УДК 004.056
DOI: 10.31673/2409-7292.2022.024552

Гоголь Б.В., Наталенко М.М.

СПОСІБ ПРОТИДІЇ ВРАЗЛИВОСТЯМ ДОДАТКІВ НА БАЗІ ОПЕРАЦІЙНОЇ СИСТЕМИ ANDROID ЧЕРЕЗ УПРАВЛІННЯ ДОЗВОЛАМИ

У статті досліджуються вразливості та способи захисту мобільних додатків на базі операційної системи Android. Досліджено проблему безпеки та проаналізовані різні види загроз безпеки мобільних додатків. На основі досліджень проведених в статті розроблено рекомендації щодо захисту мобільних додатків від кібератак шляхом управління дозволами.

Ключові слова: Android, кібербезпека, мобільні додатки, конфіденційна інформація, персональна інформація, захист, кібератаки, загрози.

Вступ

Величезна популяризація мобільних пристроїв і сучасна тенденція використовувати їх як особистий, так і діловий пристроїв, перетворили смартфони та планшети на природну мішень загроз безпеці. Ця проблема стає особливо актуальною на платформі Android, де кількість поставок пристроїв перевищила 84% частки ринку в першому кварталі 2022 року. Після постійного зростання продажів мобільних пристроїв на базі Android кількість шкідливих програм, націлених на мобільні пристрої також підвищується. Крім того, відкритість екосистеми Android викликає занепокоєння щодо безпеки.

Мета статті – дослідити вразливості операційної системи Android, виробити рекомендації щодо захисту мобільних додатків від кібератак шляхом управління дозволами.

Безпека програмного забезпечення Android

За принципом дизайну Android створений так, щоб дозволити будь-якому розробнику та виробнику робити внесок у платформу. У цьому напрямку Android полегшує налаштування версій ОС, дозволяє розповсюджувати програми з альтернативних магазинів додатків і полегшує рутинг пристроїв. Ця гнучкість супроводжується ціною порушення безпеки. Android забезпечує важливі контрзаходи безпеки для боротьби з мобільними загрозами. Ці заходи включають ізоляцію додатків і запровадження системи дозволів. Невідомі порушення безпеки досліджуються під час можливої атаки, що впливає на властивості безпеки системи Android. Отже, необхідно зрозуміти модель безпеки, яка використовується в стеку програмного забезпечення Android, представленому на рисунку 1.

Рівень додатків — це верхній рівень програмного стеку Android. Користувачі взаємодіють з цим рівнем, щоб отримати доступ до ресурсів пристрою, наприклад, для телефонних дзвінків і відправки/отримання SMS, або для використання встановлених користувачем програм (додатків). Користувачі повинні встановлювати програми за допомогою Google Play Store, або магазинів виробників пристроїв (наприклад, LG SmartWorld). Крім того, Android дозволяє встановлювати програми з інших джерел, таких як неофіційні магазини (наприклад, 1Mobile), Android Debug Bridge (ADB), Sideload або завантаження APK додатка (пакет Android).

Модель безпеки Android заснована на ізоляції програми, що запобігає доступу до інших програм конфіденційної інформації. Ця ізоляція реалізується за допомогою доступу до ресурсів пристрою, керованого системою дозволів. Розробник Android повинен повідомити дозволи програми, а користувач повинен явно авторизувати цей доступ під час встановлення програми. Завдяки системі дозволів ресурси пристрою захищені, і доступ до них можуть отримати лише системні програми, будь то програми, наявні в Android Open Source Project (AOSP), або програми, розроблені оригінальним виробником обладнання (OEM). Однак цей контроль можна обійти, коли пристрій має root-права, процес, за допомогою якого користувацькі програми отримують привілейований контроль над обмеженими ресурсами.

