensuring that vulnerabilities that tools may miss can still be accurately identified.

B. Vulnerability Assessment with OWASP Mobile Top 10

OWASP Mobile Top 10 2024 categories must be monitored by developers when developing applications. Based on updates from Priambodo *et al.* [22], the OWASP Mobile Top 10 categories can contain insecure data storage. Then Kurniawan *et al*. [23] added three categories: Insecure Communication, Insufficient Cryptography, and Client Code Quality. Finally, Gil *et al.* [24] added one category, namely reverse engineering. The OWASP Mobile Top 10 is updated regularly to address evolving threats in the mobile security landscape [25]. The vulnerability finding analysis report created by SAST will be manually checked to be classified into the latest OWASP Mobile Top 10 which is version 2024. Table IV lists the OWASP Mobile Top 10 2024 categories.

TABLE IV
OWASP MOBILE TOP 10 2024

Category	Threat	Description
M1	*Improper Credential Usage*	Vulnerabilities due to improper handling or storage of user data
M2	*Inadequate Supply Chain Security*	Vulnerabilities stemming from the use of third-party components such as libraries, SDKs, and dependencies that are insecure or potentially malicious.
M3	*Insecure Authentication/Authorization*	Vulnerabilities in process verification identification, resulting in non-authorized user access to the application.
M4	*Insufficient Input/Output Validation*	Vulnerabilities due to inadequate input and output validation, opening up gaps for various types of injections such as SQL injection and XSS.
M5	*Insecure Communication*	Vulnerabilities due to using unencrypted or non-secure communication
M6	*Inadequate Privacy Controls*	Vulnerabilities in the handling and protection of users' personal data that could result in the leakage of sensitive information.
M7	*Insufficient Binary Protections*	Vulnerability in the protection of application binary code that allows reverse engineering or manipulation of the application.
M8	*Security Misconfiguration*	Vulnerabilities resulting from security misconfiguration, including insecure default settings and server misconfiguration.
M9	*Insecure Data Storage*	Vulnerability in local data storage that could result in unauthorized access to sensitive application data.
M10	*Insufficient Cryptography*	Vulnerabilities in the implementation of cryptography, including the use of weak algorithms or improper implementation.

The occurrence process ensures classification consistency between applications and provides an objective basis for comparison [26]. For each campus mobile application (MyTel-U, MySIX ITB, We Are UI, and Simaster UGM), the number of vulnerabilities in each OWASP category is calculated and compared using the following formula:

$$Occurence(\%) = \frac{x}{n} \times 100 \quad (1)$$

This approach provides an overview of the distribution of vulnerabilities specific to each application, where x is the number of tools from AndroBugs, MobSF, and QARK that detected vulnerabilities in each category, and n = 12 is the total number of assessments (4 applications × 3 tools).

After using the formula, the 95% confidence interval will be used to calculate how accurate or reliable each occurrence distribution is [27]. The calculation uses the following binomial proportion formula:

$$CI = p \pm z \times \sqrt{\frac{p(1-p)}{n}} \tag{2}$$

where $p = \frac{x}{n}$, is the observed proportion, $z = 1.96$ is the z-score for a 95% confidence level, and $n = 12$.

C. Analysis and Data Comparison

After the vulnerability classification using OWASP Mobile Top 10 2024 is completed, we will use statistical analysis to evaluate and also compare the vulnerabilities of the campus mobile applications that have been selected and assessed. The data analysis involves several statistical methods to measure and interpret the distribution of vulnerabilities and differences between applications and tools.

First, we calculated the average number or means of vulnerabilities found per identified application. The average $(\bar{x})$ is calculated using the following formula:

$$\bar{x} = \frac{\sum_{i=1}^{n} x_i}{n} \tag{3}$$

where x_i represents the number of vulnerabilities detected in each application, and n is the total number of observations.

