A.A. Skovoroda, D.U. Gamayunov

Review of the Mobile Malware Mitigation Approaches

Keywords: mobile devices, malware, security

Mobile devices such as smartphones and tablets are extremely widespread nowadays. These devices provide users with a wide range of applications for commercial and public use. However, the contents of applications and their full behavior are not always properly reviewed which makes the presence of malware in the application marketplaces possible. Mobile security researchers have proposed many effective solutions for detection and prevention of malicious applications on mobile devices. This paper provides a comprehensive review and comparison of the most recent approaches to mobile malware mitigation.

А.А. Сковорода, Д.Ю. Гамаюнов

ОБЗОР МЕТОДОВ ОБНАРУЖЕНИЯ И ПРОТИВОДЕЙСТВИЯ ВРЕДОНОСНЫМ ПРИЛОЖЕНИЯМ ДЛЯ МОБИЛЬНЫХ ПЛАТФОРМ

Введение

Вредоносное программное обеспечение (ВПО) для мобильных платформ стало представлять существенную угрозу, когда получили широкое распространение смартфоны и электронные планшеты. Эти устройства значительно меньше персональных компьютеров (ПК), они дают пользователям возможность реализации ежедневных потребностей, таких как просмотр веб-страниц, обмен мгновенными сообщениями, интернет-банкинг и прочее. На мобильных устройствах часто хранится личная информация, различные учетные данные и сведения о банковских картах, что делает их подходящей целью для атак злоумышленников.

Глядя на историю мобильных угроз, легко отметить, что в каждый момент времени наиболее популярная операционная система была больше всего подвержена атакам. Первый мобильный червь Cabir появился в июне 2004, он был нацелен на ОС Symbian и был создан скорее для демонстрации. Он распространялся по Bluetooth на близлежащие устройства, передавая специальный .sis-файл. Однако в дальнейшем эта возможность была использована для распространения различных вредоносных приложений (ВП), например, Pbstealer, основная функция которого заключается в краже контактов из телефонной книги. Согласно [1], основными типами мобильных ВП в 2004-2006 годах были троянские приложения, имеющие целью получение финансовой выгоды, и «трояны-вандалы», созданные для причинения вреда или вывода устройства из строя. Также в мобильных ВП того времени встречалась такая функциональность, как создание каналов управления смартфоном, отключение системных и сторонних приложений, блокирование карты памяти, кража данных. ВП имели возможность самораспространения по Bluetooth и MMS. Позже стало использоваться распространение через подключаемые устройства. В 2007-2009 годах появились новые ВП, нацеленные на повреждение/удаление пользовательских данных, отключение механизмов безопасности ОС, звонки на платные номера. ОС Symbian оставалась лидирующей по количеству ВП до 2009 года.

Следующей основной целью для распространения мобильных ВП стала Java 2 Platform, Micro Edition, набравшая популярность в 2010 году. В разработанных ВП не было отмечено использования новых техник, наиболее распространенными были SMS-

трояны. В некоторых версиях троянов, например, Sejweek, адреса и тексты сообщений находились под контролем удаленного сервера. Значительную угрозу также представляли трояны, крадущие данные для доступа к электронным платежам и коды авторизации для банковских транзакций в интернете. Первые вредоносные приложения под Android и iOS появились в этот же период [1].

С 2011 года платформа Android стала наиболее привлекательной для злоумышленников. Основные цели атакующих – это кража данных или денег, получение контроля над устройством и выгоды от SMS-троянов [1]. В 2013 был отмечен значительный рост числа троянских приложений для мобильных платежей. Согласно отчету о ВП от Kindsight Security Labs, уровень зараженности мобильных устройств увеличился на 20 % в 2013, составив по оценкам компании 11,6 миллионов зараженных мобильных устройств по всему миру [2]. Также отмечено, что 60 % зараженных устройств – телефоны с ОС Android.

Первые мобильные ВП имели несколько путей распространения: Bluetooth/MMS, уязвимость в системе, некоторый интернет-ресурс. С возникновением централизованных рынков мобильных приложений распространение ВП стало еще проще для злоумышленников. Некоторые рынки приложений, например, Apple App Store и Windows Phone Store, проводят обязательный ручной анализ предлагаемых разработчиками приложений, в то время как Google Play Store проводит только автоматическую проверку новых приложений, используя Google Bouncer. Тщательная проверка значительно снижает количество выкладываемых вредоносных приложений, хотя и не исключает их появления (к примеру, приложение Find and Call в Apple App Store).

Для каждой из мобильных платформ существует несколько рынков приложений, помимо официального. Уровень зараженности в альтернативных рынках на порядок выше, чем в официальном, соответствуя примерно 0,02 % всех приложений в официальном Android Market и от 0,20 % до 0,47 % в альтернативных рынках [3].

Когда появились первые мобильные ВП, существовало множество методов обнаружения ВПО для ПК. У мобильных устройств и ПК похожее программное и аппаратное обеспечение, поэтому в организации их информационной безопасности много общего, однако, есть некоторые особенности в обнаружении мобильных ВП. Работа [4] описывает аспекты, специфичные только для мобильной безопасности. Основные отличия заключаются в ограниченности ресурсов мобильных устройств, дополнительных методах распространения мобильных ВП, и, в связи с этим, бóльших потерях от атак.

В данной работе представлен обзор существующих решений в области мобильной безопасности. Большинство рассмотренных методов предназначено для ОС Android из-за преобладания вредоносных приложений под эту платформу.

Функциональность вредоносных приложений

Вредоносная активность современных мобильных ВП очень многообразна. Наиболее часто встречается *кража пользовательских данных*, например, учетных данных, номеров счетов, месторасположения, контактов и прочих данных. Сюда относятся также сенсорные ВП, которые используют информацию от работающих сенсоров, например, данные акселерометра, для получения конфиденциальной информации, например, пользовательского ввода [5]. Собранная информация о пользователях приложения может быть использована злоумышленниками произвольным образом. *Примеры:* Plangton (2011), Svpeng (2013). Еще одна очень распространенная функция ВП – это *пользование платными сервисами* без согласия пользователя. Такие ВП рассылают SMS или осуществляют вызовы на платные номера без согласия пользователя. *Примеры:* трояны

OpFake, FakeInst, Obad (2013). OpFake и FakeInst также включают *рассылку SMS-спама* с вредоносной ссылкой по всем контактам из телефонной книги жертвы.

Некоторые ВП *устанавливают каналы удаленного доступа,* используемые для управления зараженным устройством, например, Brador (2004), троян Svpeng (2013). Также были встречены ВП, б*локирующие функциональность ОС или пользовательские данные* в целях получения выкупа, например, Cardblock (2005), Simplocker (2014). Еще один тип вредоносной активности – *изменение или удаление пользовательских данных, системных или сторонних приложений,* используется в основном так называемыми «троянами-вандалами», они часто не приносят никакой финансовой выгоды атакующим. *Пример:* Skuller (2004).

Помимо основной функции, вредоносные приложения часто несут некоторую дополнительную, необходимую для достижения цели, например, *атаки на получение прав администратора* (root-эксплоиты), *атаки на доверие* (confused deputy attacks) легитимных, системных или сторонних приложений с целью получения их привилегий [6], *атаки со сговором* (collusion attacks) нескольких приложений с целью получения множества привилегий, не подтвержденных пользователем для каждого из этих приложений по отдельности [6]. Некоторые методы обнаружения вредоносных приложений опираются на нахождение такой дополнительной функции.

Методы обнаружения мобильных вредоносных приложений

Рассмотрим методы обнаружения, предложенные для борьбы с мобильными вредоносными приложениями. Эти методы нацелены на указание потенциальной опасности приложения.