Наприклад, програми, встановлені користувачем на рутованих пристроях, можуть захоплювати Інтернет-пакети, надіслані іншими програмами. Програми Android виконуються в пісочниці, де необхідні механізми міжпроцессного зв’язку (IPC), щоб інші

програми могли отримати доступ до даних програми. Постачальники вмісту керують цим зв'язком, який доступний на рівні Application Framework. На цьому рівні є інтерфейс прикладного програмування (API), що надає набір функцій для використання при розробці додатків. Тут одне важливе джерело вразливостей вводять розробники додатків, нехтуючи правилами безпеки для розробки додатків. Крім того, можна знайти вразливі місця в Android Application Framework, такі як Cross-Signed Certificate. Наступний шар містить власні бібліотеки Android і середовище виконання Android. Нативна бібліотека написана на C/C++ і скомпільована, зокрема, для кожного пристрою. Була виявлена вразливість у бібліотеці OpenSSL, відома як Freak SSL5. Ця вразливість дозволяє розшифровувати та змінювати повідомлення SSL.

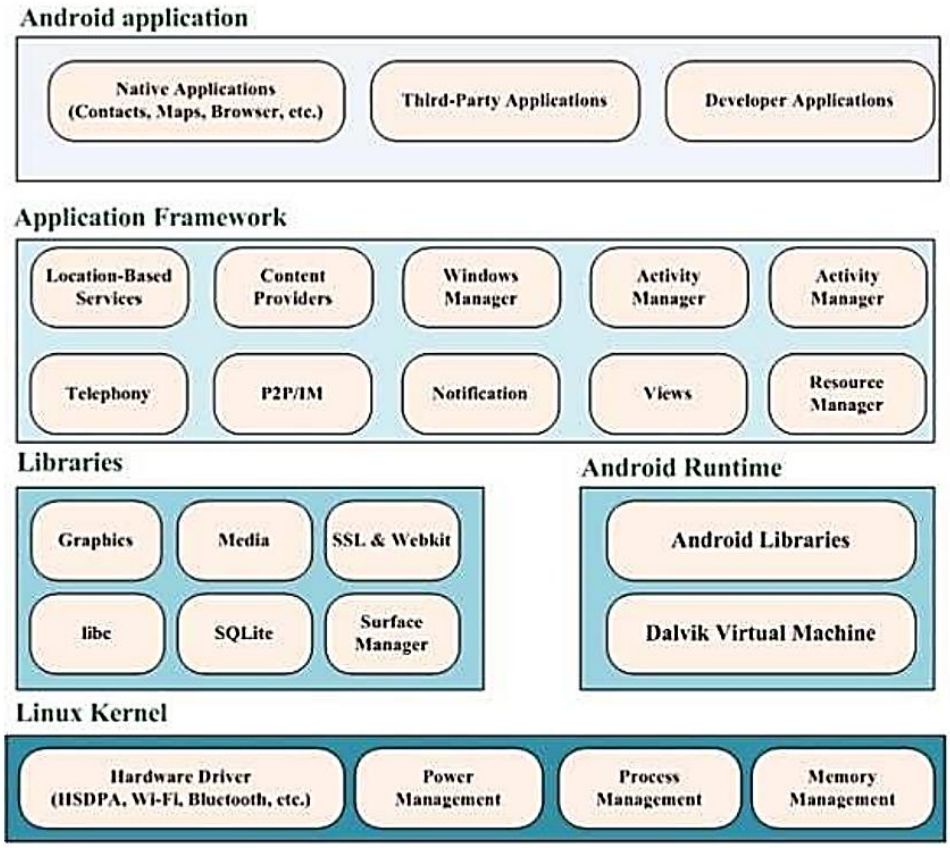

Рис. 1. Стек програмного забезпечення Android

Віртуальна машина Dalvik і основні бібліотеки знаходяться в середовищі виконання Android. Dalvik VM оптимізована для пристроїв з обмеженнями потужності та пам'яті та була повністю замінена новою віртуальною машиною під назвою ART (Android Runtime) в Android Lollipop. Безпека досягається завдяки ізоляції додатків, що виконується за допомогою віртуальних машин, поряд з механізмом контролю доступу Linux. Однак недоліки цієї моделі дозволяють здійснювати атаки з підвищенням привілеїв, минаючи цю модель безпеки.