Second, after the average number of vulnerabilities or means is calculated, the standard deviation or S.D is used to determine the vulnerability variability between applications, which is the formula:

$$S.D = \sqrt{\frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})^2} \tag{4}$$

where $\bar{x}$ is the mean value.

Third, the t-value is used to calculate how statistically significant the differences in vulnerabilities between applications are with standardized values, which is done on a one-sample basis [29]. The t-value is calculated using:

$$t = \frac{\bar{x} - u}{s/\sqrt{n}} \tag{5}$$

where $\bar{x}$ is the sample mean (for a particular application), μ is the population mean (the mean of other applications), s is the standard deviation of different applications, and n is the sample size.

Finally, the p value is used to test whether the t value makes a significant difference. The p value will be considered statistically significant or not by chance if the p value is less than 0.05 ($p < 0.05$) [30]. The results of this analysis are expected to provide in-depth insights into various security threats to campus mobile applications and recommendations for better risk mitigation.

IV. RESULTS AND DISCUSSION

A. Results

The first step was to find out how many vulnerabilities in each campus mobile application, which will be classified by OWASP Mobile Top 10 2024. The results of the SAST show that the applications from ITB and UI, MySIX ITB and WeAreUI, get 24 out of 30 vulnerabilities which is the highest, inversely proportional to the application from Telkom University, My Tel-U, which gets 21 out of 30 vulnerabilities, while the application from UGM, Simaster UGM, is in between with 22 out of 30 vulnerabilities. From these findings, it is concluded that the best SAST is Androbugs which gets an average finding of 5.00 vulnerabilities per application, followed by QARK 4.75 per application, and finally MobSF 4.00 per application. These applications were tested in March 2025 and using versions of My Tel-U 2.0.0, MySIX ITB 2.0.0, We Are UI 1.1.12, and Simaster UGM 3.4.4. Table V presents a comprehensive classification of the vulnerabilities identified in each mobile campus application, categorized according to the findings from the three analysis tools: AndroBugs, MobSF, and QARK.

TABLE V
OVERALL VULNERABILITY CLASSIFICATION

OWASP	MyTel-U			MySIX ITB			We Are UI			Simaster UGM		
Category	1	2	3	1	2	3	1	2	3	1	2	3
M1	-	√	-	-	-	-	-	√	-	-	√	-
M2	-	-	-	-	-	-	-	-	-	-	-	-
M3	√	-	-	√	√	√	√	√	-	-	-	√
M4	-	√	-	-	-	-	√	√	-	√	-	-
M5	√	√	-	√	√	√	√	√	√	√	√	-
M6	√	√	√	√	√	√	√	√	√	√	√	√
M7	-	-	-	-	-	-	√	-	-	√	-	-
M8	√	√	√	√	√	√	√	√	√	√	√	√
M9	√	√	-	√	√	√	√	√	-	√	√	-
M10	-	√	-	-	√	-	√	√	-	-	√	-

Notes: 1 For AndroBugs, 2 For MobSF, and 3 For QARK. Testing in March 2025

Using three SAST tools, we evaluated each app using the OWASP Mobile Top 10 2024 categories, from M1 to M10. The most frequently detected vulnerability across tested apps was M6 (Inadequate Privacy Controls), identified in 100% of apps by at least one tool, followed by M8 (Security Misconfiguration) at 100%, and M5 (Insecure Communications) at 91.7%. These findings indicate consistent weaknesses in privacy management, and secure communication practices across campus mobile apps. In contrast, M2 (Inadequate Supply Chain Security) was the least frequently detected vulnerability, found in 0% of cases, indicating the relative strength or lack of visibility in issues related to third-party dependencies. Fig. 2 presents summary statistics of vulnerability occurrences across all apps and 95% confidence intervals.