Для сравнения методов обнаружения учитывались критерии, значения которых содержат табл. 1 и 2. Обозначение «НИ» в ячейке таблицы означает «Нет информации», а обозначение «–» означает, что соответствующий параметр не может быть измерен для метода обнаружения. Методы обнаружения сгруппированы по используемым в них техникам и представлены в хронологическом порядке внутри каждой из групп.

Критерии сравнения

1. *Предполагаемый пользователь.* Метод обнаружения может быть предназначен для владельца устройства, либо для аналитика ВП в зависимости от предоставляемой им информации.

2. *Анализируемый объект.* В качестве анализируемого объекта могут выступать приложение, группа приложений или полное состояние системы.

3. *Подход к обнаружению.* По внутренней организации методы можно разделить на обнаружение злоупотреблений и обнаружение аномалий.

4. *Вредоносные действия, обнаруживаемые системой.* Возможная вредоносная активность приложений описана выше.

5. *Результаты анализа.* Могут содержать явное предположение о вредоносности/оценку опасности анализируемого объекта, либо информацию для дальнейшего ручного анализа. Есть группа методов, которые предоставляют пользователям советы по безопасности или информацию, ценную для аналитиков.

6. *Место анализа.* Анализ может выполняться непосредственно на мобильном устройстве, на удаленном сервере или с использованием облачных ресурсов. Он может быть распределен между перечисленными ресурсами, а также выполняться на группе мобильных устройств.

7. *Момент анализа.* Анализ может выполняться во время установки приложений, после их загрузки. Такой анализ обычно предотвращает владельцев устройств от использования потенциально опасных приложений. Анализ во время исполнения приложения может быть нацелен на обнаружение и предотвращение утечек конфиденциальной информации. Специалисты в области безопасности мобильных приложений могут запускать анализ независимо от мобильного устройства.

8. *Поддерживаемая мобильная ОС.*

9. *Дополнительные накладные расходы на вычисления.* Системы обнаружения и противодействия вредоносным приложениям требуют дополнительных ресурсов, которые могут быть критическими для устройства. Различают несколько типов накладных расходов: по времени работы, требуемой памяти и расходуемой энергии.

10. *Набор данных для тестирования.*

11. *Ошибки первого и второго рода* (только для методов, дающих явное предположение о вредоносности анализируемого объекта).

12. *Скорость работы анализа.*

Статический анализ

В методах, основанных на статическом анализе, содержимое приложения исследуется без использования запуска или эмуляции исполнения. Декомпиляция часто является первым шагом анализа, а затем строятся специфические структуры данных, отражающие интересующие свойства приложения.

В работе [7] предлагается система обнаружения вредоносных приложений PiOS, нацеленную на обнаружение возможных утечек конфиденциальной информации с телефона. На первом шаге анализа дешифруется бинарный Objective-C файл с приложением и строится граф потока управления анализируемого приложения. Полученный граф проверяется на отсутствие утечек данных – путей, берущих начало от функций, которые получают некоторый критический ресурс, и ведущих к точкам его передачи по сети.

Система Social-AV [8] основана на использовании сигнатур, главная идея заключается в разделении базы данных с сигнатурами между группой устройств в социальной сети. Сигнатуры разделяются на часто и редко встречаемые («горячее» и «холодное» множества соответственно). «Горячее» хранится на каждом из устройств в группе, остальная часть сигнатур разделяется между устройствами. Когда на устройство загружается новый файл, он сопоставляется с «горячим» множеством сигнатур, после этого с «холодным» множеством, и в случае, если не было обнаружено совпадений, отправляется запрос на соседнее устройство. Эксперименты показали, что предложенная схема почти вдвое уменьшает количество памяти, потребляемой антивирусным агентом.

С увеличением ресурсов смартфонов снижается необходимость использования методов социального взаимодействия для обнаружения ВП. Вместо этого, исследователями используются значительно более сложные методы по сравнению с сигнатурным подходом к обнаружению.

В [9] определяются приложения с высоким уровнем риска как приложения, эксплуатирующие программные уязвимости, а приложения со средним уровнем риска – как приложения, следствием работы которых являются финансовые потери или кража конфиденциальной информации. В средстве их анализа RiskRanker реализован сигнатурный анализ для обнаружения известных эксплоитов и статический анализ, основанный на символьном исполнении, для обнаружения динамической загрузки кода, ис-

пользования нативного кода, а также методов шифрования. Комбинация таких свойств считается представляющей высокий уровень риска.

AppProfiler может быть использован для построения профилей безопасности приложений [10]. На основании существующих исследований и проведенного анализа была построена база данных соответствий между вызовами API и возможными действиями приложения, описывающими цель их использования. Анализ приложения состоит из декомпиляции приложения, сопоставления извлеченных API-вызовов с правилами из базы знаний и обработки множества совпавших правил с целью получения профилей безопасности.

Большинство последних разработок, использующих статический анализ для исследования мобильных приложений, включает также динамическую часть для усовершенствования анализа или предоставляет средства для проверки результата, например AppIntent [11]. AppIntent – система, нацеленная на обнаружение утечек конфиденциальной информации без согласия пользователя. В ней используется статический анализ для обнаружения путей передачи информации и точек возможной утечки, этот шаг алгоритма завершается построением графа событий с критическими (событие, обработчик которого содержит как минимум одну инструкцию некоторого пути утечки) и существенными событиями (необходимыми предусловиями критических событий). Дальнейший анализ включает управляемое символьное исполнение, в котором генерируются цепочки событий и входные данные, приводящие к передаче конфиденциальной информации. AppIntent включает средство для упрощения работы аналитика: приложение переупаковывается таким образом, что при запуске генерируется цепочка событий, приводящая к утечке информации, а также подсвечиваются соответствующие элементы графического интерфейса. Это средство также позволяет проверить результаты статического анализа, чтобы избежать появления ложных срабатываний.

Динамический анализ

В методах, основанных на динамическом анализе, исследуемое приложение исполняется, либо используется эмуляция его исполнения. Полученная информация может быть проанализирована на различных уровнях абстракции для составления представления о поведении приложения.

Система TaintDroid [12] проводит динамический taint-анализ Android-приложений. Мобильное приложение исполняется в виртуальной машине Dalvik. При этом выполняется taint-анализ сразу на четырех уровнях: переменных, методов, сообщений и файлов. Taint-анализ помечает и отслеживает данные, берущие начало от конфиденциальных ресурсов. Основная цель заключается в определении того, что данные являются производными от конфиденциальных перед тем, как они покинут устройство по сети.

Многие системы динамического анализа используют также техники статического анализа непосредственно перед запуском приложения для того, чтобы определить предусловия потенциально опасных действий и сократить время анализа приложения, поскольку динамический анализ является более ресурсоемким.

