JURNAL ILMU KOMPUTER
KHARISMA .TECH

Jurnal KHARISMA Tech | ISSN:1907-2317 | e-ISSN: 2810-0344
https://jurnal.kharisma.ac.id/kharismatech
published by: Pusat Penelitian STMIK KHARISMA Makassar

Volume: 19 no. 02 – September 2024- hlm. 27-35

ANALISIS KEAMANAN APLIKASI REMEMBERME! MENGGUNAKAN METODE VULNERABILITY ASSESSMENT

Oleh:

Rayner Arlyn Chanarly[1], Abdul Munir S[2*], Hendra Surasa[3]
[1,2,3]Teknik Informatika, STMIK KHARISMA Makassar
e-mail: [1]raynerarlyn_20@kharisma.ac.id, [2]abdulmunir@kharisma.ac.id,
[3]hendrasurasa@kharisma.ac.id

Abstrak: *Aplikasi RememberMe! adalah aplikasi mobile password manager yang dirancang untuk mengatasi masalah mengenai password di mana pengguna harus mengingat password yang digunakan untuk setiap situs dan aplikasi. Penelitian ini bertujuan untuk mengukur tingkat keamanan aplikasi RememberMe! untuk mengetahui apakah aplikasi RememberMe! aman untuk digunakan bagi pengguna. Hasil penelitian ini kemudian diharapkan dapat menjadi acuan bagi pengembang dalam melakukan perbaikan dan pengembangan lebih lanjut agar aplikasi RememberMe! lebih aman dan bermanfaat bagi pengguna. Dengan menggunakan metode Vulnerability Assessment penguji dapat memindai aplikasi RememberMe! terhadap semua kasus uji yang mungkin terjadi dan mencari celah yang ada serta titik-titik lemah aplikasi RememberMe!. Proses ini melibatkan analisis statis terhadap aplikasi untuk mengetahui potensi kerentanan, termasuk konfigurasi apliaksi yang buruk atau tidak tepat, kelemahan perangkat lunak dan kelemahan operasional lainnya dalam proses atau penanggulangan teknis. Vulnerability Assessment dapat menguji secara keseluruhan, terintegrasi, operasional, dan tepercaya. Dimana hasil dari pengujian adalah aplikasi RememberMe! mendapatkan App Security Score 46/100 (Medium Risk) dan Grade B, yang berarti aplikasi memiliki tingkat keamanan yang tidak terlalu tinggi, tetapi juga tidak terlalu rendah.*

Kata kunci: *Vulnerability Assessment, RememberMe!, Analisis Statis, Perangkat Lunak.*

Abstract: *RememberMe! application is a mobile password manager application designed to solve problems regarding passwords where users must remember the passwords used for each site and application. This research aims to measure the security level of RememberMe! application to find out whether RememberMe! application is safe to use for users. The results of this research are then expected to be a reference for developers in making improvements and further development so that the RememberMe! application is more secure and useful for users. By using the Vulnerability Assessment method, testers can scan the RememberMe! application against all possible test cases and look for existing gaps and weak points of RememberMe! application. This process involves a Static Analysis of the application for potential vulnerabilities, including poor or improper application configuration, software flaws, and other operational weaknesses in processes or technical countermeasures. Vulnerability Assessment can test overall, integrated, operational, and reliable. Where the result of the test is the RememberMe! application gets an App Security Score of 46/100 (Medium Risk) and Grade B, which means the application has a level of security that is not too high, but also not too low.*

Keywords: *Vulnerability Assessment, RememberMe!, Static Analysis, Software*

[*] Corresponding author : Abdul Munir S (abdulmunir@kharisma.ac.id)

Diterima : *September, 2023* ***Disetujui :*** *Oktober, 2023* ***Dipublikasikan :*** *September, 2024*

1. PENDAHULUAN

Aplikasi RememberMe! adalah aplikasi *mobile password manager* yang dirancang untuk mengatasi masalah mengenai *password* di mana pengguna harus mengingat *password* yang digunakan untuk setiap situs dan aplikasi. RememberMe! memiliki *website landing page* yang dibentuk untuk mempromosikan aplikasi RememberMe! yang berisi mengenai fitur dan tombol unduh yang langsung mengarahkan pengguna ke Google Play Store.