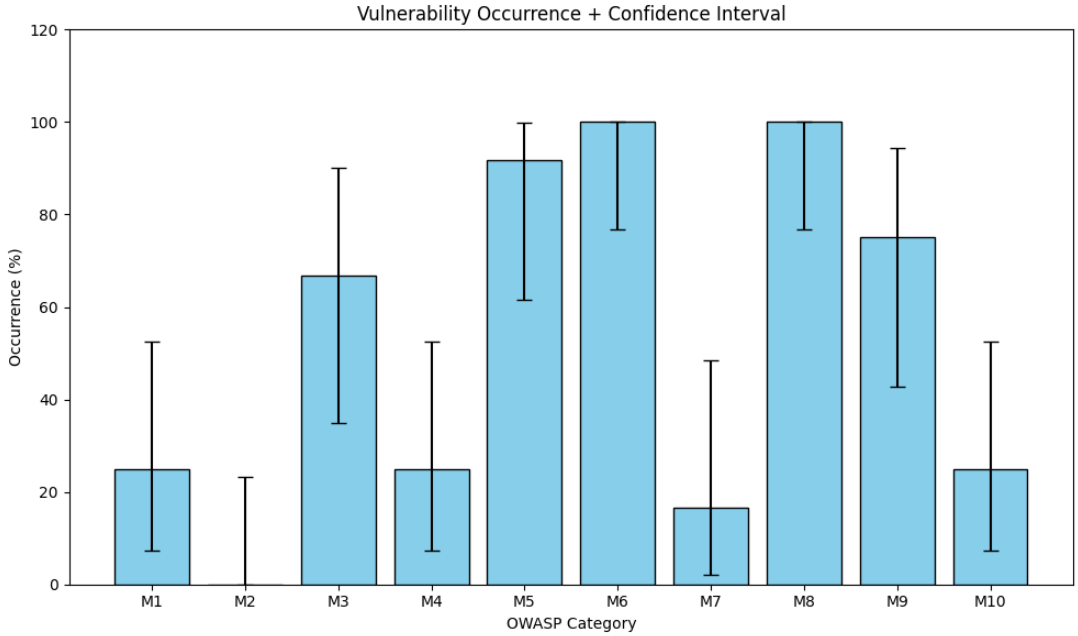

Fig. 2 Overall Vulnerability Occurrence

Among these applications, WeAreUI and MySIX ITB showed the highest number of detected vulnerabilities among the three tools, each with 24 out of 30 possible detections (10 OWASP categories × 3 tools). In contrast, My Tel-U performed slightly better with 21 out of 30 vulnerabilities, and Simaster UGM followed with 22 out of 30 vulnerabilities. This variation indicates different levels of attention to secure development practices among the university teams responsible for these applications. For each application, the table includes the number of detected vulnerabilities across all OWASP Mobile Top 10 2024 categories and tools and the mean and standard deviation of the other three applications for comparison. A one-sample t-test was performed to assess whether the security posture of each application differs significantly from the others. The resulting t-values and p-values are provided to evaluate the statistical significance of the observed differences.

Table VI presents a detailed comparative analysis of mobile campus applications using the total number of detected vulnerabilities as the main metric.

TABLE VI
COMPARISON OF VULNERABILITIES DETECTED IN EACH MOBILE APPLICATION

App	Total Vulnerability	Means	S.D (Standard Deviation)	t-value	p-value	Description
MyTel-U	21 out 30	0.47	0.50	3.50	0.073	The difference is not statistically significant
MySIX ITB	24 out 30	0.53	0.50	-1.89	0.199	The difference is not statistically significant
We Are UI	24 out 30	0.63	0.48	-1.89	0.199	The difference is not statistically significant
Simaster UGM	22 out 30	0.50	0.50	1.00	0.423	The difference is not statistically significant

The standard deviation (SD) between the groups ranged from 1.15 to 1.73, indicating relatively low variability in security posture among the applications. To assess the statistical significance of these differences, a one-sample t-test was performed for each application, comparing its number of vulnerabilities to the average of the other three applications. The results for My Tel-U yielded the highest t-value (3.50) with a p-value of

0.073, indicating a significant difference, although not statistically significant at the conventional threshold of $p < 0.05$. Similarly, MySIX ITB and WeAreUI showed t-values of -1.89 ($p = 0.199$), and Simaster UGM had a t-value of 1.00 ($p = 0.423$). Although the statistical test did not confirm a significant difference between the applications, these findings suggest that My Tel-U may have a slightly better security posture than the others, with fewer vulnerabilities detected.