Таблица 1. Сравнение методов обнаружения мобильных ВП (I)

Метод	Предполагаемый пользователь	Анализируемый объект	Подход к обнаружению	Представление результата	Место анализа	Время анализа	Поддерживаемая ОС
Статический анализ							
PiOS [7]	Фналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Возмодные утечки данных	Удаленный сервер	Независимо от мобильного устройства	iOS
Social-AV [8]	Владелец устройства	Сомнительные данные, например, приложение	Обнаружение злоупотреблений	Вредоносный или легитимный	Группа мобильных устройств	Установка (получение данных)	Не указана
RiskRanker [9]	Аналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Вредоносный или легитимный	Удаленный сервер	Независимо от мобильного устройства	Android
AppProfiler [10]	Владельцы устройств и аналитики мобильных ВП	Приложение	Обнаружение злоупотреблений	Профиль приложения	Удаленный сервер	Независимо от мобильного устройства	Android
AppIntent [11]	Аналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Пути возможных утечек данных	Мобильное устройство + удаленный сервер	Независимо от мобильного устройства	Android
Динамический анализ							
TaintDroid [12]	Владелец устройства	Приложение	Обнаружение злоупотреблений	Возможные утечки данных	Мобильное устройство	Исполнение	Android
SmartDroid [13]	Аналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Предусловия вредоносного поведения	Удаленный сервер	Независимо от мобильного устройства	Android
DroidRanger [3]	Аналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Вредоносное или легитимное	Удаленный сервер	Независимо от мобильного устройства	Android
VetDroid [14]	Аналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Граф вызовов функций	Мобильное устройство	Исполнение	Android
AppsPlayground [17]	Аналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Отчет	Независимо в эмуляторе	Исполнение	Android

Окончание табл. 1

Анализ привилегий							
Kirin [19]	Владелец устройства	Приложение	Обнаружение злоупотреблений	Предположение об опасности	Мобильное устройство	Установка	Android
WHYPER [20]	Владелец устройства	Приложение	Обнаружение злоупотреблений	Соответствие описания запрашиваемым требованиям	Удаленный сервер	Независимо от мобильного устройства	Android
Машинное обучение							
DroidAPIMiner [21]	Владельцы устройств и аналитики мобильных ВП	Приложение	Машинное обучение	вредоносное или легитимное	Мобильное устройство	Установка	Android
MAST [22]	Аналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Оценка опасности	Удаленный сервер	Независимо от мобильного устройства	Android
Машинное обучение [23]	Владельцы устройств	Состояние системы	Обнаружение аномалий	Вредоносное или легальное	Мобильное устройство + удаленный сервер	Исполнение	Android
Dendroid [24]	Вналитик мобильных ВП	Приложение	Обнаружение злоупотреблений	Семейство ВП для образца	Удаленный сервер	Независимо от мобильного устройства	Android
Мониторинг уровня заряда аккумулятора							
Мониторинг уровня заряда аккумулятора [26]	Владелец устройства	Состояние системы	Обнаружение аномалий	Вредоносное или легитимное	Мобильное устройство + удаленный сервер	Исполнение	Windows Mobile OS
VirusMeter [27]	Владелец устройства	Состояние системы	Обнаружение аномалий	Предупреждение о возможном ВП	Мобильное устройство	Исполнение	Symbian OS
Использование облачных вычислений							
Paranoid Android [29]	Владелец устройства	Состояние устройства (трейс)	Любой тип обнаружения	Вредоносное или легитимное	Устройство + облачные ресурсы	Исполнение	Android
Crowdroid [30]	Владелец устройства	Группа приложений	Обнаружение злоупотреблений	Вредоносное или легитимное	Устройство + удаленный сервер	Исполнение	Android

* Только половина методов может быть использована напрямую владельцами устройств. Большинство методов требуют дополнительных вычислительных ресурсов.

Система SmartDroid [13] предназначена для автоматического определения последовательности действий в работе с графическим интерфейсом приложения, приводящей к проявлению его вредоносной активности. Сначала строится граф вызовов функций и граф деятельностей анализируемого приложения на основе его декомпилированного представления, а затем выполняется динамический анализ для получения искомой последовательности событий. Динамический анализ включает обход элементов графического интерфейса с отсечением порождаемых деятельностей, не приводящих к выполнению опасного API. SmartDroid модифицирует эмулятор Android для сохранения исполняемых API-вызовов и их параметров в файл журнала.

В средстве DroidRanger [3] применяется комбинация техник статического и динамического анализа для классификации приложения как вредоносного или легитимного. Приложения фильтруются на основании запрашиваемых опасных привилегий. Далее подозрительные приложения сопоставляются с поведенческими отпечатками известных ВП, которые включают специальную информацию из манифеста приложения, структуру его кода, используемые пакеты, и граф вызовов функций. Приложения с динамической загрузкой кода, либо использующие нативный код могут оказаться неизвестными ВП, поэтому они проверяются динамически. API-вызовы с конкретными параметрами и некоторые системные вызовы записываются в файл журнала, который затем анализируется вручную, и при обнаружении нового ВП пополняется множество известных отпечатков.

Система VetDroid использует схему динамического анализа с учетом привилегий [14]. Анализируемое приложение исполняется в виртуальном окружении в течение некоторого времени. Для имитации пользовательских действий используется Android Monkey Tool [18], а также генерируются события (новое SMS, изменившееся местоположение и др.), для которых прописан обработчик. Модуль анализа привилегий извлекает все точки явного (возникающие при запрашивании ресурса) и неявного использования привилегий, а также связи между ними. Обнаруженное поведение сохраняется в файл журнала, который используется в построении графа вызовов функций для дальнейшего отложенного анализа. VetDroid модифицирует ядро Linux, драйвер Binder и виртуальную машину Dalvik для проведения такого анализа.

TaintDroid был первой системой динамического анализа, предложенной для Android. Он был использован в других системах, таких как DroidBox [15], AppFence [16], AppsPlayground [17]. TaintDroid предполагает взаимодействие пользователя с приложением вручную, поэтому не может быть напрямую использован в автоматическом анализе. В более поздних системах динамического анализа такое взаимодействие реализовано в каком-либо виде. Наиболее простой способ имитации действий пользователя - это использование Android Monkey Tool, который генерирует псевдослучайные последовательности действий пользователя, такие как щелчки, прикосновения или жесты, а также некоторые системные события [18]. Такое взаимодействие используется в VetDroid. Более усовершенствованные методы, такие как AppsPlayground, учитывают интерфейс приложения и его контекст при эмуляции пользовательского взаимодействия, что обеспечивает большее покрытие кода анализируемого приложения.

AppsPlayground [17] реализует обнаружение возможных утечек данных с помощью taint-анализа, основанного на TaintDroid. Отслеживается также использование потенциально опасных API-вызовов. Анализ на уровне ядра реализован для защиты от известных эксплоитов. Маскировочные техники позволяют создать более реалистичную среду для анализа. Техники исследования кода, такие как генерация событий, «умное исполнение», моделирующее взаимодействие пользователя с отображаемым интерфейсом, случайные события, обеспечивают более полное покрытие кода.

Анализ привилегий

Методы, основанные на анализе привилегий, используют только информацию из манифеста приложения для того, чтобы оценить опасность его использования. Система безопасности Kirin [19] сопоставляет конфигурацию безопасности анализируемого приложения со множеством предопределенных правил. Эти правила представляют из себя опасные конфигурации запрашиваемых привилегий и строк действий (action string), объявленных в манифесте приложения. Если устанавливаемое приложение удовлетворяет хотя бы одному правилу, оно считается опасным, и Kirin сообщает об этом владельцу устройства.

В работе [20] описана система WHYPER, служащая для проверки соответствия запрашиваемых привилегий описанию приложения. Авторы используют техники обработки естественных языков для анализа описаний. Затем обработанные описания сопоставляются с семантическими графами, представляющими привилегии.

Машинное обучение

Данный подход к обнаружению ВП использует стандартные методы машинного обучения. Они включают стадию обучения, когда строится соответствующая модель обнаружения, и стадию классификации.

В работе [21] провели анализ вредоносных и легитимных Android-приложений и выявили API-вызовы с условиями на параметры, которые используются ВП как минимум на 6% чаще, чем легитимными. Предлагаемая система DroidAPIMiner производит статический анализ байткода приложения, в результате которого формируется вектор используемого API. Далее выполняется классификация с использованием одного из известных классификаторов: ID5 DT, C4.5 DT, KNN, SVM. Классификатор KNN показал наилучшие результаты в проведенном исследовании. Авторы также использовали векторы запрашиваемых привилегий для классификации, но модель привилегий оказалась недостаточно работостособной.

