время создания мнемосхемы, а также уменьшить вероятность ошибки при создании МС за счет уменьшения количества действий, которые пользователь производит вручную, переложив определенный круг задач на автоматический программный обработчик.

1. *В.Д. Самойлов, С. Д. Винничук, Р.П. Абрамович*. Метод подъема токов нагрузок к узлу ввода для расчета энергетических распределительных сетей // Электронное моделирование. 2015 – т. 37 - № 6
2.Мнемосхема [Электронный ресурс] //Бесплатная интернет энциклопедия [сайт] URL: http: ru.wikipedia.org/wiki/Мнемосхема
3. *В.Д. Самойлов, С. Д. Винничук, Р.П. Абрамович*. Определение токовых ребер графов коммутационных структур на основе анализа фундаментальной системы циклов // Электронное моделирование. – т. 36 - № 4, С. 89-99
4.Стандарт BPMN2.0 [Электронный ресурс]//Бесплатная интернет энциклопедия [сайт] URL: http: ru.wikipedia.org/wiki/BPM
5. AdobeFlashPro [Электронный ресурс] //Официальный сайт Adobe [сайт] URL: http: www.adobe.com/ru/

Поступила 15.02.2018р.

УДК 004(9+056.53)

М.В. Антонішин, О.І. Міснік, В.В. Цуркан, Київ

ОЦІНЮВАННЯ СТАНУ ЗАХИЩЕНОСТІ ПРОГРАМНИХ ЗАСТОСУНКІВ ОПЕРАЦІЙНОЇ СИСТЕМИ ANDROID ЗА МЕТОДОЛОГІЄЮ OWASP MOBILE TOP 10

Abstract. This article discusses the methodology of OWASP Mobile TOP 10 for analyzing the vulnerability of mobile software applications and it demonstrates the process of analyzing using the test tools.

Вступ

Нині програмні застосунки для операційної системи Android набули великої популярності. Тому захисту персональних та критичних даних, наприклад, даних автентифікації, які можуть бути викрадені зловмисниками приділяється багато уваги [2, 8]. Як наслідок, під час розроблення мобільних програмних застосунків розробникам необхідно враховувати ці аспекти. На практиці найбільшої популярності набула методологія OWASP Mobile TOP 10. Нею описується десять загальних критерії оцінювання захищеності. З огляду на це, в статті розглянуто десять критеріїв оцінювання та практично продемонстровано можливості інструментального програмного забезпечення

оцінювання стану захищеності мобільних програмних застосунків операційної системи Android.

Аналіз останніх досліджень і публікацій

В роботах [1 – 5] обґрунтовано актуальність проведення аналізу на уразливості мобільних програмних застосунків. Найбільш розповсюдженою та найбільш зручною є методологія OWASP Mobile TOP 10. Але під час вивчення даної літератури не було проведено повної практичної демонстрації процесу використання методології OWASP Mobile TOP 10.

Мета

Провести практичне використання методології OWASP Mobile TOP 10 для оцінки стану захищеності мобільних програмних застосунків операційної системи Android.

Основна частина

На даний часOWASP Mobile TOP 10 – це одна з основних методологій оцінювання захищеності мобільних програмних застосунків. Нею описується десять загальних критеріїв, що зведені у табл. 1 [2, 6, 7, 11].

Таблиця 1

Критерії оцінки стану захищеності

№	Назва критерію
M1	Обходження архітектурних обмежень (*англ. ImproperPlatformUsage*)
M2	Небезпечне зберігання даних (*англ. InsecureDataStorage*)
M3	Небезпечна передача даних (*англ. InsecureCommunication*)
M4	Небезпечна автентифікація (*англ. InsecureAuthentication*)
M5	Слабка крипостійкість (*англ. InsufficientCryptography*)
M6	Небезпечна авторизація (*англ. InsecureAuthorization*)
M7	Контроль вмісту клієнтських застосунків (*англ. ClientCodeQuality*)
M8	Модифікація даних (*англ. CodeTampering*)
M9	Аналіз вихідного коду (англ.ReverseEngineering)
M10	Скритий функціонал (*англ. ExtraneousFunctionality*)

Для оцінювання стану захищеності застосовувалось спеціалізоване програмне забезпечення: Apktool,adb,dex2jar, Drozer, VCGscanner, JD-GUI, Genymotion, Pidcat [6 – 10]. Для проведення дослідження використовувався

програмний застосунок «PasswordManager-1.3-release.apk».

Для оцінювання стану захищеності програмного застосунку «PasswordManager-1.3-release.apk» спочатку проведено його декомпілювання за допомогою Apktool (див. рис. 1). Декомпілювання не дозволяє отримати вихідний код у зрозумілій формі, однак надається повний доступ до інших ресурсів програмного застосунку за якими можна отримати дані про його архітектуру.