We analyzed the distribution of security issues across each OWASP Mobile Top 10 2024 categories for further analysis, calculating the average number of vulnerabilities detected per app. Fig. 3 presents these detailed findings. This average is calculated across the three tools used (AndroBugs, MobSF, QARK), where each cell represents the proportion of tools that detected the respective vulnerability (e.g., 1.00 = detected by all three tools). This comparison allows a detailed view of vulnerabilities across apps and highlights which OWASP categories are consistently more exposed. Categories M6, M8, and M5 emerge as the most frequently detected across all apps, indicating common privacy, configuration, and communication vulnerabilities.

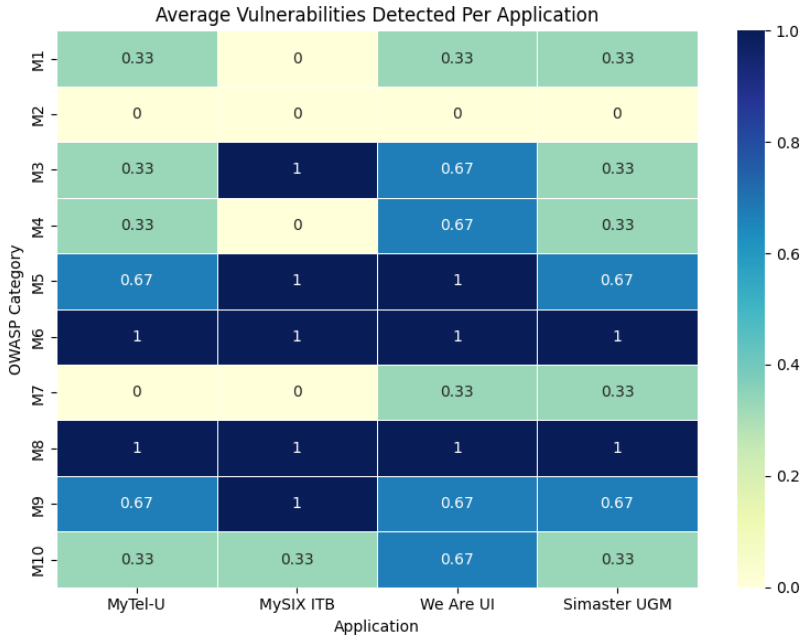

Fig. 3. Average Vulnerabilities Detected Per Application

To analyze campus mobile applications, this study used three security analysis tools—AndroBugs, MobSF, and QARK. The results show varying effectiveness among these tools. AndroBugs showed the highest detection ability, identifying 20 vulnerabilities across all applications, followed by QARK with 19 vulnerabilities. MobSF, while still valuable, detected a slightly lower total of 16 vulnerabilities. Table VII compares the effectiveness of the three tools used in this study. Effectiveness is measured by the average number of vulnerabilities detected by each tool across the four campus mobile applications. AndroBugs detected an average of 5.00 vulnerabilities per application (SD = 0.00), indicating perfect consistency in its detection results across the four applications. QARK performed comparably, identifying 4.75 vulnerabilities per application (SD = 0.96), with slightly higher variability. On the other hand, MobSF detected the fewest vulnerabilities on average - 4.00 per application (SD = 0.82) - which may indicate its relatively lower effectiveness in uncovering issues. The standard deviations offer further insight into the behavior of these tools. While AndroBugs was the most consistent, reporting the same number across all applications, QARK and MobSF showed more variability. This variability suggests that their performance may depend more on the structure of a particular application or implementation differences.