Таблица 2. Сравнение методов обнаружения мобильных ВП (II)

Метод	Накладные расходы на вычисление	Обнаруживаемая вредоносная активность	Набор данных для тестирования	Ошибка 2-го рода	Ошибка 1-го рода	Скорость работы
Статический анализ						
PiOS [7]	НИ	Утечки данных	825 прил. из рынка Apple's iTunes; 582 приложений из рынка Cydia	НИ	НИ	НИ
Social-AV [8]	НИ	Любые ВП, заданные сигнатурами	809МБ данных (текст, изображения, pdf, и др.)	5,25% для префильтрации	НИ	НИ
RiskRanker [9]	НИ	Известные root-эксплоиты, использование premium-сервисов	118318 прил. из 15 рынков; 133 образца из репозитария Contagio	0%	9%	≈1,03 с на прил.
AppProfiler [10]	НИ	ВП, использующие опасные API-вызовы	≈80000 приложений из Android Market, Contagio	16–23%	10–15%	НИ

Продолжение 1 табл. 2

Метод	Накладные расходы на вычисление	Обнаруживаемая вредоносная активность	Набор данных для тестирования	Ошибка 2-го рода	Ошибка 1-го рода	Скорость работы
AppIntent [11]	НИ	Утечки данных	1000 прил. из Google Play; 750 вредоносных прил.	≈50% для статич., 0% для динамич.	НИ	Статич.: 3,3 м.; динамич.: 5–134 м на прил.
Динамический анализ						
TaintDroid [12]	время: +3–29% память: +3,5%	Утечки данных	30 популярных приложений из Android Market	НИ	НИ	–
SmartDroid [13]	НИ	Опасное поведение: возможные утечки данных, непреднамеренная отправка SMS/MMS	19 прил. из репозиториев Antiy и Contagio	НИ	НИ	≈37,63 с на прил.
DroidRanger [3]	НИ	Любые ВП с известным профилем поведения; в частности: отправка SMS на платные номера, отслеживание SMS	204040 легитимных прил.; Geinimi, ADRD, Pjapps, Bgserv, DroidDream, zHash, BaseBridge, DroidDreamLight, Zsone, jSMSHider	0%	4.2%	≈0,08 с на прил. в обнаружении по отпечаткам
VetDroid [14]	время +32,294% память +14,110%	Утечки данных	1249 прил. из Android Market; GGTracker, SMSReplicator, TapSnake ВП	–	–	Конфигурируется
AppsPlayground [17]	НИ	Утечки данных, использование premium-сервисов, root-эксплоиты	3968 прил. из Android Market; ВП: FakePlayer, DroidDream, DroidKungFu	–	–	НИ
Анализ привилегий						
Kirin [19]	НИ, легковесный	Утечки данных, отключение антивирусных приложений, SMS-спам, SMS-каналы управления устройством	311 прил. из Android Market	1,6%	НИ	НИ, очень быстрый
WHYPER [20]	НИ	Утечки конфиденциальных данных: чтение контактов, календаря; нелегитимная запись аудио	16001 прил. из Google Play Store	НИ	НИ	НИ

Продолжение 2 табл. 2

Метод	Накладные расходы на вычисление	Обнаруживаемая вредоносная активность	Набор данных для тестирования	Ошибка 2-го рода	Ошибка 1-го рода	Скорость работы
Машинное обучение						
DroidAPIMiner [21]	НИ	Любое ВП, использующее API, чаще встречающееся во вредоносных образцах	16000 прил. из Android Market и 8987 вредоносных прил.	2,2%	1%	НИ
MAST [22]	НИ	Любые ВП с соответствующими значениями рассматриваемых признаков: опасные привилегии, фильтры назначений, и др.	36710 прил. из SoftAndroid, Ndoo, Anzhi	НИ	НИ	0,45 с на прил.
Машинное обучение [23]	Память: 7027–7287 КБ; процент (обучение): +13 – 14,5%; (неактивный): +0,0 – 1,72%	Утечки данных (само-обновляющиеся ВП)	10 созданных вручную ВП; 5 вредоносных прил., зараженных PJApps, Geinimi, DroidKungFu-B; популярные легитимные прил.	13%	0%	НИ
Dendroid [24]	НИ	Любые ВП, относящиеся к семействам ВП, использованным на стадии обучения	вредоносные образцы из проекта Android Malware Genome	5,74%	НИ	НИ
Мониторинг уровня заряда аккумулятора						
Мониторинг уровня заряда аккумулятора [26]	НИ	ВП с предопределенными сигнатурами потребления энергии	WiFifaker, Cabir, Mabir, Commwarior, Lasco	1 – 3%	0 – 5%	–
VirusMeter [27]	время +1,5% (линейная регрессия)	ВП, выполняющие действия, которые могут существенно разряжать аккумулятор	FlexiSPY, Cabir; 100 безвредных образцов	5 – 22%	1,4 – 27,4%	–
Использование облачных вычислений						
Paranoid Android [29]	энергия: +15–30%	Методы обнаружения не полностью специфицированы. Авторы рассматривают ВП с известными сигнатурами, утечки данных и различные эксплоиты	Данные от пользователей телефонов	НИ	НИ	НИ

Окончание табл. 2

Метод	Накладные расходы на вычисление	Обнаруживаемая вредоносная активность	Набор данных для тестирования	Ошибка 2-го рода	Ошибка 1-го рода	Скорость работы
Crowdroid [30]	НИ	Любые ВП, использующие API, которые могут быть отделены от API, используемого легитимными приложениями в кластеризации	60 созданных вручную прил.; PJApps, HongTouTou	0% для созданных вручную	0% для созданных вручную, 25% для HongTouTou	НИ

* Можно отметить, что статические методы имеют большую долю ложных срабатываний по сравнению с другими методами обнаружения. Большинство методов нацелено на обнаружение утечек данных или заданных вредоносных приложений.

Система ранжирования приложений MAST [22] предназначена для выделения из приложений, предоставляемых рынком, тех, которые должны быть тщательно проанализированы в первую очередь. Предлагаемый анализ является легковесным. Формируется вектор признаков приложения: присутствие некоторых запрашиваемых привилегий, фильтры назначений, присутствие нативного кода, наличие внутренних zip-архивов. Затем для подсчета расстояний (сходства) между анализируемым приложением и предопределенной группой вредоносных приложений используется Анализ Множественных Соответствий (Multiple Correspondence Analysis). Общая оценка опасности приложения выставляется по совокупности этих расстояний.

В [23] предлагается подход к обнаружению ВП, основанный на анализе трафика. Метод нацелен на обнаружение самообновляющихся приложений, проявляющих вредоносную активность спустя некоторое время после установки, после обновления или динамической загрузки скомпилированного кода. Наблюдаемое поведение системы может быть классифицировано как нормальное или аномальное на основе подсчитанной модели статистики нормального трафика. Для классификации использовались алгоритмы Decision Table и REPTree, поскольку они показали наилучшие результаты. Если последовательность состояний приложения классифицируется как аномальная, анализируемое приложение считается вредоносным.

Dendroid [24] анализирует приложения на основе структуры их кода. Средство Androguard [25] использовалось для извлечения кодовых структур ВП, каждая из которых соответствует методу одного из классов приложения. Для каждого семейства ВП формируется вектор признаков, его компоненты соответствуют мере TF-IDF определенной кодовой структуре. Чем более характерной является кодовая структура для семейства, тем больше соответствующее значение в векторе. Для анализа приложения система сначала строит для него такой вектор, а затем вычисляет косинусный коэффициент сходства с каждым из известных классификатору семейств ВП. Метод может использоваться для определения семейства анализируемого ВП.

Глядя на перечисленные выше методы, можно отметить, что методы машинного обучения в применении к обнаружению мобильных ВП развивались в последние годы: от использования простых признаков, таких как присутствие некоторых привилегий или API-вызовов, до признаков, учитывающих структуру приложения.

