Санкт-Петербургский государственный университет

Математическое обеспечение и администрирование информационных систем
Кафедра информатики

Лысенко Юлия Олеговна

Снятие образа Android с помощью агента

Бакалаврская работа

Научный руководитель:
ст. преп. Губанов Ю.А.

Рецензент:
ст. преп. Луцив Д.В.

Санкт-Петербург
2018

SAINT-PETERSBURG STATE UNIVERSITY

Software and Administration of Information Systems
Technology in Programming

Lysenko Yulia

Agent approach for Android data acquisition

Graduation Thesis

Scientific supervisor:
professor Gubanov Y.A

Reviewer:
assistant Luciv D.V.

Saint-Petersburg
2018

Оглавление

Введение

Компьютерная криминалистика - это процесс сбора и интерпретации электронных данных. Целью этого процесса является сохранение любых доказательств в форме, максимально близкой к исходной, путем сбора, идентификации и проверки цифровой информации с целью восстановления прошедших событий.

Исследования цифровой криминалистики имеют множество применений. Наиболее распространенное из них — подтверждение или опровержение гипотезы до начала расследования в уголовном или гражданском суде. Судебная экспертиза также может присутствовать в частном секторе. Например, во время внутренних корпоративных расследований или расследования вторжений специалист исследует характер и масштабы несанкционированного вторжения.

Технический аспект исследования разделен на несколько подразделов, в зависимости от типа задействованных цифровых устройств: компьютерная криминалистика, сетевая криминалистика, анализ судебных данных и криминалистика мобильных устройств.

1. Постановка задачи

Целью данной работы является разработка программного средства для извлечения данных с нерутированных мобильных устройств с операционной системой Android 7.0 и выше с помощью агента. Интересующие данные: список контактов, звонков, SMS-сообщений, информация из календаря, аудиозаписи, фотографии, видеозаписи и список установленных приложений. Для ее достижения были поставлены следующие задачи:

- Разработать приложение-агент для сбора требуемых данных на устройстве.
- Разработать клиент-приложение для управления агентом.
- Настроить связь между клиентом и агентом.
- Провести апробацию.

2. Обзор

2.1. Мобильная криминалистика

Термин "мобильные устройства" охватывает широкий спектр гаджетов: от мобильных телефонов, смартфонов, планшетов до GPS-устройств и КПК. Их всех объединяет то, что они могут содержать много информации о пользователях. Эти компактные устройства удобны для работы с информацией, с их помощью можно легко планировать встречи, вести электронную переписку и просматривать необходимые документы.

В настоящее время их использование широко распространено, что особенно полезно в контексте цифровой криминалистики, потому что данные, извлеченные из них, могут быть полезны в процессе расследования и существенно его облегчить. В случае, когда хозяин устройства совершил какое-то преступление, могут понадобиться специальные средства, чтобы извлечь доказательства из памяти мобильных устройств.

2.2. Android

Android - это мобильная операционная система, разработанная компанией Google на основе модифицированной версии ядра Linux и другого программного обеспечения с открытым исходным кодом, предназначенная в основном для мобильных устройств с сенсорным экраном, таких как смартфоны и планшеты. Является самой распространенной мобильной ОС в мире - по данным на февраль 2018, 74.78% устройств работают на этой ОС [7]. Последняя версия Android 8.1 Oreo (27 версия API) была представлена 5 декабря 2017.

2.3. Извлечение данных

Начиная с Android 4.0 разработчикам стала доступна функция получения данных через Android Debug Bridge (ADB) Backup. Однако резервные копии ADB могут содержать не все требуемые данные. Из-за фрагментации Android производители контролируют, что будет или

не будет частью резервной копии. В результате некоторые устройства могут сохранять истории вызовов, контакты и текстовые сообщения, в то время как у других нет к ним доступа. Например [9], Samsung, Sony и HTC часто игнорируют историю звонков, контакты и текстовые сообщения, одновременно с этим сохраняя резервные копии данных Google Chrome. Сами разработчики приложений также контролируют возможность бэкапа, в итоге большинство приложений не позволяют делать резервную копию своих данных.

После получения прав суперпользователя снятие образа устройства будет доступно базовыми командами ADB. Но у такого подхода есть существенный недостаток — этот процесс может повредить телефон, что приведет к потере всех данных. Это разрушающий метод и его стараются использовать когда другие методы недоступны.

В данной работе рассматривается извлечение данных с нерутированных устройств.

2.4. Метод агентов

Для получения дополнительных данных, не включенных в резервные копии ADB, используется метод извлечения файлов с помощью агента. С помощью специального программного обеспечения на устройство устанавливается приложение (агент). Для получения доступа к требуемым файлам агент запрашивает необходимые права доступа. Находясь в разделе пользовательских данных, доступных после получения прав, он может передать необходимую информацию на подключенный компьютер. Этот метод требует, чтобы устройство было включено, разблокировано и на нем была разрешена USB-отладки. После окончания работы агент удаляется с устройства.