Основою стеку Android є ядро Android на базі Linux. Це ядро включає апаратні абстракції, які дозволяють взаємодіяти з апаратним забезпеченням пристрою. Він надає такі послуги: керування пам'яттю, керування живленням, драйвери пристроїв, керування процесами, мережа та безпека. Навіть із застосуванням кількох засобів контролю безпеки на цьому рівні є вразливості.

Сценарії атак

Традиційні методи або режими введення шкідливого коду в програми для Android останнім часом втрачають ефективність і швидкість, і їх легко виявити Google Play Store

Protect (GPSP), який базується на додатках машинного навчання та захисту від шкідливих програм. Щоб протидіяти цій реалізації механізму безпеки, хакери знайшли новий метод ін'єкції шкідливого коду в програми за допомогою Intent. У цьому векторі атаки, користувач хакера заражає кілька заражених програм для вилучення цінних даних зі смартфона користувача. Щоб здійснити цю атаку, хакеру потрібні принаймні дві програми, одна з яких містить усі необхідні привілеї та витягує дані з пристрою користувача, друга програма витягує всі дані з першої програми та надсилає їй на віддалений хост або на задану URL-адресу.

Існує кілька способів, за допомогою яких зловмисник може виконати та успішно зламати пристрій користувача, наприклад:

коли ми заходимо в Play Store, ми можемо знайти багато програм, для повної роботи яких потрібна додаткова програма - наприклад плагін або ключ. Зловмисник може ввести деякі дозволи в головну програму і змінити другу програму таким чином, щоб надсилати дані користувача за межами пристрою користувача без відома користувача;

зловмисник може розробляти програми, націлені на такий тип атаки, і завантажувати їх у Play Store. Є велика ймовірність, що ці програми не позначатимуться як шкідливі, оскільки програми скануються й тестуються окремо, а перша програма просто вимагає дозволу, як і багато законних програм. Деякі ігри-програми вимагають багато дозволів для правильного виконання та функціонування, а користувачі не знають і для задоволення від гри витончено надають ті дозволи, які друга програма від зловмисника надсилатиме дані користувача з першої програми на віддалений хост.

Змовні програми

Платформа безпеки Android призначена для захисту даних користувачів, а також критичних системних ресурсів і програм від порушення безпеки та зловживань загалом. Модель захищає програми та ресурси, поєднуючи підписи програм, пісочницю та дозволи. На жаль, цей механізм можна обійти за допомогою змовних програм. Комбінований дозвіл програм включає в себе можливість здійснювати шкідливі атаки, які не можуть бути можливими для окремої програми. Наприклад, розглянемо, де змова включає програми, які надали доступ до особистих даних користувача, і це передає дані другій програмі, якій дозволено передавати дані за межі пристрою на віддалений хост. Однак ОС Android перевіряє, чи програма, яка отримує доступ до захищеного дозволом ресурсу через іншу програму, сама має цей дозвіл.

Модель дозволу Android

Безпека Android залежить від обмеження програм шляхом поєднання підписів додатків, ізольованого програмного середовища та дозволів. Підпис додатків є необхідною умовою для включення на офіційний ринок додатків Android (Google Play Store). Підпис або сертифікат програми є точкою довіри між сторонніми розробниками програм і Google, щоб забезпечити відповідність, цілісність та репутацію розробника та Google. Дозвіл додатка регулює, як програми отримують доступ до певних конфіденційних даних або системних ресурсів на пристрої користувача, таких як служби LOCATION, CAMERA, CONTACTS, WIFI_STATE, GPS тощо.

Програми Android виконуються в середовищі «пісочниці» для захисту системних ресурсів ОС, даних користувача, програми розробника, мережі або постачальника послуг, а також розміщених програм від шкідливого програмного забезпечення. Кожна програма захищена присвоєним унікальним ідентифікатором (UID) ядром Linux в ізольованій пісочниці. Пісочниця не дозволяє іншим додаткам або їхнім системним службам перешкоджати іншим програмам.