TABLE VII
EFFECTIVENESS OF SECURITY ANALYSIS TOOLS

Tools	Raw Vulnerability Counts	Mean Vulnerabilities Detected per App	S.D. (Standard Deviation)
AndroBugs	20	5.00	0.00
MobSF	16	4.00	0.82
QARK	19	4.75	0.96

B. Discussion

Tudela *et al.* [31] used a combination of SAST and dynamic black box security analysis (DAST) to improve vulnerability detection of the OWASP Mobile Top 10 2024. Meanwhile, Shen *et al.* [32] explored the reliability of SAST tools by testing them on open-source embedded software. On the other hand, Li *et al.* [33] said that using SAST tools and OWASP Mobile Top 10 can improve reliability in vulnerability testing. However, there has never been a study that combines several SAST tools to check the reliability of each tool. This study found that each tool consistently says that M6 and M8 are vulnerable in every campus mobile app, which M6 often arises due to absence of encryption on personal data and weak internal access control, and M8 arises due to unsecured default configuration of security settings, permissions, and controls. This indicates that each tool is reliable for M6 and M8. In addition, each tool also agrees that M2 is not vulnerable in every campus mobile app. The contribution of our study is a research methodology that shows that AndroBugs, MobSF, and QARK are reliable for testing M2, M6, and M8 in the OWASP Mobile Top 10 2024.

Chen *et al.* [34] stated that several SAST tools, including AndroBugs, MobSF, and QARK, have weaknesses, such as high false positives, in detecting vulnerabilities. However, they did not state which tools are the least reliable. Our research shows that QARK provides the most different assessments when used with AndroBugs and MobSF. It is recorded that QARK provides eight different assessments from AndroBugs and MobSF. The contribution of our research is the knowledge that QARK is a SAST tool that produces different results in detecting vulnerabilities compared to AndroBugs and MobSF.

Several studies, such as those by King *et al.* [35] have used the OWASP top 10 to detect vulnerabilities in mobile apps. Silva *et al.* [36] stated that vulnerabilities arise from less-than-ideal mobile app programming practices and following OWASP guidelines can be the solution. Sakhtivel *et al.* [37] did not focus on using OWASP top 10 in mobile applications, but rather on websites. There has never been a study that focuses on using the OWASP Mobile Top 10 2024 to find vulnerabilities in campus mobile apps. Our research contribution is the knowledge that vulnerabilities M6 and M8 in OWASP, or Inadequate Privacy Controls and Security Misconfiguration, respectively, are the vulnerabilities that require the most attention in campus mobile apps, especially from four Indonesian campuses: Telkom University, ITB, UI, and UGM.

Our study provides unique insights into the security challenges faced by Indonesian universities in developing mobile applications, an area that has received little attention in the global literature. Compared to studies such as Paramasivam *et al*. analysis of Malaysian applications [38], our findings show similar patterns of security misconfigurations but with a higher prevalence of privacy control issues.

V. CONCLUSION

This study aims to enrich knowledge about campus mobile apps' vulnerabilities by testing four campus mobile apps using three SAST tools. My Tel-U, MySIX ITB, WeAreUI, and Simaster UGM are the four campus mobile apps. The three SAST tools are AndroBugs, MobSF, and QARK. We also used OWASP Mobile Top 10 2024. The test results show that MySIX ITB and WeAreUI have the most vulnerabilities compared to the other three campuses, with 24 vulnerabilities from three different tools. However, if we look at the consensus between the three tools, MySIX ITB is the most vulnerable application, with vulnerabilities in five categories: M3, M5, M6, M8, and M9. In addition to using three different tools to strengthen the vulnerability detection rate, we also found some new knowledge. The first is that the three tools have the same agreement for detecting M2, M6, and M8, which shows high reliability of the three tools for the categories mentioned. The second is the knowledge that QARK makes the most different decisions from the other two tools. The test results show