Website RememberMe! dapat diakses melalui website https://byenigma.com/ dan dapat diunduh di Google Play Store melalui website https://play.google.com/store/apps/details?id=com.rememberme2. Masalah utama yang ingin ditangani dalam penelitian ini adalah keamanan aplikasi RememberMe!. Meskipun aplikasi ini dirancang untuk membantu pengguna mengelola kata sandi mereka, kerentanan yang tidak terdeteksi dapat mengancam keamanan data pengguna. Oleh karena itu, dalam penelitian ini penguji menggunakan metode *Vulnerability Assessment* untuk menganalisis keamanan aplikasi RememberMe!. *Vulnerability Assessment* melibatkan penemuan subset ruang input yang dapat digunakan pengguna jahat untuk mengeksploitasi *error* dalam sistem untuk membuatnya menjadi tidak aman [1]. Keseluruhan proses deteksi dan *Vulnerability Assessment* bertujuan untuk memindai sistem target terhadap semua kasus uji yang mungkin terjadi dan mencari celah yang ada serta titik-titik lemah dalam sistem target. Proses ini melibatkan analisis aktif terhadap sistem untuk mengetahui potensi kerentanan, termasuk konfigurasi sistem yang buruk atau tidak tepat, kelemahan perangkat lunak dan kelemahan operasional lainnya dalam proses atau penanggulangan teknis. *Vulnerability Assessment* dapat menguji secara keseluruhan, terintegrasi, operasional, dan tepercaya [1], [2].

Penelitian ini bertujuan untuk mengukur tingkat keamanan aplikasi RememberMe! untuk mengetahui apakah aplikasi RememberMe! aman untuk digunakan bagi pengguna. Hasil penelitian ini kemudian diharapkan dapat menjadi acuan bagi pengembang dalam melakukan perbaikan dan pengembangan lebih lanjut agar aplikasi RememberMe! lebih aman dan bermanfaat bagi pengguna.

2. METODE PENELITIAN

2.1 Studi Literatur

1. Vulnerability Assessment

Kerentanan *(vulnerability)* adalah celah pada perangkat lunak atau perangkat keras atau kesalahan konfigurasi yang dapat dieksploitasi oleh oknum yang tidak bertanggung jawab [3].

Vulnerabilty Assessment adalah proses mendefiniskan, mengidentifikasi, dan mempriotaskan kerentanan dalam sistem komputer, aplikasi, dan infrastruktur jaringan dan memberikan organisasi penilaian dengan pengetahuan, kesadaran, dan latar belakang risiko yang diperlukan untuk memahami ancaman terhadap lingkungannya dan bereaksi dengan tepat [4].

2. Mobile Security Framework (MobSF)

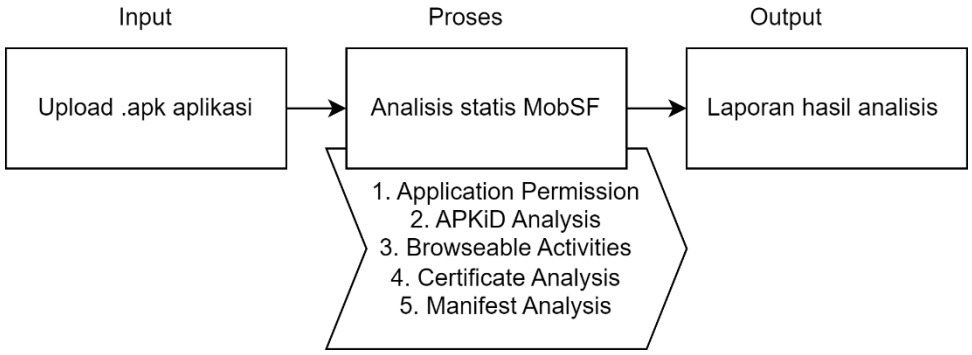
Mobile Security Framework (MobSF) adalah framework yang digunakan untuk pengujian penetreasi terhadap aplikasi seluler *(Android / iOS / Windows)* otomatis yang mampu melakukan analisis statis, dinamis, dan malware. MobSF dapat digunakan untuk analisis keamanan yang efektif dan cepat dari aplikasi seluler Android, iOS dan Windows dan mendukung kedua binari (APK, IPA & APPX) dan kode sumber zip. MobSF dapat melakukan pengujian aplikasi dinamis saat runtime untuk aplikasi Android dan memiliki kemampuan *fuzzing* API Web yang didukung oleh *CapFuzz*, pemindai keamanan khusus Web API [5], [6].

