2. C. Lindemann, U. Jahnke, M. Moi, R. Koch. Analyzing Product Lifecycle Costs for a Better Understanding of Cost Drivers in Additive Manufacturing / Conference: Solid Freeform Fabrication Symposium - An Additive Manufacturing Conference At: Austin, TX, USA Volume: 23th
3. Агаповичев А.В., Сотов А.В., Смелов В.Г. Исследование структуры и механических свойств изделий, полученных методом селективного лазерного сплавления из порошка стали 316 L / Черные металлы, 2017, № 9, С. 61-65.

ЗАЩИТА АНДРОЙД ПРИЛОЖЕНИЯ ОТ РЕВЕРС-ИНЖИНИРИНГА

Жолнеров Д.И., студент группы ФИТ 14-2,
научный руководитель: старший преподаватель кафедры ИТБ Клюева Е.Г.
Карагандинский государственный технический университет
100027, Карагандинская обл., г. Караганда, Бульвар Мира, 56

В настоящее время, мобильные приложения разрабатываются с небывалой скоростью и с уверенностью завоевывают рынок прикладного программного обеспечения. Рынок мобильных приложений в основном принадлежит двум операционные системы - Android и iOS. Так как Android устройство является наиболее доступным, соответственно и рынок охватывает гораздо большее количество пользователей.

С инструментами для разработки приложений дела обстоят также. Чтобы начать разрабатывать приложения для Android, достаточно знать основы одного из самых популярных языков программирования - Java, бесплатной среды разработки Android Studio и официальной документации по Android SDK. Поэтому многие начинают разрабатывать свои приложения именно под эту операционную систему.

Множество разработчиков создали огромное количество уникальных приложений. Не все хотят покупать приложения, либо мириться с наличием рекламы в нем. Поэтому умельцы придумывают разные методы, чтобы пользоваться бесплатно и без рекламы тем, что определенно заслуживает своей платы.

Не всегда модификации приносят пользу потенциальному пользователю приложения, зачастую в приложения встраивают вирусный программный код, пользователи устанавливают приложения ожидая получить крутую функцию, которой нет в официальной версии приложения, но получают проблемы с устройством и даже утечкой информации. Разработчикам такие модификации в большинстве случаев не нужны. Так как зачастую в модифицированных версиях платный контент делают бесплатным, что в свою очередь ведет к снижению прибыли.

На следующем рисунке представлена сравнительная статистика взломанных приложений на двух самых популярных мобильных операционных системах.

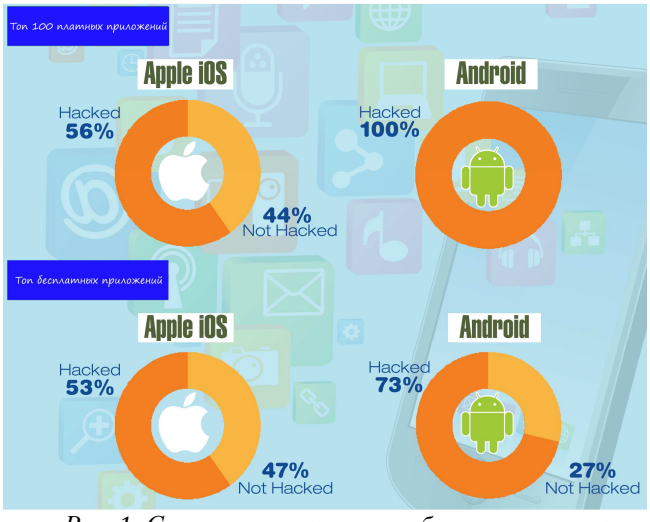

Рис. 1. Статистика взлома мобильных приложений

Для борьбы с хакерами разработчики используют разные способы защиты программного продукта. Эффективность метода зависит от его уникальности. Не стоит использовать тот, который выставлен в сети интернет, и доступен тому же хакеру.

Для того чтобы модифицировать Android приложение не нужно знать язык ассемблера, достаточно прочитать пару статей и приступить к практике.

Чтобы грамотно защитить свое приложение, нужно уметь его изменить без наличия исходного кода, т.е. модифицировать, сделать то, что захочет сделать потенциальный взломщик.

Для модификации приложения нужно проделать всего несколько шагов, в общем виде следующие: распаковать программу с помощью специальной утилиты (например, apktool), изменить код, запаковать, подписать своим ключом. Код в распакованном виде будет выглядеть иначе, не так как привыкли java программисты. Этот код называется Smali и схож с Java-байт-кодом. Но, так как в процессорах ARM-архитектуры много регистров, Google решили сэкономить и заменить долгие прогулки в память (в стек, как в JVM) на быстрые походы в регистры. Поэтому основное отличие байт-кода Dalvik от байт-кода JVM - ориентированность на регистры. Также существуют утилиты, которые конвертируют Smali в Java, с их помощью можно быстрее разобраться с кодом и понять, что и где изменить, исходя из поставленной цели. Но менять нужно именно Smali код, так как конвертировать придется его.

После того как получится изменить свое приложение, можно приступить к разработке кода способного защитить его от взлома. Для защиты приложения существует несколько решений, их комплексное применение позволит уменьшить вероятность взлома.

На самом деле нет такого приложения, которое нельзя взломать, но можно сделать защиту такой, чтобы ее взлом не был оправдан, из-за сложности и потраченного времени. Бывают случаи, когда защита и не нужна вовсе, но даже в таком случае можно произвести не затратный прием, который затруднит взлом приложения. Такой прием называется - обфускация кода, по-другому запутывание. Благодаря этому приему, анализ кода займет больше времени, а функционал не изменится.