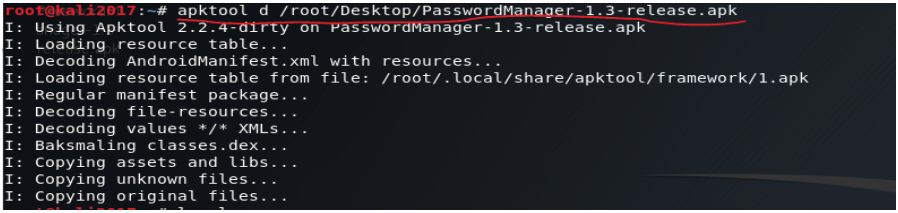

Рис. 1. Декомпілювання «PasswordManager-1.3-release.apk»

Програмний застосунок «PasswordManager-1.3-release.apk» має наступну структуру:

– файл AndroidManifest.xml – описує дозволи, компоненти, версію SDK, яку рекомендується використовувати та інші налаштування застосунку;

– файл apktool.yml – містить службову інформацію, яка необхідна Apktool для повторного компілювання;

– директорія lib – містить бібліотеки, які додатково завантажені у програмний застосунок розробником. У даному випадку, використовується бібліотека з розширенням *.so;

– директорія original – містить службові файли програмного застосунку;

– директорія res – містить файлові, графічні та інші ресурси програмного застосунку;

– директорія smali – містить файли вихідного коду у вигляді байт-коду.

Для отримання вихідного коду застосунку у читабельній формі використовується інструмент dex2jar (див. рис. 2). Це дозволить проаналізувати вихідний код програмою VCG-scanner або виконати це вручну.

Рис. 2. Декомпілювання «PasswordManager-1.3-release.apk» за допомогою dex2jar

Далі охарактеризуємо програмний застосунок відповідно до OWASP Mobile TOP 10.

Небезпечне зберігання даних (англ. InsecureDataStorage)

В програмному застосунку, що досліджується, частина службової інформації залишилась загальнодоступною. В цьому випадку, сторонні програмні застосунки, які мають права READ_LOGS (наприклад logcat або pidcat) можуть отримати доступ до критичної інформації, тим самим порушуючи її конфіденційність.

Під час тестування використовувалась програма pidcat. Уразливість була виявлена у програмному коді (див. рис. 3) – розробник залишив не видаленою функцію Log.d(), яка використовується для налаштування роботи вихідного коду [6 – 8].

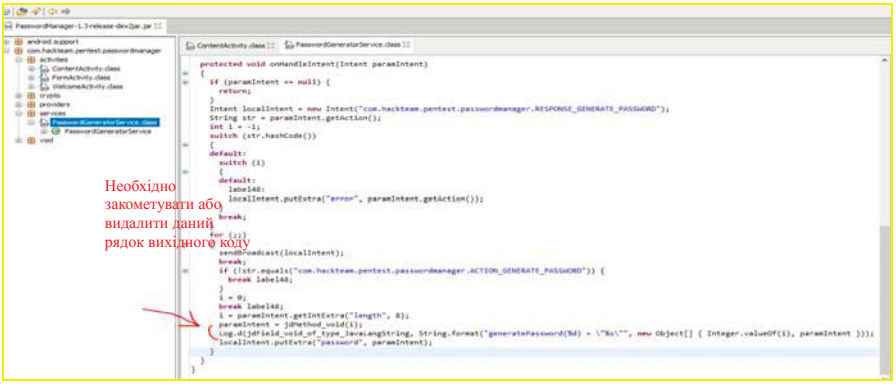

Рис. 3. Приклад відображення вразливості програмного застосунку

Під час аналізу програмного застосунку фреймворком Drozer, було виявлено експортовані компоненти ContentProvider (див. рис. 4), які дозволяють переглянути URI програмного за стосунку та отримати доступ до локальної бази даних, яку використовує програмний застосунок, що досліджується [9].

За допомогою модулю app.provider.query отримано доступ до локальної бази даних програмного застосунку.

Рис. 4. Експортовані URI-ContentProvider та приклади верифікації уразливості

Аналізуючи програмний код знайдено вставку у локальну базу даних програмного за стосунку (див. рис. 5). Використання цих даних дозволяє отримати несанкціонований доступ до системи. Також було знайдено вставку в базу даних логіну та паролю одного з користувачів.

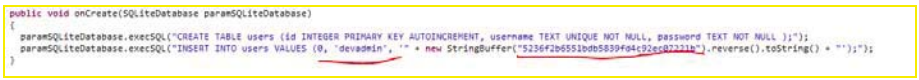

Рис. 5. Зберігання даних у відкритому вигляді у програмному коді

Небезпечна передача даних (англ. InsecureCommunication)

У програмі реалізоване автоматичне переключання з протоколу HTTPS на HTTP. Якщо останній не підтримує шифрування, що може призвести до передавання інформації відкритим каналом передачі.