that QARK gives different decisions eight times. We also learned that for the four campus mobile apps, developers should pay more attention to two categories detected by each tool, namely M6 and M8, or Inadequate Privacy Controls and Security Misconfiguration, respectively. Finally, there is knowledge that the four mobile apps' strength is resistance to M2; in other words, each campus has used third-party libraries well. Future work is recommended to integrate dynamic analysis and penetration testing and expand the number of application samples, including the iOS platform, to obtain a more comprehensive picture of the security posture of campus mobile apps.

ACKNOWLEDGMENT

The authors conducted this research as part of their research master plan. We want to thank Telkom University, the Directorate of Research and Community Service of Telkom University, the University of Indonesia, Bandung Institute of Technology, and Gadjah Mada University for providing campus mobile applications as samples for this research.

REFERENCES

[1] C. M. Gunawan, L. Rahmania, and I. H. Kenang, "The Influence Of Social Influence And Peer Influence On Intention To Purchase In E-Commerce," *Review of Management and Entrepreneurship*, vol. 7, no. 1, Art. no. 1, Apr. 2023.

[2] M. A. Y. Putranda and I. K. A. Mogi, "Analisis Keamanan pada Aplikasi Udayana Mobile Mengacu pada OWASP Mobile Top 10 2016," *JELIKU (Jurnal Elektronik Ilmu Komputer Udayana)*, vol. 12, no. 3, pp. 655–662, Jan. 2024.

[3] C. Titiakarawongse, S. Taksin, J. Ruangsawat, K. Deeduangpan, and S. Boonkrong, "Comparative Vulnerability Analysis of Thai and Non-Thai Mobile Banking Applications," *Journal of Cybersecurity and Privacy*, vol. 4, no. 3, Art. no. 3, Sep. 2024.

[4] F. P. Utama and R. M. H. Nurhadi, "Uncovering the Risk of Academic Information System Vulnerability through PTES and OWASP Method," *CommIT (Communication and Information Technology) Journal*, vol. 18, no. 1, pp. 39–51, 2024.

[5] M. Liyanage, A. Braeken, S. Shahabuddin, and P. Ranaweera, "Open RAN security: Challenges and opportunities," *Journal of Network and Computer Applications*, vol. 214, p. 103621, May 2023.

[6] Y. Dhingra, V. Ranga, and D. K. Vishwakarma, "A Study of the Security and Privacy Risks in Health-Related Mobile Applications in India," in *2024 International Conference on Communication, Control, and Intelligent Systems (CCIS)*, Dec. 2024, pp. 1–6.

[7] E. P. Wicaksono, "Evaluasi keamanan sistem informasi akademik berbasis mobile dengan menggunakan mobile security framework (mobsf) dan owasp mobile top 10," bachelorThesis, Fakultas Sains dan Teknologi UIN Syarif Hidayatullah Jakarta, 2022.

[8] S. U. Kusreynada and A. S. Barkah, "Android Apps Vulnerability Detection with Static and Dynamic Analysis Approach using MOBSF," *Journal of Computer Science and Engineering (JCSE)*, vol. 5, no. 1, pp. 46–63, 2024.

[9] R. Almohaini, I. Almomani, and A. AlKhayer, "Hybrid-Based Analysis Impact on Ransomware Detection for Android Systems," *Applied Sciences*, vol. 11, no. 22, Art. no. 22, Jan. 2021.

[10] R. Flood, S. C. Chan, W. Chen, and D. Aspinall, "Checking Contact Tracing App Implementations with Bespoke Static Analysis," *SN COMPUT. SCI.*, vol. 3, no. 6, p. 496, Sep. 2022.

[11] N. Kohli and M. Mohaghegh, "Security testing of android based COVID tracer applications," in *2020 IEEE Asia-Pacific Conference on Computer Science and Data Engineering (CSDE)*, IEEE, 2020, pp. 1–6.