3. Analisis Statis

Teknik pengujian yang dilakukan tanpa menjalankan program. Analisis statis melibatkan pemeriksaan kode sumber atau file biner program secara manual atau dengan menggunakan perangkat lunak analisis statis. Tujuannya adalah untuk mengidentifikasi kerentanan keamanan yang mungkin terdapat dalam kode sumber atau file biner program [7], [8].

2.2. Observasi

Penelitian ini dilakukan dengan metode observasi dengan beberapa tahap. Tahap yang dilakukan dapat dilihat pada Gambar 1.



Gambar 1. Tahap Penelitian [10]

1. Upload .apk Aplikasi

Pada tahap ini, apk aplikasi RememerMe! akan diupload ke program *Mobile Security Framework (MobSF)* untuk dilakukan pengujian.

2. Analisis Statis Mobile Security Framework (MobSF)

Pada tahap ini, *Mobile Security Framework (MobSF)* akan melakukan analisis statis pada .apk aplikasi RememberMe!. Metrik yang diuji pada tahap ini ditunjukkan pada Tabel 1.

Tabel 1 Metrik Pengujian [1], [4], [11]

Metrik	Penjelasan
Application Permission	Analisis terhadap permission dilakukan dengan melihat pada seberapa banyak *dangerous permissions* yang digunakan oleh aplikasi.
APKiD Analysis	*Android Application Identifier (APKiD)* dapat memberikan informasi tentang *compilers, packers, obfuscators* dan hal lainnya yang digunakan aplikasi.
Browseable Activities	Analisis mengidentifikasi dan memitigasi potensi risiko keamanan yang terkait dengan aktivitas yang dapat diakses dari luar aplikasi.
Certificate Analysis	Dilakukan dengan melihat apakah terdapat implementasi algoritma kriptografi yang lemah atau penggunaan algoritma kriptografi yang sudah usang atau sudah dianggap tidak layak.
Manifest Analysis	Analisis ini memeriksa file *AndroidManifest.xml* untuk mengidentifikasi masalah terkait keamanan seperti *permission, component, exported component,* dan *intent.*

3. Laporan Hasil Analisis

Pada tahap ini, program *Mobile Security Framework (MobSF)* akan membuat laporan hasil analisis yang dapat diunduh dalam bentuk pdf.

3. HASIL DAN PEMBAHASAN

3.1 Hasil Analisis

Hasil analisis dari program *Mobile Security Framework (MobSF)* dapat dilihat pada Gambar 2.

File Name:	RememberMe.apk
Package Name:	com.rememberme2
Scan Date:	Aug. 9, 2023, 7:07 a.m.
App Security Score:	46/100 (MEDIUM RISK)
Grade:	B

APPLICATION PERMISSIONS

PERMISSION	STATUS	INFO	DESCRIPTION
android.permission.INTERNET	normal	full Internet access	Allows an application to create network sockets.
android.permission.ACCESS_NETWORK_STATE	normal	view network status	Allows an application to view the status of all networks.

APKID ANALYSIS

FILE	DETAILS
classes2.dex	FINDINGS: Compiler — DETAILS: dx
classes4.dex	FINDINGS: Compiler — DETAILS: r8 without marker (suspicious)
classes3.dex	FINDINGS: Compiler — DETAILS: r8 without marker (suspicious)

FILE	DETAILS
classes6.dex	FINDINGS: Compiler — DETAILS: r8 without marker (suspicious)
classes5.dex	FINDINGS: Compiler — DETAILS: r8 without marker (suspicious)
classes.dex	FINDINGS: Anti-VM Code — DETAILS: Build.FINGERPRINT check Build.MODEL check Build.MANUFACTURER check Build.BRAND check FINDINGS: Compiler — DETAILS: r8 without marker (suspicious)

BROWSABLE ACTIVITIES

ACTIVITY	INTENT
com.google.firebase.auth.internal.GenericIdpActivity	Schemes: genericidp://, Hosts: firebase.auth, Paths: /,
com.google.firebase.auth.internal.RecaptchaActivity	Schemes: recaptcha://, Hosts: firebase.auth, Paths: /,

CERTIFICATE ANALYSIS

HIGH: 1 | WARNING: 2 | INFO: 1

TITLE	SEVERITY	DESCRIPTION
Signed Application	info	Application is signed with a code signing certificate
Application vulnerable to Janus Vulnerability	warning	Application is signed with v1 signature scheme, making it vulnerable to Janus vulnerability on Android 5.0-8.0, if signed only with v1 signature scheme. Applications running on Android 5.0-7.0 signed with v1, and v2/v3 scheme is also vulnerable.
Application signed with debug certificate	high	Application signed with a debug certificate. Production application must not be shipped with a debug certificate.