IPC/ICC в механізмі зв'язку Android є ключовим механізмом, який дозволяє компоненту однієї програми отримати доступ до даних користувача і може передавати такі дані іншому компоненту в тій самій програмі або іншому компоненту в іншій програмі, іноді на віддалений хост (сервер за межами місцезнаходження та доступу користувача). IPC/ICC допомагає усунути дублювання функцій у різних програмах. Розробники додатків можуть

використовувати дані, ресурси та послуги, що надаються іншими додатками, для власного використання.

Наприклад, програма для бронювання таксі, як-от Uber, може запитувати у Google Maps місцезнаходження клієнтів або водія, ця можливість IPC/ICC може зменшити навантаження на розробників і сприяти повторному використанню функцій. Механізм намірів Android (AIM) дозволяє компонентам програми викликати інші компоненти тієї ж програми або інші програми на смартфоні користувача. Він використовується додатками для передачі інформації між двома або більше компонентами однієї або іншої програми через пакети.

За бажанням, Intent містить призначення, назву компонента, рядок дії, категорію та іншу інформацію, яку розробник вважає придатною. Значення є переважним механізмом маршруту повідомлень для асинхронного IPC в ОС Android. Інтерфейс програмного забезпечення ОС Android (API) визначає класи, які називаються методами IPC/ICC, які можуть отримувати наміри та виконувати дії відповідно. IPC/ICC широко спрощується через значення. Підкреслено, що 2955/33258 додатків використовують IPC/ICC через значення. Використання значення можна розділити на широкі типи залежно від їх призначення викликів IPC/ICC, і вони бувають IMPLICIT та EXPLICIT INTENT.

Додаток може вступити в змову та завдати шкоди смартфону користувача та поставити під загрозу конфіденційність користувача, використовуючи ресурси та отримувати доступ до даних, які явно не надані цим програмам. Відкрите спілкування використовується для явного значення, таким чином, прямо запитує системний ресурс або дані користувача за допомогою програми на смартфоні користувача або передає в інші програми, встановлені на пристрої. Значення відіграє важливу роль у таких видах комунікації. Значення представляє абстрактний опис операції, яка має бути виконана, яка може включати додаткові дані, необхідні для виконання набору інструкцій для виконання завдання або дії. Явне значення – це значення, яке визначає, яку програму та компонент програми слід запустити. Наприклад, App A створює і значення, де безпосередньо вказує на запуск конкретного модуля додатка B, цей процес може бути досягнутий, як показано на рисунку 2.

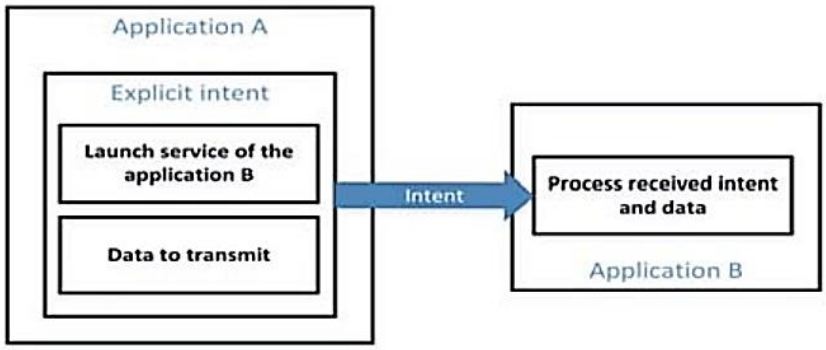

Рис. 2. Приклад явного значення для додатку

Impicit Intent — це той, який не визначає програму та модуль передбачуваної програми, яку потрібно запустити. Цей тип значення визначає лише дію, яку потрібно виконати, без даних, які точно визначають, яку програму може виконати або виконати її дію. Наприклад, на пристрій з магазину додатків можна встановлювати різноманітні смартфони. Коли користувач набирає номер телефону, щоб зателефонувати, для обробки запиту користувача можна вибрати один із встановлених набірників.