Слабка криптостійкість (англ. InsufficientCryptography)

В результаті ручного аналізу вихідного коду мобільного програмного застосунку розглянуто клас FastCrypto.java, який здійснює перетворення масиву байт з даними у хеш-суму алгоритмом MD5 (див. рис. 6). Алгоритм MD5 на час тестування визнано не надійним. Його хеш-суму можливо підібрати, як за допомогою онлайн-ресурсів, та і за допомогою програмних інструментів. [13]

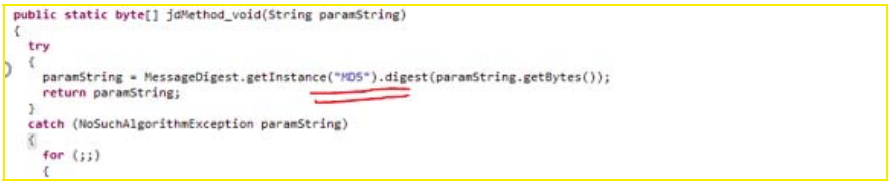

Рис. 6. Підключення бібліотеки та визначення алгоритму хешування

Паролі у локальній базі даних мобільного програмного застосунку зберігаються у хеш-формі (див. рис. 4). За допомогою Drozer отримано доступ до бази даних та знайдено паролі. Після цього, використавши онлайн сервіс MD5 Decrypter, підібрано пароль (див. рис. 7).

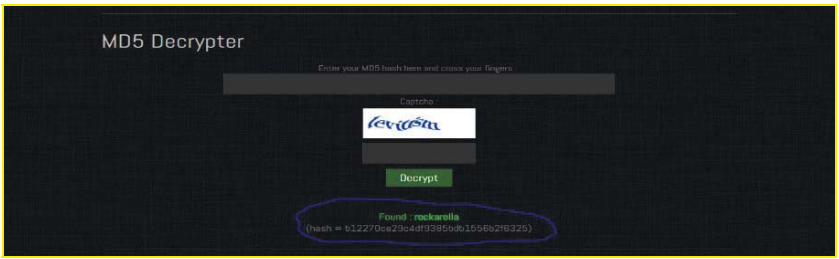

Рис. 7. Результат підбору хеш-суми

Після декомпіляції коду його було просканово VCG сканером та виявлено використання уразливої бібліотеки (див. рис. 8), яка використовується для генерації блоків ключа. Водночас при використанні бібліотеки java.util.Random є можливість знайти наступне випадкове значення. Тому рекомендується Oracle використовувати бібліотеку java.security.SecureRandom [12].

Рис. 8. Критична уразливість вихідного коду

Контроль вмісту клієнтських застосунків (англ. ClientCodeQuality)

За допомогою VCG сканера було виявлено відсутність контролю за введенням імені файлу, контролю за помилками, а саме блоків try/catch, а також об'єктів Intent. Це може привести до помилки виконання застосунку, завантаження та запуск виконуваного файл.

Рекомендації:

– використання блоків контролю за виключенням try/catch;
– контроль за вхідними параметрами та іменами файлів;
– верифікація об'єктів Intent під час прийому.

Модифікація даних (англ. CodeTampering)

За допомогою фреймворку Drozer встановлено наявність SQL injection (див рис. 9). Дана уразливість дозволяє модифікувати дані, які збережені у локальній базі даних. Для верифікації здійснено спробу їх модифікування (див. рис. 10), але на запит отримано відповідь – «Notyetimplemented» (див. рис. 10). Це означає, що запит на модифікування даних не реалізовано у програмі (див. рис. 11), тому, за допомогою фреймворку Drozer, на час проведення тестування, не має можливості модифікувати дані [9].

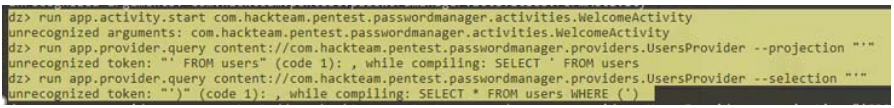

Рис. 9. Перевірка наявності уразливості бази даних

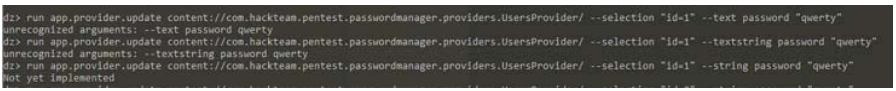

Рис. 10. Верифікація знайденої уразливості

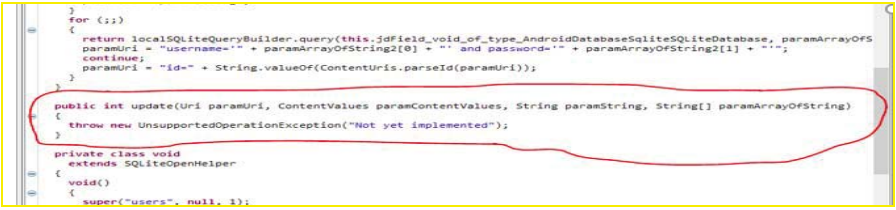

Рис. 11. Уразливе місце модифікації даних

Аналіз вихідного коду (англ. Reverse Engineering)