Мониторинг уровня заряда аккумулятора

Система, зараженная вредоносным приложением, часто потребляет больше энергии по сравнению с оригинальным состоянием. Дополнительное потребление энергии может стать существенным, принимая во внимание ограниченность энергоресурсов некоторых мобильных устройств.

В [26] и др. предложено средство обнаружения ВП, разряжающих аккумулятор мобильного устройства. Средство состоит из двух компонентов: журнала уровня заряда аккумулятора и анализатора данных. В функции журнала уровня заряда аккумулятора входит наблюдение за потреблением энергии, запечатление текущих состояний и сбор истории потребления энергии. Анализатор данных используется для извлечения уникального шаблона из истории, что необходимо для генерации сигнатуры потребления энергии. Эта сигнатура сопоставляется с базой известных сигнатур на основе $\chi 2$-расстояния.

VirusMeter [27] отслеживает использование энергоресурсов на мобильном устройстве. На основании заданной модели энергопотребления, VirusMeter подсчитывает сколько энергии должно было уйти с учетом запущенных сервисов, а затем сравнивает это значение с наблюдаемым. Если разница превышает заданный порог, генерируется сигнал тревоги. VirusMeter строит пользовательские модели энергопотребления, в которых количество потребляемой энергии подсчитывается на основе стандартных пользовательских действий, их длительности, а также факторов окружения. Совокупность этих факторов используется в трех разных подходах: линейной регрессии, нейронных сетях и деревьях решений. По результатам экспериментов наиболее высокий уровень обнаружения у подхода, основанного на нейронных сетях.

Авторы [28] исследовали потребление энергии на современных смартфонах. Они провели несколько экспериментов с ВП, а также со сгенерированной дополнительной нагрузкой, указывающих на то, что дополнительное потребление энергии незначительно и сравнимо с шумом, возникающим из-за непредсказуемых действий пользователя и взаимодействий среды. Авторы заключают, что методы, основанные на мониторинге уровня заряда аккумулятора непригодны для современных Android-смартфонов.

Использование облачных вычислений

Облачные вычисления используются для преодоления проблемы ограниченности ресурсов мобильных устройств, обеспечивая эффективные схемы организации передачи данных с устройства на машины для анализа.

В системе Paranoid Android (PA) [29] на устройстве трассировщик записывает всю информацию, необходимую для воспроизведения выполнения приложения, и передает ее в облачные ресурсы по зашифрованному каналу. В облаке воспроизводится выполнение мобильной программы с возможностью использования нескольких проверок безопасности одновременно. Для снижения накладных расходов на передачу данных PA использует прокси-соединение, перехватывающее и временно сохраняющее входящий трафик перед его получением программой для воспроизведения.

Система Crowdroid [30] использует поведенческий подход, она состоит из клиентского приложения, устанавливаемого на устройство, и центрального сервера. Клиентские приложения посылают данные о системных вызовах, используемых наблюдаемым приложением. Сервер формирует по ним вектор признаков (для каждого приложения). После сбора достаточного количества данных проводится кластеризация (в частности, используется алгоритм k средних). Это позволяет разделить проявляемое поведение на

легитимное и вредоносное. Используя эту информацию, Crowdroid может извещать пользователей о найденном ВП.

Методы противодействия вредоносным приложениям

В данном разделе описаны методы противодействия ВП, предложенные для мобильной среды. Такие методы предназначены для предотвращения ущерба от вредоносных приложений. Табл. 3 содержит рассмотренные в обзоре параметры методов противодействия.

Критерии сравнения

1. Вредоносные действия, предотвращаемые системой.
2. Компоненты ОС, модифицируемые системой.
3. Тестовый набор данных.
4. Накладные расходы на вычисления.
5. Поддерживаемая операционная система.

Предотвращение утечек данных

AppFence [16] реализует защиту пользовательских данных от получения вредоносными приложениями, комбинируя техники оттенения данных и блокирования утечек. Оттенение данных предотвращает доступ приложений к конфиденциальной информации, заменяя ее на сгенерированные данные или пустую строку. Блокирование утечек конфиденциальных данных по сети основано на системе TaintDroid. Однако система вносит побочные эффекты в исполнение приложений, проявившиеся в 44% тестов.

Система TISSA обеспечивает защищенный режим работы приложений [31]. Пользователь может ограничить доступ к конфиденциальным источникам для каждого из приложений. В TISSA доступны различные уровни защиты: анонимизированные данные, поддельные данные и пустые данные, каждый из этих видов используется вместо настоящих данных в защищаемом приложении.

Таблица 3. Сравнение методов противодействия мобильным ВП

Метод	Подавляемая вредоносная активность	Модифицированные компоненты системы	Набор данных для тестирования	Накладные расходы на вычисления	Поддерживаемая ОС
Предотвращение утечек данных					
AppFence [16]	Утечки данных	Менеджер ресурсов, компоненты файловой системы	50 прил. из Android market	НИ	Android
TISSA [31]	Утечки данных	Компоненты, располагающие конфиденциальными данными	24 бесплатных прил. из Android market	НИ	Android
ProtectMyPrivacy [32]	Утечки данных	Различные методы, используемые для доступа к конфиденциальной информации, Springboard	225685 приложений	НИ	iOS

Продолжение табл. 3

Метод	Подавляемая вредоносная активность	Модифицированные компоненты системы	Набор данных для тестирования	Накладные расходы на вычисления	Поддерживаемая ОС
Переупаковывание приложений					
Aurasium [33]	Утечки данных, использование платных сервисов, root-эксплоиты	Нет	3491 прил. из рынка lisvid.com, 1260 вредоносных прил.	Время +14-35%	Android
Переупаковывание приложений [35]	Утечки данных	Нет	100 прил. из WP Store	НИ	Windows Phone
Принудительный контроль доступа					
FlaskDroid [37]	root-эксплоиты, утечки данных, сенсорные ВП, атаки на доверие и атаки со сговором	Package Manager, Сервисы, Content Provider	Синтетические тесты, WhatsApp, Facebook, эксплоит mempodroid	Базовые настройки: время +37% память +1.2%	Android
XManDroid [5]	Атаки на доверие и атаки со сговором	Package Manager, Reference Monitor	50 сторонних приложений, созданные вручную ВП	Время на ICC +50%, время на доступ для чтения: +2.4% - 48%	Android

* Из представленных в таблице методов только методы переупаковывания приложений не требуют изменений компонентов операционной системы.

ProtectMyPrivacy – система безопасности, разработанная для телефонов iOS, на которых был выполнен jailbreak [32]. Она предоставляет функциональность по установке ограничений безопасности для различных приложений. Исходные данные, запрашиваемые приложением, могут быть заменены поддельными. В системе предусмотрена возможность запросить рекомендованные настройки безопасности, полученные на основе собранных настроек пользователей ProtectMyPrivacy.

Переупаковывание приложений

Aurasium – это средство для переупаковывания Android-приложений, усиливающее безопасность пользователей и конфиденциальность их данных [33]. Aurasium выполняет в контексте API-вызовов множество проверок безопасности и конфиденциальности. Часть политики безопасности Aurasium реализована как нативная библиотека, кроме этого средство модифицирует java-код с помощью apktool [34]. Получающееся в итоге переупакованное приложение показывает пользователю сообщения при встрече опасных API-вызовов.

Авторы [35] предлагают сделать обязательными запросы на доступ к конфиденциальным пользовательским данным в приложениях для Windows Phone. Анализ включает построение графа потока управления и добавление проверок непосредственно перед доступом к критическим ресурсам. Предполагается использование рынками приложений перед размещением новых приложений.