Додаток, у якому користувач набирає номер телефону, може створити неявне значення та отримати цей запит. Це значення визначає лише дію «Intent.ACTION DIAL». ОС Android пересилає це значення до програми, яка може обробити цю дію. Якщо є більше однієї програми, яка може обробити запит, користувачеві буде представлено засіб вибору

(спливаюче меню), щоб вибрати одну із запропонованих програм. Значення може бути стандартним (попередньо визначеним) або налаштованим (створеним розробником), приклад IPC/ICC, який використовує неявне значення, як показано на рисунку 3.

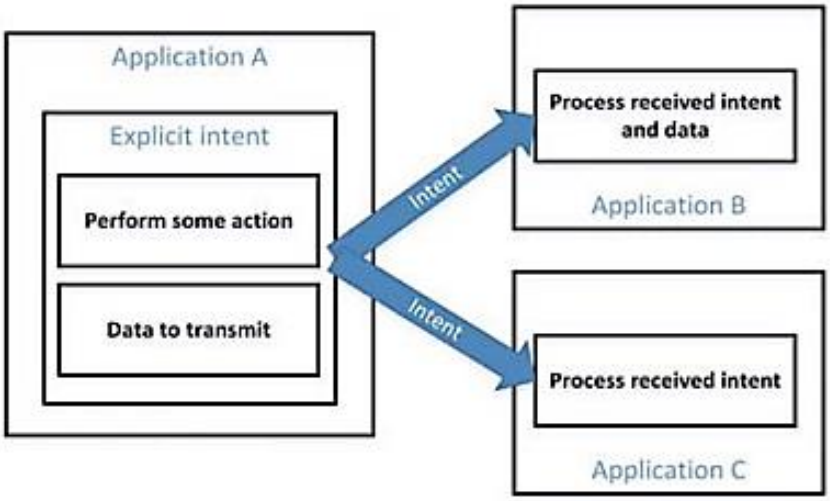

Рис. 3. Приклад неявного значення для додатку

Механізм Intent на Android

Значення дозволяє компонентам програми викликати інші компоненти тієї ж програми чи інших програм. Воно використовується програмами для передачі даних між двома або більше компонентами зразка програми або іншими програмами через пакети. За бажанням, значення містить назву цільового компонента або рядок дії, категорію та дані. Вони є переважним механізмом маршруту повідомлень для асинхронного IPC в ОС Android. Системний API ОС Android визначає класи, які називаються класами IPC/ICC, які можуть отримувати значення і виконувати дії відповідно.

iBAP: захист від атак на основі Intent

Ринки додатків Android містять широкий спектр програм сторонніх розробників, і користувачі можуть встановлювати програми з різними рівнями довіри. Користувачі встановлюють програми від відомих і невідомих розробників з багаторічною репутацією на цих ринках, а ті розробники, чия адреса та ім'я навіть не існують, деякі з цих програм для встановлення можуть обробляти особисті та приватні дані користувачів, такі як фотографії, відео. Наприклад, користувач вирішить встановити програму для здоров'я або медичну програму, яка є надійною програмою, а також деякі безкоштовні ігри. Ці безкоштовні ігрові програми не повинні мати можливість за звичайних обставин знову отримати доступ до медичних даних користувачів, створених і збережених за допомогою програми охорони здоров'я або медичного обслуговування.

Згідно з моделлю та структурою безпеки Android, усі програми вважаються потенційно шкідливими. Кожна програма виконується у своєму власному процесі з низьким рівнем привілеїв (UID), і програми за замовчуванням мають доступ лише до власних файлів. Цей рівень політики ізоляції має на меті захистити програми з конфіденційними даними користувача від зловмисного або зловмисного програмного забезпечення.

Незважаючи на всю свою політику ізоляції за замовчуванням, програми можуть за бажанням спілкуватися один з одним за допомогою механізму передачі повідомлень, такого як Intent. Зв'язок через цей засіб може стати вектором атаки, якщо розробник навмисно чи ненавмисно розкриває методи, класи та функціональні можливості, тоді додаток може бути обдуреним, щоб виконати небажане завдання або дію, щоб поставити під загрозу безпеку смартфона користувача. На рисунку 4 представлена концепція моделі загрози викрадення компонентів в ОС Android за допомогою програми Intent як засобу зв'язку між двома

програмами. Супротивником є програма абонента, а жертва — програма абонента, яка містить компоненти, які експортуються. Зловмисний компонент зловмисника в абоненті обманув абонента за допомогою створеного запиту IPC/ICC до експортованого компонента, щоб зловмисно ініціювати виконання його коду для дії привілеїв.

