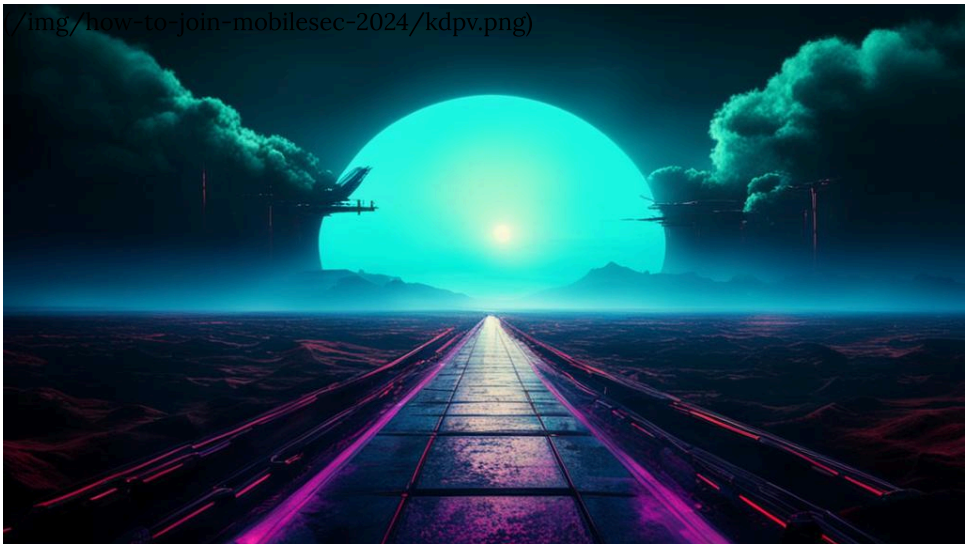
Как вкатиться в безопасность android приложений в 2024

С завидным постоянством получаю подобные вопросы лично или вижу их в нашем уютном чате Android Guards (https://t.me/android_guards). Поэтому решил систематизировать информацию чтобы упростить процедуру входа как для тех, кто только начинает свой путь в инфобезе, так и для тех, кто уже что-то умеет в соседних областях.



Особенности предметной области

По своей природе, поиск уязвимостей в мобильных приложениях гораздо ближе к работе бинарного реверсера, чем к работе специалиста по web приложениям. Обычно нет никакого смысла исследовать приложения black-box-ом и поэтому первое, что делается это декомпиляция файла приложения с последующим поиском точек входа и прочих интересных вещей. Поэтому любой *хороший* исследователь android приложений должен понимать как работает виртуальная машина и ОС в целом, какие бывают форматы файлов, не бояться байт-кода (местный ассемблер) и т.п. Все это требует определенного времени на освоение, и ожидать быстрых результатов не стоит. Разве что на CTF. Реальные же приложения, как правило, на порядок сложнее стандартных crackme типа InsecureBank (https://github.com/dineshshetty/Android-InsecureBankv2), на примере которых обычно демонстрируют уязвимости. Короче, базовый набор навыков должен выглядеть как-то так:

- Программирование: Java, Kotlin
- Архитектурные топики: MVP/MVVM, Dependecy Injection, шаблоны проектирования
- Android: основные компоненты приложений, библиотеки JetPack, Dalvik/ART, Android Debug Bridge
- Инструменты: декомпилятор (jadx, jeb, procyon, etc...), дизассемблер (ghidra, idapro), Frida (+frida based утилиты вроде `objection`), Magisk, Android Studio, apktool

Это минимальный набор. Отсутствие какого-либо знания из этого набора будет немного усложнять жизнь исследователя. Например: можно ли обойтись без знания DI? Да. Но тогда есть риск потратить лишнее время на изучения кода, который не нужно изучать, потому что он сгенерирован автоматически и к логике приложения отношения не имеет. Разве что без декомпилятора может быть совсем грустно, т.к. придется изучать не java-like псевдокод, а smali, который чуть более низкоуровневый и многословный. Но прелесть в том, что это все можно изучать параллельно с практикой. Главное не забывать это делать.