[12] F. Mohsen, M. Shehab, U. Aydin, J. van Thuijl, and N. Drong, "Upgrading and Expanding Androbugs to Address Emerging Vulnerabilities," Oct. 12, 2023, *Social Science Research Network, Rochester, NY*: 4600829.

[13] S. A. Al-Delayel, "Security Analysis of Mobile Banking Application in Qatar," Apr. 06, 2023, *arXiv*: arXiv:2202.00582.

[14] T. H. Chiboora, L. Chacha, T. Byagutangaza, and A. Gueye, "Evaluating Mobile Banking Application Security Posture Using the OWASP's MASVS Framework," in *Proceedings of the 6th ACM SIGCAS/SIGCHI Conference on Computing and Sustainable Societies*, in COMPASS '23. New York, NY, USA: Association for Computing Machinery, Aug. 2023, pp. 99–106.

[15] M. Y. Darus, M. Farhan Bin Bolhan, A. Kurniawan, Y. Muliono, C. R. Pardomuan, and M. Mohamad Hata, "Enhancing Web Application Penetration Testing with a Static Application Security Testing (SAST) Tool," in *2023 IEEE 8th International Conference on Recent Advances and Innovations in Engineering (ICRAIE)*, Dec. 2023, pp. 1–6.
[16] A. Nagaraj, B. Sinha, M. Sood, Y. Mathur, S. Gupta, and D. Sitaram, "Learning Algorithms in Static Analysis of Web Applications," Oct. 14, 2022, *arXiv*: arXiv:2210.07465.
[17] J. Zhu, K. Li, S. Chen, L. Fan, J. Wang, and X. Xie, "A Comprehensive Study on Static Application Security Testing (SAST) Tools for Android," *IEEE Trans. Softw. Eng.*, vol. 50, no. 12, pp. 3385–3402, Dec. 2024.
[18] R. Sun, W. Wang, M. Xue, G. Tyson, S. Camtepe, and D. C. Ranasinghe, "An Empirical Assessment of Global COVID-19 Contact Tracing Applications," in *2021 IEEE/ACM 43rd International Conference on Software Engineering (ICSE)*, May 2021, pp. 1085–1097.
[19] B. Yankson, J. V. K, P. C. K. Hung, F. Iqbal, and L. Ali, "Security Assessment for Zenbo Robot Using Drozer and mobSF Frameworks," in *2021 11th IFIP International Conference on New Technologies, Mobility and Security (NTMS)*, Apr. 2021, pp. 1–7.
[20] E. Blancaflor, R. L.-P. J. Pastrana, M. J. C. Sheng, J. R. D. Tamayo, and J. A. M. Umali, "A Security and Vulnerability Assessment on Android Gambling Applications," in *Computer and Communication Engineering*, F. Neri, K.-L. Du, V. Varadarajan, A.-A. San-Blas, and Z. Jiang, Eds., Cham: Springer Nature Switzerland, 2023, pp. 106–115.
[21] S. A. Khan *et al.*, "An Android Applications Vulnerability Analysis Using MobSF," in *2024 International Conference on Engineering & Computing Technologies (ICECT)*, May 2024, pp. 1–7.
[22] D. F. Priambodo, G. S. Ajie, H. A. Rahman, A. C. F. Nugraha, A. Rachmawati, and M. R. Avianti, "Mobile Health Application Security Assesment Based on OWASP Top 10 Mobile Vulnerabilities," in *2022 International Conference on Information Technology Systems and Innovation (ICITSI)*, Nov. 2022, pp. 25–29.
[23] C. Kurniawan and N. Trianto, "Security Assessment pada Aplikasi Mobile Android XYZ dengan Mengacu pada Kerentanan OWASP Mobile Top Ten 2016," *Info Kripto*, vol. 15, no. 1, pp. 11–18, 2021.
[24] C. Gil, L. Baquero, M. Hernández, and J. D. Rodrıguez, "A conceptual exploration for the safe development of mobile devices software based on OWASP," *Int. J. Appl. Eng. Res*, vol. 13, no. 18, pp. 13603–13609, 2018.
