СТРАТЕГІЇ ОБХОДУ СТАТИЧНОГО І ДИНАМІЧНОГО АНАЛІЗУ АНТИВІРУСНИМИ СИСТЕМАМИ ANDROID OS

Стеблина Олександр Станіславович
студент групи ІАСм -1-24-1.4д,
Київського столичного університету імені Бориса Грінченка,
науковий керівник – к.т.н., доц. Мельник І.Ю.

У контексті Android-безпеки використовуються два основних підходи до аналізу застосунків: статичний аналіз (аналіз коду та структури без виконання) та динамічний аналіз (аналіз поведінки при запуску в ізольованому середовищі). Я проаналізував найпоширеніші техніки ухилення від статичного аналізу (обфускація, пакування коду, поліморфізм/метаморфізм, алгоритмічне маскування) та динамічного (виявлення емульованого середовища, відкладене виконання, маскування реальної поведінки) і ось найдієвіші з них, на які треба звернути увагу:

Використання нативного коду. Нативні бібліотеки (.so) можуть вміщувати шифрований або заплутаний шкідливий код, який важко аналізувати стандартними засобами для DEX. При цьому сам нативний код може теж змінюватися (наприклад, використовувати поліморфні шифри для приховування). Тактика запускати частину логіки в нативі використовується як для оптимізації, так і для обходу - багато сучасних мобільних вірусів ховають критичну функціональність у .so, розраховуючи, що антивірус, націлений на Java/Dalvik, пропустить цю частину [1].

Обфускація і пакування коду. Шкідлива логіка маскується шифруванням, перейменуванням класів і змінних, захованими рядками, динамічним завантаженням .dex-файлів. Це робиться, щоб статичний аналіз не розпізнав зловмисних фрагментів за сигнатурами. Наприклад, поліморфні трояни успішно генерують безліч нових варіантів коду, змінюючи його структуру, але зберігаючи ту ж поведінку [2].

Анти-емуляційні та анти-дебаг прийоми. У динамічному аналізі антивірус зазвичай виконує додаток у віртуальному середовищі. Зловмисники перевіряють Build.MODEL, сенсори, SIM-карту й інші ознаки “реального” пристрою: якщо виявляється емулятор, троян вимикає шкідливу функціональність. Так, банківські трояни на кшталт BankBot або Anubis вмикали злочинну активність лише за умови, що пристрій справжній [3; 4].

Відкладена активація та anti-fuzzing. Деякі загрози чекають певного часу чи події (наприклад, перезавантаження або підключення до мережі) або дій від користувача, які не може зробити бот, щоб активувати шкідливий функціонал. Це ускладнює динамічний аналіз у sandbox, де зазвичай перевірка триває короткий проміжок. Відносно відомий приклад - BrainTest, що залишалася

непоміченою в Google Play, бо його шкідлива частина активувалася лише через кілька запусків [3; 5].

На завершення, можна сказати, що баланс між захистом та продуктивністю поступово зміщується в бік більшої глибини аналізу, оскільки загрози ускладняються. Завдяки хмарним технологіям і спільним зусиллям, антивірусні системи можуть дозволити собі витрачати більше ресурсів на перевірку підозрілих програм, не обтяжуючи пристрій кінцевого користувача. Це позитивна тенденція для безпеки. Та все ж не варто покладатися лише на автоматичні засоби. Необхідно дотримуватись базової гігієни безпеки: встановлювати додатки тільки з надійних джерел, оновлювати ОС і сигнатури AV, перевіряти дозволи, які запитує застосунок. Багато складних атак можна попередити ще на цьому рівні - не давши професійно замаскованому вірусу шансів проникнути на пристрій. Антивірусні технології продовжать удосконалюватись, але і зловмисники шукатимуть нові шляхи - тому пильність і багаторівневий підхід до безпеки залишаються найкращою рекомендацією.

ДЖЕРЕЛА

1. Chen L. Using capa Rules for Android Malware Detection | Google Cloud Blog. Google Cloud Blog. URL: https://cloud.google.com/blog/topics/threat-intelligence/capa-rules-android-malware-detection/ (date of access: 11.04.2025).

2. Elsersy W. The rise of obfuscated Android malware and impacts on detection methods - PMC. PMC Home. URL: https://pmc.ncbi.nlm.nih.gov/articles/PMC9044361/ (date of access: 11.04.2025).

3. Parvez F. A Survey and Evaluation of Android-Based Malware Evasion Techniques and Detection Frameworks. MDPI. URL: https://www.mdpi.com/2078-2489/14/7/374 (date of access: 11.04.2025).

4. Білоус, В. В., Бодненко, Д. М., Хохлов, О. К., Локазюк, О. В., & Стаднік, І. П. (2024). Open Source Intelligence for War Crime Documentation. In *Workshop Cybersecurity Providing in Information and Telecommunication Systems (CPITS 2024)* (No. 3654, pp. 368-375). Kyiv, Ukraine.

5. Abramov, V., Astafieva, M., Boiko, M., Bodnenko, D., Bushma, A., Vember, V., Hlushak, O., Zhyltsov, O., Ilich, L., Kobets, N., Kovaliuk, T., Kuchakovska, H., Lytvyn, O., Lytvyn, P., Mashkina, I., Morze, N., Nosenko, T., Proshkin, V., Radchenko, S., & Yaskevych, V. (2021). Theoretical and practical aspects of the use of mathematical methods and information technology in education and science. https://doi.org/10.28925/9720213284km