Запутывать свой код самому плохая идея, лучше всего воспользоваться специальной программой, называемой обфускатор. К тому же данная программа Proguard встроена в среде разработки Android Studio и доступна бесплатно. Для того чтобы приложение выполняло свой функционал после применение данной программы, необходимо прописать правила, которые пропустят некоторые части кода ввиду их особенностей, что может привести к трудностям, но это единственная плата за пользование этой программой.

Теперь можно приступить к следующему этапу, защита подписи приложения. Дело в том, что, любое приложение подписывается специальным ключом, прежде чем его можно будет установить. Благодаря этому нельзя обновить приложение в случае, если обновление не подписал сам разработчик. Но пользователи в большинстве готовы удалить приложение от официального разработчика и установить его модификацию.

Чтобы обнаружить несовпадение подписи, нужно встроить в программу специальный код, который произведет сравнение подписей. В случае несовпадения можно вывести сообщение о просьбе установить официальную версию приложения или применить друге санкции, например, урезать функционал.

Так, как и такой код можно убрать или изменить, нужно усложнить его поиск. Для этого можно написать часть функционала на языке C/C++. Android Studio при создании проекта предложит вам включить поддержку C/C++ кода. В этом случае модифицировать код будет еще сложнее.

Так как функционал будет расположен в приложении в виде динамически загружаемой библиотеки (DLL), потенциальному взломщику понадобятся навыки дисассемблирование нативного кода Android, что сразу отсекает любителей, которые взламывают все подряд. Применив к коду обфускатор C/C++, можно будет приобрести больше уверенности, что взлом затянется на долго.

Если есть возможность использовать сервера для функционирования приложения, тогда некоторый функционал можно переписать на них, таким образом, взлом сервера будет гораздо сложней.

Исходя из выше сказанного, можно сделать следующие выводы:

- при разработке приложения необходимо использовать обфускатор кода;
- часть функционала, который отвечает за безопасность приложения лучше всего писать на языках C/C++;
- по возможности особо важный функционал стоит перенести на web-сервис;

- при тестировании приложения необходимо использовать модель потенциального взломщика, если разработчик не сможет взломать свое приложение, хакерам будет еще сложнее достичь этого.

Литература.

1. Android за 24 часа. Программирование приложений под операционную систему Google (2011). – Лорэн Дерси, Шейн Кондер.
2. Design Patterns: Elements of Reusable Object-Oriented Software (2009) – Э. Гамма, Р. Хелм, Р. Джонсон, Д. Влиссидс.
3. Java. Методы программирования (2013) - И. Н. Блинов. В. С. Романчик.
4. Java. Эффективное программирование (2012) – Д. Блох.
5. Android Studio [Элекстронный ресурс], developer.android.com : Сайт разработчика URL: https://developer.android.com/studio/intro/index.html.
6. Ретабоуил Сильвен Android NDK. Разработка приложений под Android на C/C++; ДМК Пресс - Москва, 2012. - 496 с.
7. Habrahabr. Защита Android приложений от взлома [Элекстронный ресурс] URL: https://habrahabr.ru/post/179487/.

УМНАЯ ПАРКОВКА, ЕЕ ПРЕИМУЩЕСТВА И ПРОБЛЕМЫ

И.В. Грасмик, студент группы 17В41
Юргинский технологический институт (филиал)
Томского политехнического университета
652055, Кемеровская обл., г. Юрга, ул. Ленинградская, 26

Создание парковочных мест для автомобилей началось в одно время с появлением первых автомобилей. Количество автомобилей очень быстро растет и для решения возникнувших проблем ограниченности стояночных мест стали внедрять современные технологии.

Основным назначением умных парковок является создание удобных условий для граждан, повышение качества жизни и лояльности клиента к объекту, точное определение количества и месторасположения свободных мест на стоянке, передача информации об этом водителю и персоналу, обеспечение простой и понятной навигации на автостоянке, регулирование трафика и т.д.

В основе умных парковок лежат датчики, проверяющие наличие автомобилей. Они устанавливаются над каждым парковочным местом. Такие датчики называются «умными» и используют следующие технологии связи: LoRa, NB-IoT, SigFox, RFID. Датчики могут работать при температурах от -40 до +50°С. Напряжение питания периферийных устройств не превышает 24В и соответствует принятым нормам безопасности. Подключение этих датчиков к электросети не требуется, так как питаются они за счет аккумуляторов, из-за чего считаются экономичными. Это очень сильно облегчает работу по проектированию, а также и непосредственно саму установку датчиков и увеличивает скорость процесса организации всего проекта. На всей территории парковки размещаются датчики, фиксирующие факт постановки автомобиля на парковочное место. Информация, которую собрали датчиками, в дальнейшем передается на базовую станцию и закрепляется в базе данных. Тут происходит обработка полученной информации, формируются различного рода аналитические отчеты для будущего использования различными службами.

Информация с датчиков поступает на:

- Визуальные индикаторы, сообщающие водителя о том свободно ли или же занято парковочное место;
- информационное табло показывающее сколько на данный момент имеется свободных мест, а также их направление;
- сервер системы для программной обработки информации и отображения ее на мониторе оператора.

При въезде на автостоянку устанавливается информационное табло, сообщающее количество свободных мест на автопарковке. Данное информационное табло в соответствии с потребностями заказчика могут быть различных видов: двух- или же трехразрядными, со стрелками или без. Дальше, по парковке водителю автомобиля следует двигаться в соответствии с указательными стрелками, которые показывают направление и информационное табло уровня, подсказывающие, в какой части стоянки и в каком количестве имеются свободные места.