ПІДВИЩЕННЯ СТІЙКОСТІ МОБІЛЬНИХ ДОДАТКІВ НА ПЛАТФОРМІ ANDROID ДО ДИНАМІЧНОГО АНАЛІЗУ

Іванов Є.В., Сидоренко З.М.
Харківський національний університет радіоелектроніки, Харків, Україна

У світі налічується понад 3 мільярдів користувачів Android, і більшість з них зберігають свої конфіденційні дані на мобільних пристроях, що робить їх привабливою мішенню для зловмисних хакерів. З огляду на це, розробники повинні надавати пріоритет безпеці своїх додатків на Android, особливо коли додаток повинен захищати бізнес-активи або стримувати зловживання на стороні клієнта. Підвищення стійкості до динамічного аналізу може допомогти зменшити такі ризики, як: крадіжка або компрометація запатентованих алгоритмів, комерційних таємниць, даних клієнтів, моделей штучного інтелекту або машинного навчання; шахрайство, фінансових додатках [1-3].

Метою доповіді є аналіз загроз динамічного аналізу для мобільних додатків реалізації механізмів захисту у вигляді нативної бібліотеки для Android.

Використання нативної бібліотеки допомагає зробити більш стійкі алгоритми виявлення фальсифікації додатку, та його запуску у небезпечній середі. Одним з найбільш відомих методів виявлення підробки додатку є перевірка його підпису [3], оскільки для встановлення його на мобільний пристрій знадобиться сертифікат, який буде втрачено після перепакування. Звідси й з'являється необхідність цього методу захисту, який зупинить запуск фальсифікованого додатку, виявивши незбіжність між підписами, а також за допомогою JNI буде стійким до атаки зі зміною інформації про підпис, яку отримує PackageManager [3].

Попри зазначені переваги, цей метод захисту доволі легко обійти за допомогою інструмента динамічного аналізу Frida [4]. Цей інструмент має архітектурну модель «клієнт-сервер», який дозволяє обходити будь-які механізми захисту та підроблювати повернуті значення у додатку. Саме завдяки ньому пентестери аналізують виклики певних класів, щоб зрозуміти як додаток працює всередині. Спочатку Frida знаходить процес додатка, а після впроваджується у його пам'ять. Frida-server може бути запущеним на root-девайсі, або у якості динамічно завантаженої бібліотеки на старті програми.

Очевидним методом виявлення Frida та подібних фреймворків є перевірка середовища на наявність відповідних артефактів, таких як файли пакетів, бінарні файли, бібліотеки, процеси, тимчасові файли та інші.

Для підвищення стійкості мобільних додатків до динамічного аналізу необхідно імплементувати наступні елементи захисту:

– обов'язкова перевірка підпису додатку на нативному рівні;

– перевірка на наявність артефактів у пам'яті, файлах пакетів, бінарних файлах, бібліотеках, процесах та тимчасових файлах. Для Firda це може бути

frida-server, або рядок LIBFRIDA у виконуваній секції пам'яті, який свідчить про наявність спільної бібліотеки frida-gadget;

– перевірка відкритих TCP-портів. Процес frida-server за замовчуванням прив'язується до TCP-порту 27042;

– перевірка портів, що відповідають на запити D-Bus Auth.

На основі даних елементів захисту можна побудувати систему RASP, що дозволить виявляти та блокувати динамічні модифікації додатку, такі як Hooking, Memory Patching, Runtime Code Injection.

На основі цих досліджень було розроблено нативну бібліотеку, що підвищує стійкість мобільних додатків на платформі Android і у сукупності з обфускацією коду ускладнює аналіз навіть для кваліфікованих реверс-інженерів, що знизить ризик витоку конфіденційної інформації та порушення цілісності додатку. Методами підвищення стійкості також можуть зловживати зловмисні програми. Автори шкідливого програмного забезпечення часто використовують вищеперелічені методи стійкості, щоб створити перешкоди для дослідників та ускладнити криміналістичний аналіз. З цієї причини тестувальники повинні розуміти ці методи, знати, як їх виявити, і практикувати їх в обхід, в той час як розробники повинні додавати ці елементи, щоб забезпечити додатковий захист від атак, специфічних для загроз. Таким чином, розроблені елементи захисту підвищать стійкість мобільних додатків на платформі Android і запобігатимуть його аналізу менш кваліфікованими реверс-інженерами, що знизить ризик витоку конфіденційної інформації та порушення цілісності додатку.

Список літератури

1. MASVS-RESILIENCE: resilience against reverse engineering and tampering - OWASP mobile application security. OWASP Mobile Application Security - OWASP MAS. URL: https://mas.owasp.org/MASVS/11-MASVS-RESILIENCE/.

2. Сєвєрінов, О.В., Федорченко, В.М., Перепадя, В.І. Аналіз загроз персоальним обільному пристрої під час використання різноманітних додатків. *Системи озброєння і військова техніка*, 4 (2016): 42-45.

3. Сидоренко, З.М., & Мартовицький, В.О. (2022). Управління безпекою мобільних абонентських пристроїв у корпоративних мережах / "Проблеми інформатизації": ЧДТУ, ВА ЗС АР, УТіГН, НТУ "ХПІ", ХНУРЕ, "ПД ПКНДІ АП", С. 52

4. MASTG-KNOW-0030: detection of reverse engineering tools - OWASP mobile application security. OWASP Mobile Application Security - OWASP Mobile Application Security. URL: https://mas.owasp.org/MASTG/knowledge/android/MASVS-RESILIENCE/MASTG-KNOW-0030/.

5. GitHub - L-JINBIN/ApkSignatureKiller. GitHub. URL: https://github.com/L-JINBIN/ApkSignatureKiller.

6. Frida. Frida - A world-class dynamic instrumentation toolkit. URL: https://frida.re/.