[25] M. Elhoseny *et al.*, "Security and Privacy Issues in Medical Internet of Things: Overview, Countermeasures, Challenges and Future Directions," *Sustainability*, vol. 13, no. 21, Art. no. 21, Jan. 2021.
[26] S. Peng, "Law of large numbers and central limit theorem under nonlinear expectations," *Probability, Uncertainty and Quantitative Risk*, vol. 4, no. 1, p. 4, Apr. 2019.
[27] A. J. Bishara and J. B. Hittner, "Confidence intervals for correlations when data are not normal," *Behav Res Methods*, vol. 49, no. 1, pp. 294–309, Feb. 2017.
[28] A. J. Bishara and J. B. Hittner, "Confidence intervals for correlations when data are not normal," *Behav Res*, vol. 49, no. 1, pp. 294–309, Feb. 2017, doi: 10.3758/s13428-016-0702-8.
[29] M. Muhammed Al-Kassab, "The Use of One Sample t-Test in the Real Data," *JOURNAL OF ADVANCES IN MATHEMATICS*, vol. 21, 2022.
[30] S. Liu, R. Liu, and M. Xie, "p-Value as the Strength of Evidence Measured by Confidence Distribution," Jan. 31, 2020, *arXiv*: arXiv:2001.11945.
[31] F. Mateo Tudela, J.-R. Bermejo Higuera, J. Bermejo Higuera, J.-A. Sicilia Montalvo, and M. I. Argyros, "On combining static, dynamic and interactive analysis security testing tools to improve owasp top ten security vulnerability detection in web applications," *Applied Sciences*, vol. 10, no. 24, p. 9119, 2020.
[32] M. Shen, A. A. Pillai, B. A. Yuan, J. C. Davis, and A. Machiry, "Finding 709 Defects in 258 Projects: An Experience Report on Applying CodeQL to Open-Source Embedded Software (Experience Paper) -- Extended Report," Apr. 25, 2025, *arXiv*: arXiv:2310.00205.
[33] J. Li, "Vulnerabilities Mapping based on OWASP-SANS: A Survey for Static Application Security Testing (SAST)," *AETiC*, vol. 4, no. 3, pp. 1–8, Jul. 2020.
[34] S. Chen, Y. Zhang, L. Fan, J. Li, and Y. Liu, "AUSERA: Automated Security Vulnerability Detection for Android Apps," in *Proceedings of the 37th IEEE/ACM International Conference on Automated Software Engineering*, in ASE '22. New York, NY, USA: Association for Computing Machinery, Jan. 2023, pp. 1–5.

[35] J. King, *Android Application Security with OWASP Mobile Top 10 2014*. 2014.
[36] L. O. Silva, E. B. Rodrigues, and I. de S. Santos, "Uma Abordagem para Testes de Segurança em Aplicações Android," in *Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg)*, SBC, Sep. 2023, pp. 225–236.
[37] M. Sakthivel, S. Sivanantham, N. Bharathiraja, N. Bala Krishna, R. Kamalraj, and V. S. Kumar, "Ensuring Web Application Security: An OWASP Driven Development Methodology," in *2024 International Conference on Knowledge Engineering and Communication Systems (ICKECS)*, Apr. 2024, pp. 1–7.
[38] D. a/l Paramasivam, N. L. Ismail, and A. Al-Nahari, "Static Security Analysis of Government and Non-Government Android Mobile Applications in Malaysia: A Comparative Study Using MobSF and OWASP Mobile Top 10," *International Journal of Academic Research in Business and Social Sciences*, vol. 14, no. 10, pp. 2640–2662, Oct. 2024.