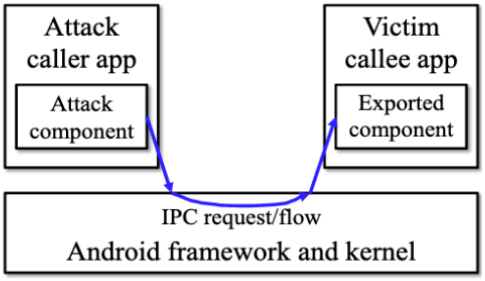

Рис. 4. Приклад моделі загрози викрадення компонентів

iBAP розроблено, щоб зосередитися на перспективі відправників Intent і одержувачів Intent. Акцент робиться на відправленні Intent не в той компонент програми, помилка є фатальною і може призвести до витоку даних користувача, цей підхід iBAP також зосереджується на зовнішніх Intent, що надходять від інших програм, якщо компонент випадково опублікований (exported=true), то зовнішні програми можуть дивовижним чином викликати цей компонент віддалено та впровадити в нього шкідливий код. iBAP сильно покладається на інструмент FlowDroid і функції для наступних кроків, як показано на рисунку 5.

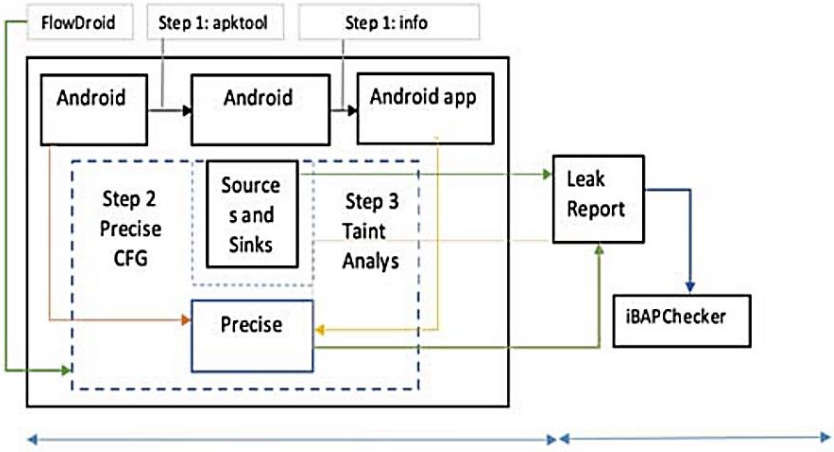

Рис. 5. iBAP та iBAP checker

Перші три кроки описують iBAP. На першому кроці iBAP витягує список доступних компонентів програми (діяльність, послуга, трансляція тощо). На другому кроці iBAP створює точний графік потоку керування (CFG) з виведенням даних першого кроку. На третьому кроці iBAP використовує збірку методів джерела та прийому, розраховану Susi, щоб виконати аналіз забруднень на точному CFG, отриманому на другому етапі, а потім створити список потенційного захоплення намірів і потенційних витоків конфіденційних даних користувача, які він виявив.

Оцінка моделі iBAP

iBAP сильно залежить від FlowDroid, деякі програми не були належним чином проаналізовані через зависання та надмірне споживання пам'яті, це сталося в результаті не точного налаштування FlowDroid спеціально для цього проекту та певної модифікації бінарних файлів рівня додатків архітектурі ОС Android, однак, деякі результати були

отримані для підтвердження концепції. Результат iBAP був поставлений на десятикратну перехресну перевірку, яка передбачає приблизно 95,6% точність здатності моделі iBAP відстежувати витоки в програмах через Intents, використовуючи дані тесту та навчений набір даних, як показано на рисунку 6.