Литература

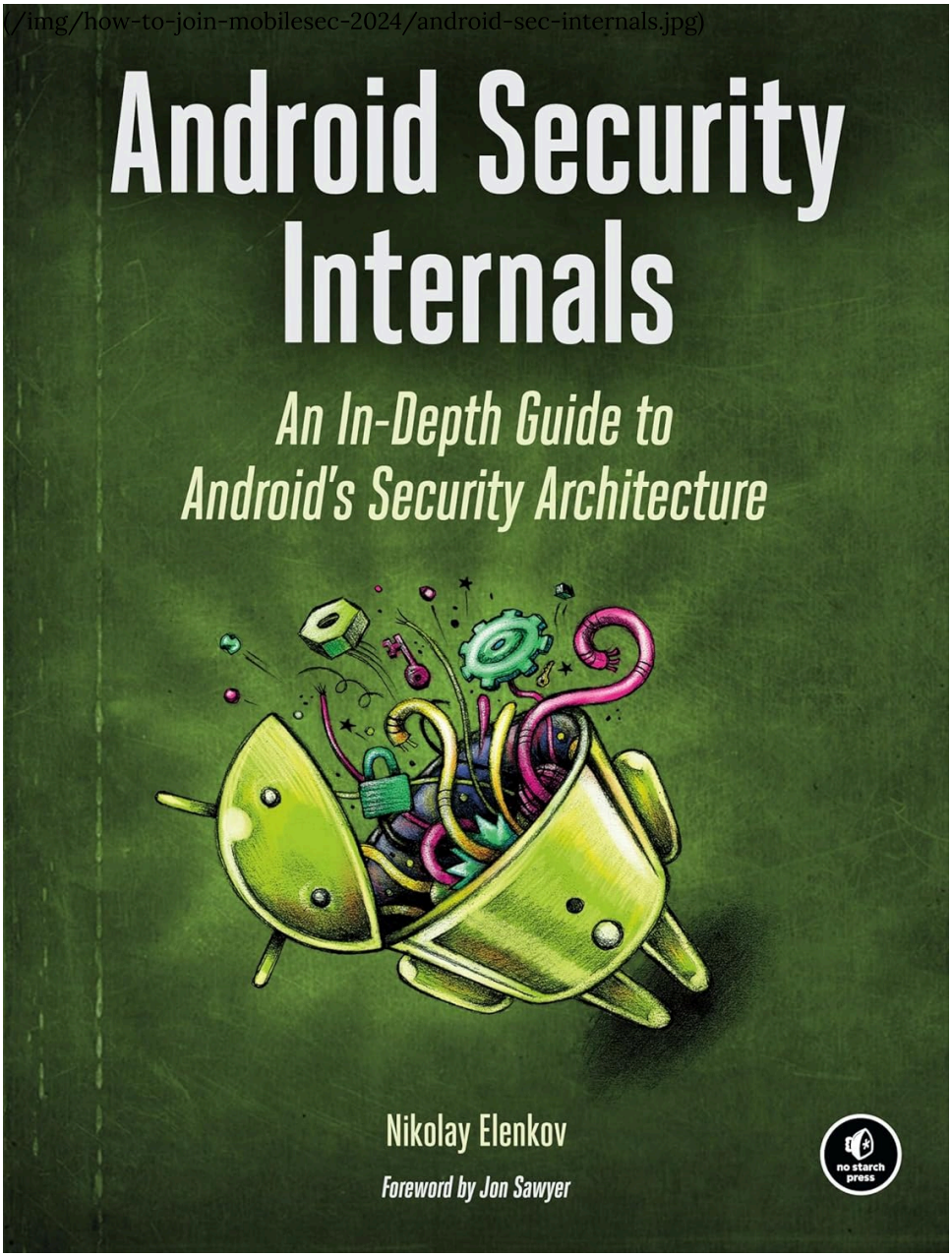
Хорошей литературы по анализу приложений не так много как хотелось бы и многие книги уже устарели частично или полностью. Да и не особо любит народ читать книги. Все любят статьи где быстро и доходчиво расскажут как чего-нибудь захекать и заработать свою первую штуку баксов. Но такие статьи хороши, когда уже есть некоторые базовые знания. Которые лучше получать из книг и официальной документации, чем из статей. В общем если есть мотивация читать книги, то рекомендую эти:

Android глазами хакера. 2-е изд.

(/img/how-to-join-mobilesec-2024/android-ez.webp)

Весьма неплохая книга, которая повествует обо всем по чуть-чуть, но тем и хороша для новичков. Прочитав ее вы получаете общее представление о том как устроен анализ android приложений и понимание куда двигаться дальше. У меня на канале есть обзор (https://www.youtube.com/watch?v=urgbx0QmHxs) на первое издание.

Android Security Internals



Просто отличная книга, которая, к сожалению, стремительно устаревает. Ее все еще можно читать, т.к. часть фундаментальных принципов не сильно изменилась. Книга была написана во времена выхода Android 6 и поэтому не учитывает очень многих современных механизмов. В общем можно полистать оглавление и почитать какие-то конкретные главы/разделы.

Android Internals (Vols I & II)

(/img/how-to-join-mobilesec-2024/levin2.png)


Очень толковые и довольно актуальные по наполнению книги. Они не совсем про безопасность, скорее про внутрянку ОС android и приложений. Непосредственно про безопасность автор обещает третью часть, которая пока в процессе написания. Но эти книги однозначно стоят прочтения. После, их можно использовать как справочник.

Пожалуй эта вся литература, которую я бы мог рекомендовать. Все остальное или безнадежно устарело или просто является низкокачественным продуктом не стоящим внимания.

Прочие обучающие материалы

Помимо книг, есть смысл изучить OWASP Mobile Top Ten (https://owasp.org/www-project-mobile-top-10/), который как раз обновился в прошлолм году. Я уже выкладывал небольшую выжимку (https://fi5t.xyz/posts/owasp-mobile-top-ten-2023/) этой новой редакции. Можно начать с нее, а потом уже углублять знания в официальном гайде. Ну и конечно нужно хотябы один раз прочитать Mobile Security Testing Guide (https://mobile-security.gitbook.io/mobile-security-testing-guide/) от OWASP. Там есть много полезной информации и всякие не попсовые техники анализа приложений.

Конечно же есть какое-то нереальное количество всяких курсов, лаб, статей и прочих youtube-менторов, которые обещают вас научить всему ~~за сущие копейки~~ и сразу. Этим тоже можно пользоваться, но с оглядкой. Часто качество

материала хромает. Вот список того, что стоит потраченного времени:

- Слив довольно сносного курса по реверсу android приложений (https://www.youtube.com/@user-hs1iy9ci3j/playlists)
- Блог Oversecured (https://blog.oversecured.com/)
- Блог Guardsquare (https://www.guardsquare.com/hs-search-results?term=android&type=BLOG_POST)
- Блог Quarkslab's (https://blog.quarkslab.com/category/android.html)
- Блог 8KSEC (https://8ksec.io/blog/)
- Блог The Binary Hick (https://thebinaryhick.blog/?category=android)
- Блог NowSecure (https://www.nowsecure.com/blog/)
- Telegram канал Android Security & Malware (https://t.me/androidMalware)

При вдумчивом изучении, этих материалов начинающему исследователю мобильных приложений должно хватить надолго. Особенно с учетом того, что постоянно появляются новые.