Вихідний код програмного застосунку, що досліджувався, не захищено засобами обфускації. Це дає можливість його аналізувати на наявні уразливості. Для аналізу програмного коду застосунок декомпільовано за допомогою apktoolta dex2jar. Після цього проаналізовано архітектуру та фукціонал, а також проведено його статичне сканування та виявлено потенційні вразливості.

Для захисту вихідного коду необхідно провести його обфускацію, а також передбачити шифрування та використання засобів детектування підробки коду.

Висновок

Таким чином, використання методології OWASP Mobile TOP 10 для оцінювання захищеності мобільних програмних застосунків операційної системи Android дозволяє наочно та у цифрах встановити кількість потенційних уразливостей. Наявність таких уразливостей може призвести до порушення конфіденційності, цілісності та доступності даних програмних застосунків.

При цьому деякі уразливостей можна одночасно віднести до різних категорій. Це ускладнює їх оцінювання та, як наслідок, способи усунення. З огляду на це, проведено наочне демонстрування використання методології OWASP Mobile TOP 10 та зробити висновок, що програмний застосунок не можна вводити у експлуатацію. Відомості стосовно кількості уразливостей наведено у табл. 2.

Таблиця 2

Уразливості

№	Категорія	Кількість уразливостей
M1	Обходження архітектурних обмежень	0
M2	Небезпечне зберігання даних	2
M3	Небезпечна передача даних	1
M4	Небезпечна аутентифікація	0
M5	Слабка крипостійкість	2
M6	Небезпечна авторизація	0
M7	Контроль вмісту клієнтських застосунків	2
M8	Модифікація даних	1
M9	Аналіз вихідного коду	1
M10	Скритий функціонал	0

1. *Sreenivasa Rao Basavala, Narendra Kumar, Alok Agarrwal* Mobile Applications – Vulnerability Assessment. Through the Static and Dynamic Analysis. – Conference on Advances in Communication and Control Systems 2013.
2. Vulnerability Testing: A Security Health Check-Up for Mobile Apps [Електронний ресурс] – Режим доступу: https://www.wired.com/insights/2013/04/vulnerability-testing-a-security-health-check-up-for-mobile-apps/ – Назва з екрану
3. *Alejandro Argudo, Gabriel López, Franklin Sánchez.* Privacy vulnerability analysis for Android Applications: A practical approach (2017). Електронна бібліотека IEEE Xplore Digital Library [Електронний ресурс] – Режим доступу: http://ieeexplore.ieee.org/document/7962545/
4. *Ricky M.,Monique L. Magalhaes* Assessing the Security of Mobile Applications – Part 1. Planning. [Електронний ресурс] – Режим доступу: http://techgenix.com/assessing-security-mobile-applications-part1/ – Назва з екрану
5. *Ricky M.,Monique L. Magalhaes* Assessing the Security of Mobile Applications – Part 2. Testing the application. [Електронний ресурс] – Режим доступу: http://techgenix.com/assessing-security-mobile-applications-part2/ – Назва з екрану
6. Mobile Security Wiki. [Електронний ресурс] – Режим доступу: https://mobilesecuritywiki.com – Назва з екрану
7. DefconRU. Mobile security. [Електронний ресурс] – Режим доступу: https://defcon.ru/category/mobile-security/ – Назва з екрану
8. Information security. Basic to Advanced. Android. [Електронний ресурс] – Режим доступу: https://securitylabexpert.wordpress.com/android/ – Назва з екрану.
9. MWR Labs. Drozer. [Електронний ресурс] – Режим доступу: https://labs.mwrinfosecurity.com/tools/drozer/ - Назва з екрану
10. Appie – Android Pentesting Portable Integrated Environment. [Електронний ресурс] – Режим доступу: https://manifestsecurity.com/appie/ – Назва з екрану
11. OWASP Mobile Security Project [Електронний ресурс] – Режим доступу: https://www.owasp.org/index.php/OWASP_Mobile_Security_Project – Назва з екрану
12. Взлом генератора случайных чисел Java. [Електронний ресурс] – Режим доступу: https://xakep.ru/2015/07/20 /java-random-hack/ – Назва з екрану
13. Все методы взлома MD5. [Електронний ресурс] – Режим доступу: https://xakep.ru/2013/10/13/md5-hack/ – Назва з екрану

Поступила 24.02.2018р.