TITLE	SEVERITY	DESCRIPTION
Certificate algorithm might be vulnerable to hash collision	warning	Application is signed with SHA1withRSA. SHA1 hash algorithm is known to have collision issues. The manifest file indicates SHA256withRSA is in use.

MANIFEST ANALYSIS

HIGH: 1 | WARNING: 2 | INFO: 0 | SUPPRESSED: 0

NO	ISSUE	SEVERITY	DESCRIPTION
1	App can be installed on a vulnerable Android version [minSdk=23]	warning	This application can be installed on an older version of android that has multiple unfixed vulnerabilities. Support an Android version > 8, API 26 to receive reasonable security updates.
2	Debug Enabled For App [android:debuggable=true]	high	Debugging was enabled on the app which makes it easier for reverse engineers to hook a debugger to it. This allows dumping a stack trace and accessing debugging helper classes.
3	Application Data can be Backed up [android:allowBackup=true]	warning	This flag allows anyone to backup your application data via adb. It allows users who have enabled USB debugging to copy application data off of the device.

Gambar 2. Hasil Analisis

Dari hasil analisis diperoleh 17 hasil analisis di mana 10 hasil analisis dengan tingkat keparahan *normal,* 5 hasil analisis dengan tingkat keparahan *medium* dan 2 hasil analisis dengan tingkat keparahan *high*

3.2 Pembahasan

Berdasarkan hasil analisis diatas, berikut penjelasan dari hasil analisis:

1) android.permission.INTERNET

Permission ini berarti bahwa aplikasi memiliki izin untuk terhubung dengan internet.

2) android.permission.ACCESS_NETWORK_STATE

Permission ini berarti bahwa aplikasi memiliki izin untuk mengakses informasi tentang status jaringan pada perangkat seperti apakah perangkat terhubung ke internet melalui *Wi-Fi* atau jaringan seluler.

3) Anti-VM Code

Anti-VM merupakan teknik *anti virtual machine* yang sering digunakan oleh penulis malware untuk menggagalkan upaya identifikasi atau analisis. Dengan teknik ini *malware* akan mencoba untuk mendeteksi apakah malware sedang dijalankan dalam *virtual machine* atau tidak. Jika *malware* mendeteksi *virtual machine*, malware akan bertindak berbeda atau tidak berjalan sama sekali. [9]

4) Application vulnerable to Janus Vulnerability

Janus Vulnerability di Android memungkinkan penyerang untuk memodifikasi kode dalam aplikasi tanpa mempengaruhi *signature* aplikasi.

5) Application signed with debug certificate

Aplikasi yang *disigned with debug certificate* tidak aman karena aplikasi bisa saja tidak diuji secara ketat, dapat mengakses perangkat pengguna dan mengandung *malware*.

6) Certificate algorithm might be vulnerable to hash collision

Hash collision dapat dimanfaatkan oleh penyerang untuk menciptakan sertifikat palsu yang memiliki hash yang sama dengan sertifikat yang sah. Ini bisa digunakan untuk melakukan serangan seperti "*man-in-the-middle*" di mana penyerang dapat menyusup ke dalam komunikasi antara dua pihak yang sah.

7) App can be installed on a vulnerable Android version [midSdk=23]

Aplikasi dapat diinstal pada versi *Android* yang rentan dengan masalah keamanan yang mungkin ada pada versi *Android* dengan *midSdk (Minimum SDK Version)* 23. Hal ini bisa meningkatkan risiko keamanan jika perangkat menggunakan versi *Android* yang lebih lama, karena versi lama mungkin memiliki kerentanan yang tidak diperbaiki.