Принудительный контроль доступа (ПКД)

ПКД со специально определенными политиками безопасности также используется для противодействия мобильным ВП. Система XManDroid [6, 37] реализует усиление безопасности на уровне связующего программного обеспечения, применяя правила безопасности к анализируемому межкомпонентному взаимодействию (ICC). Решения безопасности кэшируются и снова используются в случае того же ICC. Правила безопасности основаны на опасных комбинациях привилегий, которые могут привести к атакам на доверие и атакам со сговором. Для реализации ПКД на уровне ядра используется TOMOYO Linux v 1.8. Для сообщенной работы на двух уровнях авторами была написана нативная библиотека с доступом к интерфейсам TOMOYO.

Другая архитектура безопасности, предложенная авторами [6], – это FlaskDroid [5], созданный на основе SE Android [36], расширяя его на уровне связующего программного обеспечения. ПКД на уровнях ядра и связующего программного обеспечения динамически синхронизируется. FlaskDroid приспособлен для одновременной поддержки нескольких политик безопасности различных заинтересованных сторон (например, создатель приложений, владелец устройства), согласуя их в случае необходимости.

Заключение

В данной работе представлен обзор методов обнаружения и противодействия мобильным ВП. Многие из рассмотренных методов обнаружения не адаптированы для владельцев устройств, напрямую может использоваться только 55% методов. Системы обнаружения, основанные на анализе запрашиваемых привилегий, являются наиболее простыми и могут быть легко внедрены на любом из мобильных устройств. Однако, они не дают достаточно точных результатов для их использования отдельно от других средств обнаружения ВП.

Методы мониторинга уровня заряда аккумулятора не применимы к современному состоянию мобильной среды [28]. Тем не менее, несмотря на возросшие ресурсы мобильных устройств, методы организации облачных вычислений остаются значимыми для построения тщательного анализа ВП.

Методы машинного обучения оказались достаточно эффективными в применении к задаче обнаружения ВП. Среди их недостатков нужно отметить существенную зависимость от набора данных для обучения, модели обучения должны постоянно обновляться в связи с появлением новых ВП. Кроме этого, такие методы не дают точных результатов, поэтому они больше подходят для ранжирования приложений.

Методы статического анализа предоставляют ценную информацию, но без динамических проверок использование таких методов может привести к большому количеству ложных срабатываний. Наиболее тщательный динамический анализ в то же время является наиболее ресурсоемким подходом. Большинство рассмотренных динамических методов предназначены для специалистов в области безопасности, что значительно сокращает возможность их распространения. Можно также отметить, что имеется тенденция к созданию усовершенствованных средств динамического анализа, принимающих во внимание графический интерфейс анализируемого приложения и его семантику для достижения более полного покрытия кода. Комбинация техник статического, динамического анализа, а также машинного обучения может быть использована для достижения лучших результатов в области обнаружения мобильного ВП в будущем.

Системы противодействия мобильным ВП покрывают гораздо больше вредоносной активности. Методы противодействия, встроенные в работающие системы, требуют модификации компонентов ОС, что значительно затрудняет их распространение. Методы переупаковывания не требуют модификаций в ОС, но они во многом зависят от средств декомпиляции, которые не всегда дают точное представление анализируемого приложения. Методы, реализующие ПКД, имеют наиболее широкое покрытие предотвращаемых вредоносных действий.

Автоматические проверки новых приложений на рынках совершенствуются с течением времени, тем не менее, не гарантируют безопасность предоставляемых приложений. Проверки с помощью установленных на устройство антивирусных приложений могут дать лишь небольшое усиление безопасности. Обязательный ручной анализ мог бы существенно улучшить ситуацию, однако, это не представляется возможным для рынка Android-приложений. Другой подход – усиление безопасности мобильных операционных систем и установка систем предотвращения вредоносной активности, покрывает некоторые недостатки автоматической проверки приложений. Комбинация этих двух подходов – это весь спектр средств мобильной безопасности, используемых на сегодняшний день, и развитие обоих направлений определяет будущее мобильной безопасности.

СПИСОК ЛИТЕРАТУРЫ:

1. Gostev A. et al. Mobile Malware Evolution: An Overview, Parts 1-6 [HTML]. URL: http://securelist.com/analysis/malware-evolution-monthly/36109/mobile-malware-evolution-an-overview-part-1/ (**дата обращения** 20.04.2015).

2. Kindsight Security Labs Malware Report - Q4 2013 [PDF]. URL: http://www.tmcnet.com/tmc/whitepapers/documents/whitepapers/2014/9861-kindsight-security-labs-malware-report-q4-2013.pdf (**дата обращения** 20.04.2015).

3. Zhou Y., Wang Z., Zhou W., Jiang X., Hey You. Get Off of My Market: DetectingMalicious Apps in Official and Alternative Android Markets // 19th Annual Network & Distributed System Security Symposium. The Internet Society, 2012 [PDF].URL: http://www.internetsociety.org/hey-you-get-my-market-detecting-malicious-apps-official-and-alternative-android-markets (**дата обращения** 20.04.2015).

4. Ramu S. Mobile Malware Evolution, Detection and Defense [PDF].
URL: http://blogs.ubc.ca/computersecurity/files/2012/04/Sramu_EECE572_SurveyPaper-SrikanthRamu.pdf (**дата обращения** 20.04.2015).

5. Bugiel S., Heuser S., Sadeghi A. Flexible and Fine-grained Mandatory Access Control on Android for Diverse Security and Privacy Policies //Proceedings of the 22Nd USENIX Conference on Security. Berkeley, CA, USA: USENIX Association, 2013. P. 131–146.

6. Bugiel S., Davi L., Dmitrienko A., Fischer T., Sadeghi A. XManDroid: A New Android Evolution to Mitigate Privilege Escalation Attacks [PDF]. URL: https://www.trust.informatik.tu-darmstadt.de/fileadmin/user_upload/Group_TRUST/PubsPDF/xmandroid.pdf (**дата обращения** 20.04.2015).

7. Egele M., Kruegel C., Kirda E., Vigna G. PiOS: Detecting Privacy Leaks in iOS Applications // 18th Annual Network & Distributed System Security Symposium. The Internet Society, 2011 [PDF]. URL: http://www.internetsociety.org/doc/pios-detecting-privacy-leaks-ios-applications-paper (**дата обращения** 20.04.2015).

8. Yang L., Ganapathy V., Iftode L. Enhancing Mobile Malware Detection with Social Collaboration //SocialCom/PASSAT. IEEE, 2011. P. 572–576.

9. Grace M., Zhou Y., Zhang Q., Zou S., Jiang X. RiskRanker: Scalable and Accurate Zero-day Android Malware Detection // Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services. NY, USA: ACM, 2012. P. 281–294.

10. Rosen S., Qian Z., Morely Z. Mao AppProfiler: A Flexible Method of Exposing Privacy-related Behavior in Android Applications to End Users //Proceedings of the Third ACM Conference on Data and Application Security and Privacy. NY, USA: ACM, 2013. P. 221–232.

11. Yang Z., Yang M., Zhang Y., Gu G., P. Ning, X. Sean Wang AppIntent: Analyzing Sensitive Data Transmission in Android for Privacy Leakage Detection // Proceedings of the 2013 ACM SIGSAC conference on Computer & Communications Security. NY, USA: ACM, 2013. P. 1043–1054.

12. Enck W., Gilbert P., Chun B., Cox L., Jung J., McDaniel P., Sheth A. TaintDroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smartphones // Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation. Berkeley, CA, USA: USENIX Association, 2010. P. 1–6.