Пара слов о практике

Повторюсь - в реальности, разрыв между лабами/crackme и настоящими приложениями (особенно теми, которые представлены в bug bounty программах) довольно большой. Но есть очень хорошая опция, благодаря которой можно не только совершенствовать навыки анализа android приложений, но и приносить какую-то пользу как себе в виде некоего портфолио, так и разработчикам приложений. Эта опция называется F-Droid (https://f-droid.org/) (да и вообще любые приложения с открытым исходным кодом).

Прицнип такой: берем любое понравившееся приложение, лучше небольшого размера, и спокойно ищем в нем ошибки. При чем можно устроить себе "реальные условия" и не смотреть в исходный код сразу, а попробовать декомпилировать бинарь и разбираться уже с ним. А можно сразу смотреть white-box-ом. В любом случае это все будет полезно. У этого варианта практики одни плюсы: это уже не CTF-ка в которой точно есть уязвимости (часто глупые!), а вполне себе нормальное приложение, в которое разработчик не планировал ничего закладывать. При этом, само приложение часто не слишком сильно защищено всякой обфускацией, да и исходники можно всегда посмотреть. А также есть возможность подобрать комфортный размер кодовой базы для тренировки. За найденные таким образом уязвимости можно даже получить CVE или просто сделать доброе дело, зарепортив разработчику уязвимость.

Как выглядит процесс анализа

Как обычно, все зависит от задачи, но типичный аудит строится примерно следующим образом:

1. Каким-то образом получаем apk-файл. Тут вариантов много. Можно стянуть с телефона, можно скачать из F-Droid, AuroraStore или воспользовать утилитой вроде apkeep (https://github.com/EFForg/apkeep).
2. Декомпилируем любым понравившимся декомпилятором. Новичкам рекомендую начать с jadx (https://github.com/skylot/jadx). Для большего удобства, декомпилированный код можно выгрузить в виде проекта для Android Studio и в ней же открыть.
3. Анализируем дерево исходников на предмет всяких интересных конфигов (папка `assets`), строк (папка `res/values/strings.xml`) и вообще всего, за что зацепится глаз.
4. Ищем в файле `AndroidManifest.xml` экспортированные компоненты, диплинки (https://developer.android.com/training/app-links/deep-linking) и пакеты (https://t.me/android_guards_today/181) с которыми работает (https://t.me/android_guards_today/222) приложение.
5. После локализации всех точек входа на предыдущем шаге, приступаем к анализу реализации экспортированных компонентов на предмет возможности нестандартного взаимодействия с ними.
6. ??????
7. Где-то тут вы находите RCE 🤯

Так верхнеуровнево выглядит начало процесса полного аудита мобильного приложения. Там есть еще всякие ответвления типа анализа песочницы приложения и поиск там незашифрованной конфиденциальной информации, поиск файлов, которые приложение сохраняет в общедоступных директориях, да и много чего еще. Но задача полного аудита стоит не всегда и поэтому алгоритм будет варьироваться в определенных пределах. Хотя шаг с декомпиляцией существует практически всегда. Если только вы не хотите смотреть исключительно нативные библиотеки. Тогда можно ничего не декомпилировать, а просто распаковать приложение как архив и забирать библиотеки из каталога `libs/<arch>` на анализ в дизассемблер.

Вообще, анализ приложения можно разделить на два направления: статический и динамический. И довольно часто они дополняют друг-друга. Нет большого смысла убивать глаза всматриваясь в статику и “компилируя в голове”, если можно поставить хук на нужные классы/методы и посмотреть как они работают. Динамический анализ в основном выполняется с помощью DBI фреймворка Frida (https://frida.re/). Есть и другие, но начать я рекомендую именно с него. Вся суть работы с этим фреймворком сводится к написанию хуков на JavaScript чтобы понять как работают те или иные куски приложения. На базе Frida построено множество инструментов. Наиболее удобный на мой взгляд (из публичных): objection (https://github.com/sensepost/objection). Он сильно упрощает создание базовых хуков и позволяет не погружаться в недра JS и отвратительной документации по Frida 🤯. Кстати помимо документации есть пара книг для изучения Frida:

1. Beginning Frida: Learning Frida use on Linux and (just a bit on) Wintel and Android systems with Python and JavaScript (https://www.amazon.com/Beginning-Frida-Learning-Android-JavaScript/dp/B094ZQ1HHC) (+ мой небольшой обзор (https://t.me/android_guards_today/129) этой книжки).
2. Frida HandBook (https://learnfrida.info/) - можно использовать как продолжение предыдущего пункта. Разобрано довольно много интересных тем по бинарной инструментации.

Также, в большинстве приложений стоит задача анализа сетевого трафика. В основном это нужно для расширения поверхности атаки с целью поиска специфичных для “мобильного API” серверных уязвимостей. Для анализа, трафик нужно каким-то образом завернуть на прокси, но при этом нужно иметь предварительно настроенный девайс/эмулятор чтобы успешно перехватывать TLS соединения. Начать изучения этой темы рекомендую с этого гайда (https://fi5t.xyz/posts/intercept-android-traffic/). В качестве прокси можно использовать BurpSuite (https://portswigger.net/burp/documentation/desktop/mobile/config-android-device), Charles (https://www.charlesproxy.com/documentation/configuration/) или mitmproxy (https://docs.mitmproxy.org/stable/overview-getting-started/). Если хочется смотреть трафик прямо на устройстве, то можно воспользоваться приложением PCAPdroid (https://github.com/emanuele-f/PCAPdroid).

Заключение

Вряд-ли этим материалом я смог ответить на все возможные вопросы новичков, но он и не задумывался для этого. Это скорее конспект того, что я каждый раз пишу в приватные и публичные чаты, когда меня просят рассказать “как запрыгнуть в анализ android приложений?”. В качестве бонуса, прикреплю еще несколько материалов, которые я готовил в разное время как раз на аудиторию людей, желающих познакомиться с миром анализа android приложений:

- Открытая лекция: Основы информационной безопасности для мобильных разработчиков (https://www.youtube.com/watch?v=oZEFUA-unDo) - лекция была прочитана в рамках стажировки для мобильных разработчиков
- Как взламывают android-приложения и что после этого бывает (Workshop) (https://www.youtube.com/watch?v=x1d1ZnxHUms) - воркшоп по поиску уязвимостей в мобильных приложениях в рамках конференции Google devfest

- Безопасность Android приложений (https://www.youtube.com/watch?v=x_COvmEIyko) - посиделки в Android Broadcast на тему безопасности приложений.
- Вопросы новичков (https://www.youtube.com/watch?v=zeU2wlcj_Bw) - разбор наиболее часто встречающихся мне вопросов от новичков.

Теперь точно все. Легкой дороги!

Tags: #mobile (https://fi5t.xyz/tags/mobile/) #android (https://fi5t.xyz/tags/android/) #безопасность (https://fi5t.xyz/tags/%D0%B1%D0%B5%D0%B7%D0%BE%D0%BF%D0%B0%D1%81%D0%BD%D0%BE%D1%81%D1%82%D1%8C/)

Смотрите также

- Аутентификация по короткому коду с поддержкой биометрии. Почти RFC (/posts/pin-code-authentication-rfc/)
- Android Jetpack Navigation: Go Even Deeper (/posts/jetpack-compose-navigation-bugs/)
- Три случайных уязвимости (/posts/three-random-bugs/)
- CVE-2023-40121 (/posts/cve-2023-40121/)
- Святая корова SSL Pinning-а (/posts/sslpinning-holy-cow/)

← ПРЕДЫДУЩИЙ (HTTPS://FI5T.XYZ/POSTS/OWASP-MOBILE-TOP-TEN-2023/)

СЛЕДУЮЩИЙ → (HTTPS://FI5T.XYZ/POSTS/SSLPINNING-HOLY-COW/)