8) Debug Enabled For App [android:debuggable=true]

Pengaturan "*debuggable*" yang aktif seharusnya hanya digunakan selama tahap pengembangan dan pengujian, bukan dalam rilis aplikasi yang ditujukan untuk pengguna akhir. Pengaturan *android:debuggable=true* memungkinkan penyerang untuk men-*debug* aplikasi, memudahkan penyerang mendapatkan akses ke bagian aplikasi yang seharusnya dirahasiakan [6].

9) Application Data can be Backed up [android:allowBackup=true]

Pengaturan *"android:allowBackup=true"* dianggap tidak aman karena dapat menyebabkan potensi risiko keamanan dan privasi, seperti pencurian data sensitif, akses oleh aplikasi lain, atau kerentanan terhadap pencurian data.

4. KESIMPULAN

Kesimpulan dari analisis statis yang dilakukan terhadap aplikasi RememberMe! adalah sebagai berikut:

1) Aplikasi memiliki izin ke internet dan dapat melihat status jaringan. Hal ini dianggap aman.
2) Terdapat file yang mengandung *Anti-VM Code* yang dapat menimbulkan potensi masalah keamanan dan perlu ditindaklanjuti.
3) Aplikasi memiliki aktivitas yang dapat dibuka melalui skema tertentu. Namun, ini perlu diperiksa secara cermat untuk memastikan tidak ada celah keamanan terkait aktivitas ini.
4) Aplikasi di*signed* dengan *signed certificate*. Namun, terdapat beberapa peringatan, yaitu aplikasi di*signed* dengan *debug certificate*, penggunaan algoritma hash SHA-1 yang rentan *collision hash*, dan potensi *Janus Vulnerability* pada versi Android tertentu.
5) Analisis manifest menunjukkan beberapa peringatan terkait kompatibilitas dengan versi Android yang lebih lama dan pengaturan yang memungkinkan pencadangan data aplikasi.

DAFTAR PUSTAKA

[1] S. Sparks, S. Embleton, R. Cunningham, and C. Zou, “Automated vulnerability analysis: Leveraging control flow for evolutionary input crafting,” in *Proceedings - Annual Computer Security Applications Conference, ACSAC*, 2007, pp. 477–486. doi: 10.1109/ACSAC.2007.27.

[2] S. Shah and B. M. Mehtre, “An overview of vulnerability assessment and penetration testing techniques,” *Journal of Computer Virology and Hacking Techniques*, vol. 11, no. 1, pp. 27–49, Feb. 2015, doi: 10.1007/s11416-014-0231-x.

[3] B. , S. L. , C. Z. Liu, “2010 Eighth Annual International Conference on Privacy, Security and Trust.,” 2010.

[4] J. Pendidikan and D. Konseling, “Analisis Keamanan Website Universitas Singaperbangsa Karawang Menggunakan Metode Vulnerability Assessment.”

[5] C. Hanifurohman and D. Durbin Hutagalung, “ANALISIS STATIS MENGGUNAKAN MOBILE SECURITY FRAMEWORK UNTUK PENGUJIAN KEAMANAN APLIKASI MOBILE E-COMMERCE BERBASIS ANDROID.”

[6] S. Sachdeva, R. Jolivot, and W. Choensawat, “Android Malware Classification based on Mobile Security Framework.”

[7] A. S. S. J. I. Ismail. Aan Kartono, “MEMBANGUN SISTEM PENGUJIAN KEAMANAN APLIKASI ANDROID MENGGUNAKAN MOBSF”.

[8] Erbeliza S, “Analisis Keamanan Aplikasi Mobile Commerce Menggunakan Mobile Security Framework (Mobsf) dan Owasp Mobile Application Security Testing Guide (Mastg)”.

[9] K. Dari and P. Sebelumnya, “BAB 2 TINJAUAN PUSTAKA.”

[10] E. Tansen and D. Wahyu Nurdiarto, “ANALISIS DAN DETEKSI MALWARE DENGAN METODE HYBRID ANALYSIS MENGGUNAKAN FRAMEWORK MOBSF,” *Jurnal Teknologi Informasi*, vol. 4, no. 2, 2020.

[11] F. Sirait, M. Sofyan, and K. Putra, “Implementasi Metode Vulnerability Dan Hardening Pada Sistem Keamanan Jaringan,” *Januari*, vol. 9, no. 1, p. 16, 2018.