13. Zheng C., Zhu S., Dai S., Gu G., Gong X., Han X., Zou W. SmartDroid: An Automatic System for Revealing UI-based Trigger Conditions in Android Applications // Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices. NY, USA: ACM, 2012. P. 93–104.
14. Zhang Y., Yang M., Xu B., Yang Z., Gu G., Ning P., Sean Wang X., Zang B. Vetting Undesirable Behaviors in Android Apps with Permission Use Analysis // Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security. NY, USA: ACM, 2013. P. 611–622.
15. DroidBox: Android Application Sandbox [HTML] URL: http://code.google.com/p/droidbox/ (**дата обращения** 20.04.2015).
16. Hornyack P., Han S., Jung J., Schechter S., Wetherall D. These Aren'T the Droids You'Re Looking for: Retrofitting Android to Protect Data from Imperious Applications // Proceedings of the 18th ACM Conference on Computer and Communications Security. NY, USA: ACM, 2011. P. 639–652.
17. Rastogi V., Chen Y., Enck W. AppsPlayground: Automatic Security Analysis of Smartphone Applications // Proceedings of the Third ACM Conference on Data and Application Security and Privacy. NY, USA: ACM, 2013. P. 209–220.
18. UI/Application Exerciser Monkey [HTML].URL: http://developer.android.com/tools/help/monkey.html (**дата обращения** 20.04.2015).
19. Enck W., Ongtang M., McDaniel P. On Lightweight Mobile Phone Application Certification // Proceedings of the 16th ACM Conference on Computer and Communications Security. NY, USA: ACM, 2009. P. 235–245.
20. Pandita R., Xiao X., Yang W., Enck W., Xie T. WHYPER: Towards Automating Risk Assessment of Mobile Applications // Proceedings of the 22nd USENIX Security Symposium. Berkeley, CA, USA: USENIX Association, 2013. P. 527-542.
21. Aafer Y., Du W., Yin H. DroidAPIMiner: Mining API-Level Features for Robust Malware Detection in Android // Security and Privacy in Communication Networks, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, volume 127. Springer International Publishing, 2013. P. 86–103.
22. Chakradeo S., Reaves B., Traynor P., Enck W. MAST: Triage for Market-scale Mobile Malware Analysis // Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks. NY, USA: ACM, 2013. P. 13–24.
23. Shabtai A., Tenenboim-Chekina L., Mimran D., Rokach L., Shapira B., Elovici Y. Mobile malware detection through analysis of deviations in application network behavior // Computers & Security. 2014. volume 43. P. 1–18.
24. Suarez-Tangil G., Tapiador J., Peris-Lopez P., Blasco J. Dendroid: A text mining approach to analyzing and classifying code structures in Android malware families // Expert Systems with Applications. 2014. volume 41. P. 1104 – 1117.
25. Androguard reverse engineering tool [HTML]. URL: http://code.google.com/p/androguard/ (**дата обращения** 20.04.2015).
26. Kim H., Smith J. and Shin K. Detecting Energy-greedy Anomalies and Mobile Malware Variants // Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services. NY, USA: ACM, 2008. P. 239–252.
27. Liu L., Yan G., Zhang X., Chen S. VirusMeter: Preventing Your Cellphone from Spies // Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection. Berlin, Heidelberg: Springer-Verlag, 2009. P. 244–264.
28. Hoffmann J., Neumann S., Holz T. Mobile Malware Detection Based on Energy Fingerprints – A Dead End? // Research in Attacks, Intrusions, and Defenses, Lecture Notes in Computer Science, volume 8145. Springer Berlin Heidelberg, 2013. P. 348–368.
29. Portokalidis G., Homburg P., Anagnostakis K., Bos H. Paranoid Android: Versatile Protection for Smartphones // Proceedings of the 26th Annual Computer Security Applications Conference. NY, USA: ACM, 2010. P. 347–356.
30. Burguera I., Zurutuza U., Nadjm-Tehrani S. Crowdroid: Behavior-based Malware Detection System for Android // Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices. NY, USA: ACM, 2011. P. 15–26.
31. Zhou Y., Zhang X., Jiang X., Freeh V. Taming Information-stealing Smartphone Applications (on Android) // Proceedings of the 4th International Conference on Trust and Trustworthy Computing. Berlin, Heidelberg: Springer-Verlag, 2011. P. 93–107.
32. Agarwal Y., Hall M. ProtectMyPrivacy: Detecting and Mitigating Privacy Leaks on iOS Devices Using Crowdsourcing // Proceedings of the 11th Annual International Conference on Mobile Systems, Applications, and Services. NY, USA: ACM, 2013. P. 97–110.
33. Xu R., Saïdi H., Anderson R. Aurasium: Practical Policy Enforcement for Android Applications // Proceedings of the 21st USENIX Conference on Security Symposium. Berkeley, CA, USA: USENIX Association, 2012. P. 27–27.
34. Android apktool: A tool for reengineering Android apk files [HTML]. URL: code.google.com/p/android-apktool/ (**дата обращения** 20.04.2015).
35. Livshits B., Jung J. Automatic Mediation of Privacy-Sensitive Resource Access in Smartphone Applications. // Proceedings of the 22nd USENIX Security Symposium. Berkeley, CA, USA: USENIX Association, 2013. P. 113–130.
36. Smalley S., Craig R. Security Enhanced (SE) Android: Bringing Flexible MAC to Android // 20th Annual Network & Distributed System Security Symposium. The Internet Society, 2013 [PDF]. URL: http://internetsociety.org/doc/security-enhanced-se-android-bringing-flexible-mac-android (**дата обращения** 20.04.2015).
37. Bugiel S., Davi L., Dmitrienko A., Fischer T., Sadeghi A., Shastry B. Towards Taming Privilege-Escalation Attacks on Android // 19th Annual Network & Distributed System Security Symposium. The Internet Society, 2012 [PDF]. URL: http://www.internetsociety.org/towards-taming-privilege-escalation-attacks-android (**дата обращения** 20.04.2015).

REFERENCES:

1. Gostev A. et al. Mobile Malware Evolution: An Overview, Parts 1-6 [HTML]. URL: http://securelist.com/analysis/malware-evolution-monthly/36109/mobile-malware-evolution-an-overview-part-1/ (date of access: 20.04.2015).
2. Kindsight Security Labs Malware Report - Q4 2013 [PDF]. URL: http://www.tmcnet.com/tmc/white-papers/documents/whitepapers/2014/9861-kindsight-security-labs-malware-report-q4-2013.pdf (date of access: 20.04.2015).
3. Zhou Y., Wang Z., Zhou W., Jiang X., Hey You. Get Off of My Market: DetectingMalicious Apps in Official and Alternative Android Markets // 19th Annual Network & Distributed System Security Symposium. The Internet Society, 2012 [PDF].URL: http://www.internetsociety.org/hey-you-get-my-market-detecting-malicious-apps-official-and-alternative-android-markets (date of access: 20.04.2015).
4. Ramu S. Mobile Malware Evolution, Detection and Defense [PDF].
URL: http://blogs.ubc.ca/computersecurity/files/2012/04/Sramu_EECE572_SurveyPaper-SrikanthRamu.pdf (date of access: 20.04.2015)..
5. Bugiel S., Heuser S., Sadeghi A. Flexible and Fine-grained Mandatory Access Control on Android for Diverse Security and Privacy Policies //Proceedings of the 22Nd USENIX Conference on Security. Berkeley, CA, USA: USENIX Association, 2013. P. 131–146.
6. Bugiel S., Davi L., Dmitrienko A., Fischer T., Sadeghi A. XManDroid: A New Android Evolution to Mitigate Privilege Escalation Attacks [PDF]. URL: https://www.trust.informatik.tu-darmstadt.de/fileadmin/user_upload/Group_TRUST/PubsPDF/xmandroid.pdf (date of access: 20.04.2015).
7. Egele M., Kruegel C., Kirda E., Vigna G. PiOS: Detecting Privacy Leaks in iOS Applications // 18th Annual Network & Distributed System Security Symposium. The Internet Society, 2011 [PDF]. URL: http://www.internetsociety.org/doc/pios-detecting-privacy-leaks-ios-applications-paper (date of access: 20.04.2015).
8. Yang L., Ganapathy V., Iftode L. Enhancing Mobile Malware Detection with Social Collaboration //SocialCom/PASSAT. IEEE, 2011. P. 572–576.
9. Grace M., Zhou Y., Zhang Q., Zou S., Jiang X. RiskRanker: Scalable and Accurate Zero-day Android Malware Detection // Proceedings of the 10th International Conference on Mobile Systems, Applications, and Services. NY, USA: ACM, 2012. P. 281–294.
10. Rosen S., Qian Z., Morely Z. Mao AppProfiler: A Flexible Method of Exposing Privacy-related Behavior in Android Applications to End Users //Proceedings of the Third ACM Conference on Data and Application Security and Privacy. NY, USA: ACM, 2013. P. 221–232.
11. Yang Z., Yang M., Zhang Y., Gu G., P. Ning, X. Sean Wang AppIntent: Analyzing Sensitive Data Transmission in Android for Privacy Leakage Detection // Proceedings of the 2013 ACM SIGSAC conference on Computer & Communications Security. NY, USA: ACM, 2013. P. 1043–1054.
12. Enck W., Gilbert P., Chun B., Cox L., Jung J., McDaniel P., Sheth A. TaintDroid: An Information-flow Tracking System for Realtime Privacy Monitoring on Smartphones // Proceedings of the 9th USENIX Conference on Operating Systems Design and Implementation. Berkeley, CA, USA: USENIX Association, 2010. P. 1–6.
13. Zheng C., Zhu S., Dai S., Gu G., Gong X., Han X., Zou W. SmartDroid: An Automatic System for Revealing UI-based Trigger Conditions in Android Applications // Proceedings of the Second ACM Workshop on Security and Privacy in Smartphones and Mobile Devices. NY, USA: ACM, 2012. P. 93–104.
14. Zhang Y., Yang M., Xu B., Yang Z., Gu G., Ning P., Sean Wang X., Zang B. Vetting Undesirable Behaviors in Android Apps with Permission Use Analysis // Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security. NY, USA: ACM, 2013. P. 611–622.
15. DroidBox: Android Application Sandbox [HTML] URL: http://code.google.com/p/droidbox/ (date of access: 20.04.2015).
16. Hornyack P., Han S., Jung J., Schechter S., Wetherall D. These Aren'T the Droids You'Re Looking for: Retrofitting Android to Protect Data from Imperious Applications // Proceedings of the 18th ACM Conference on Computer and Communications Security. NY, USA: ACM, 2011. P. 639–652.
17. Rastogi V., Chen Y., Enck W. AppsPlayground: Automatic Security Analysis of Smartphone Applications // Proceedings of the Third ACM Conference on Data and Application Security and Privacy. NY, USA: ACM, 2013. P. 209–220.
18. UI/Application Exerciser Monkey [HTML].URL: http://developer.android.com/tools/help/monkey.html (date of access: 20.04.2015).
19. Enck W., Ongtang M., McDaniel P. On Lightweight Mobile Phone Application Certification // Proceedings of the 16th ACM Conference on Computer and Communications Security. NY, USA: ACM, 2009. P. 235–245.
20. Pandita R., Xiao X., Yang W., Enck W., Xie T. WHYPER: Towards Automating Risk Assessment of Mobile Applications // Proceedings of the 22nd USENIX Security Symposium. Berkeley, CA, USA: USENIX Association, 2013. P. 527-542.
21. Aafer Y., Du W., Yin H. DroidAPIMiner: Mining API-Level Features for Robust Malware Detection in Android // Security and Privacy in Communication Networks, Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, volume 127. Springer International Publishing, 2013. P. 86–103.
22. Chakradeo S., Reaves B., Traynor P., Enck W. MAST: Triage for Market-scale Mobile Malware Analysis // Proceedings of the Sixth ACM Conference on Security and Privacy in Wireless and Mobile Networks. NY, USA: ACM, 2013. P. 13–24.
23. Shabtai A., Tenenboim-Chekina L., Mimran D., Rokach L., Shapira B., Elovici Y. Mobile malware detection through analysis of deviations in application network behavior // Computers & Security. 2014. volume 43. P. 1–18.

24. Suarez-Tangil G., Tapiador J., Peris-Lopez P., Blasco J. Dendroid: A text mining approach to analyzing and classifying code structures in Android malware families // Expert Systems with Applications. 2014. volume 41. P. 1104 – 1117.
25. Androguard reverse engineering tool [HTML]. URL: http://code.google.com/p/androguard/ (date of access: 20.04.2015).
26. Kim H., Smith J. and Shin K. Detecting Energy-greedy Anomalies and Mobile Malware Variants // Proceedings of the 6th International Conference on Mobile Systems, Applications, and Services. NY, USA: ACM, 2008. P. 239–252.
27. Liu L., Yan G., Zhang X., Chen S. VirusMeter: Preventing Your Cellphone from Spies // Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection. Berlin, Heidelberg: Springer-Verlag, 2009. P. 244–264.
28. Hoffmann J., Neumann S., Holz T. Mobile Malware Detection Based on Energy Fingerprints – A Dead End? // Research in Attacks, Intrusions, and Defenses, Lecture Notes in Computer Science, volume 8145. Springer Berlin Heidelberg, 2013. P. 348–368.
29. Portokalidis G., Homburg P., Anagnostakis K., Bos H. Paranoid Android: Versatile Protection for Smartphones // Proceedings of the 26th Annual Computer Security Applications Conference. NY, USA: ACM, 2010. P. 347–356.
30. Burguera I., Zurutuza U., Nadjm-Tehrani S. Crowdroid: Behavior-based Malware Detection System for Android // Proceedings of the 1st ACM Workshop on Security and Privacy in Smartphones and Mobile Devices. NY, USA: ACM, 2011. P. 15–26.
31. Zhou Y., Zhang X., Jiang X., Freeh V. Taming Information-stealing Smartphone Applications (on Android) // Proceedings of the 4th International Conference on Trust and Trustworthy Computing. Berlin, Heidelberg: Springer-Verlag, 2011. P. 93–107.
32. Agarwal Y., Hall M. ProtectMyPrivacy: Detecting and Mitigating Privacy Leaks on iOS Devices Using Crowdsourcing // Proceedings of the 11th Annual International Conference on Mobile Systems, Applications, and Services. NY, USA: ACM, 2013. P. 97–110.
33. Xu R., Saïdi H., Anderson R. Aurasium: Practical Policy Enforcement for Android Applications // Proceedings of the 21st USENIX Conference on Security Symposium. Berkeley, CA, USA: USENIX Association, 2012. P. 27–27.
34. Android apktool: A tool for reengineering Android apk files [HTML]. URL: code.google.com/p/android-apktool/ (date of access: 20.04.2015).
35. Livshits B., Jung J. Automatic Mediation of Privacy-Sensitive Resource Access in Smartphone Applications. // Proceedings of the 22nd USENIX Security Symposium. Berkeley, CA, USA: USENIX Association, 2013. P. 113–130.
36. Smalley S., Craig R. Security Enhanced (SE) Android: Bringing Flexible MAC to Android // 20th Annual Network & Distributed System Security Symposium. The Internet Society, 2013 [PDF]. URL: http://internetsociety.org/doc/security-enhanced-se-android-bringing-flexible-mac-android (date of access: 20.04.2015).
37. Bugiel S., Davi L., Dmitrienko A., Fischer T., Sadeghi A., Shastry B. Towards Taming Privilege-Escalation Attacks on Android // 19th Annual Network & Distributed System Security Symposium. The Internet Society, 2012 [PDF]. URL: http://www.internetsociety.org/towards-taming-privilege-escalation-attacks-android (date of access: 20.04.2015).