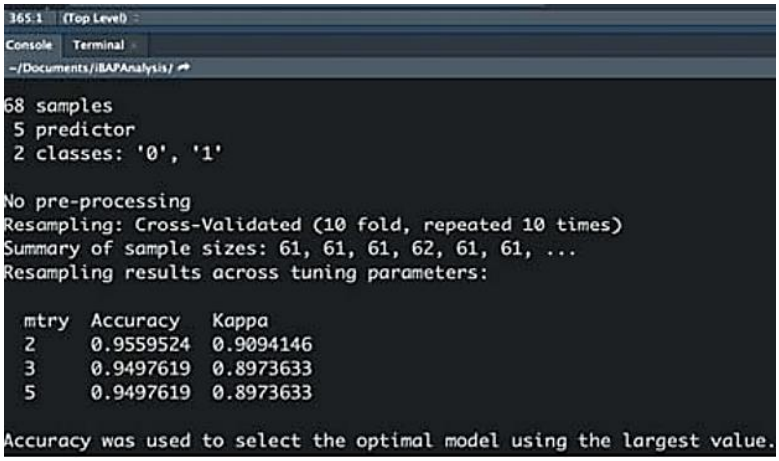

Рис. 6. Результат перевірки моделі iBAP за допомогою Ten Fold Cross-Validation

Обмеження iBAP

Наразі iBAP не обробляє уніфікований ідентифікатор ресурсу (URI), який добре використовується ContentProvider в ОС Android. iBAP наразі не захищає від декількох потоків, відображень та умов.

Висновки

У цій статті iBAP представлений як альтернатива, інструмент для використання потенційних витоків компонентів програми через механізм абстракції Intent, а iBAPChecker — інструмент для перевірки результатів звіту iBAP з аналізу плям. iBAP спочатку створює точний CFG для аналізованих додатків. Потім він виконує статичне забруднення за допомогою добре відомого набору методів джерела та поглинача для виявлення потенційних витоків компонентів. Однією з фундаментальних проблем є те, що Intent використовуються як для внутрішньокомпонентних, так і для міжкомпонентних, і їх використання в програмі може піддати програму непередбачуваній зовнішній атаці, якщо розробник не буде особливо пильним.

Перелік посилань

1. Montealegre C. Security vulnerabilities in android applications [Електронний ресурс] / Crischell Montealegre // Edith Cowan University. – 2018.

2. Laud Boateng F. Vulnerabilities in Android Apps Permissions A critical look at the implicit and explicit intent mechanism for communication between app and their components and its associated security challenges. [Електронний ресурс] / Frank Laud Boateng // International Journal of Computer Science and Information Security (IJCSIS). – 2019.

3. Andre Batista de Carvalho C. Neutralizing vulnerabilities in Android: a process and an experience report [Електронний ресурс] / Carlos Andre Batista de Carvalho // International Journal of Computer Science and Information Security (IJCSIS). – 2016.

4. 8 Best Free Hacking Books for Ethical Hackers [Електронний ресурс] // securedyou. – 2022. – Режим доступу до ресурсу: https://www.securedyou.com/8-best-hacking-books-pdf-free-download-for-ethical-hackers/.

5. Pardo Lopez L. Open Source Hacking Tools [Електронний ресурс] / Luis Alejandro Pardo Lopez – Режим доступу до ресурсу: https://pdfcoffee.com/open-source-hacking-toolspdf-pdf-free.html.

6. Top 7 Vulnerabilities In Android Applications 2022 [Електронний ресурс] // CODERSERA. – 2022. – Режим доступу до ресурсу: https://codersera.com/blog/top-7-vulnerabilities-in-android-applications-2019/.

7. Android apps with millions of downloads exposed to high-severity vulnerabilities [Електронний ресурс] // Mircosoft. – 2022. – Режим доступу до ресурсу: https://www.microsoft.com/security/blog/2022/05/27/android-apps-with-millions-of-downloads-exposed-to-high-severity-vulnerabilities/.

Надійшла: 11.06.2022 Рецензент: д.т.н., професор Вишнівський В.В.