2.5. Разрешения

Целью механизма разрешений является защита конфиденциальности пользователя Android. Приложения для Android должны запрашивать разрешение на доступ к конфиденциальным данным пользо-

вателя (например, контакты и SMS), а также некоторым системным функциям (например, камера). В зависимости от требуемой функции система может предоставить разрешение автоматически или может попросить пользователя одобрить запрос. Разрешения организованы в группы, которые связаны с возможностями или функциями устройства. Таким образом запросы на разрешение обрабатываются на уровне группы, а одна группа соответствует нескольким объявлениям разрешений в манифесте приложения. Например, группа SMS включает в себя как READ_SMS, так и RECEIVE_SMS. Такая группировка позволяет пользователю делать более значимые и осознанные варианты, не перегружая сложными и техническими разрешениями [11].

2.5.1. Уровни защиты

Разрешения делятся на несколько уровней безопасности, которые влияют на необходимость выполнения запросов на доступ к данным во время работы приложения. Существует три уровня защиты:

- Обычные разрешения (normal)

  Охватывают области, где приложения должны получить доступ к данным или ресурсам вне песочницы приложения, но где существует очень мало риска для конфиденциальности пользователя или для работы других приложений. Например, разрешение на установку часового пояса является нормальным разрешением. Если в манифесте приложения заявлено, что ему требуется нормальное разрешение, система автоматически предоставляет такое разрешение во время установки. Пользователь не контролирует этот уровень защиты.

- Разрешения с подписью (signature)

  Система предоставляет эти разрешения только если приложение, которое пытается использовать разрешение, подписано тем же сертификатом, что и приложение, в котором объявлено право доступа.

- Опасные разрешения (dangerous)

  Охватывают области, в которых приложение хочет получать данные или ресурсы, которые связаны с личной информацией пользователя, или потенциально повлияет на сохраненные данные пользователя или на работу других приложений. Например, способность читать контакты пользователя является опасным разрешением. Если приложение заявляет, что ему требуется опасное разрешение, пользователь должен явно его предоставить. Пока он не одобрит доступ, приложение не сможет выполнять функции, зависящие от этого разрешения. Чтобы использовать опасное разрешение, разработчик должен предложить пользователю одобрить его во время работы приложения.

2.5.2. Разрешения в последних версиях Android

- Android 6.0

  Изначально все разрешения, которые требовались приложению, запрашивались при его установке на устройство. В версии 6.0 (Marshmallow) разработчики изменили подход к этому вопросу и представили новую модель разрешений, в которой пользователи могут напрямую управлять разрешениями приложения во время его работы. Если пользователь отказывает в запросе на разрешение, при следующем запросе диалоговое окно будет содержать поле, в котором пользователь может указать, что больше не хочет видеть этот запрос [3].

- Android 7.0

  В целях повышения безопасности личных файлов, начиная с версии 7.0 каталог приложений имеет ограниченный доступ. Этот параметр предотвращает утечку метаданных приватных файлов, таких как их размер или существование [4].

- Android 8.0

До этой версии системы существовала следующая проблема, если приложение запрашивало некоторое разрешение и это разрешение было предоставлено, то система неправильно предоставляла приложению остальные разрешения из манифеста, принадлежащие той же группе. Начиная с этой версии системы приложение получает только явно запрошенные разрешения. Однако, как только пользователь одобряет доступ к данным, все последующие запросы на разрешения из этой группы предоставляются автоматически [5].

2.6. Android Debug Bridge

Android Debug Bridge (ADB) — это инструмент командной строки, который обеспечивает обмен данными между устройством Android и персональным компьютером. Эта связь чаще всего выполняется по USB-кабелю или Wi-Fi соединению. Команды ADB позволяют выполнять такие действия как установка и отладка приложений, обеспечивает доступ к подмножеству команд Unix-оболочки, которые может использоваться для запуска различных команд на устройстве. Эту клиент-серверную программу составляют три компонента:

Клиент, который отсылает команды. Обычно в качестве клиента используют командную строку.

Демон (ADBD), который запускает команды на устройстве. Демон запускается как фоновый процесс на каждом устройстве.

Сервер, который регулирует связь между клиентом и демоном. Сервер работает как фоновый процесс на персональном компьютере. Он представляет собой один гигантский цикл мультиплексирования, целью которого является организация обмена пакетами данных между клиентами, сервисами и устройствами. Сервисы - команды, посылаемые клиентом на сервер.

Основная цель сервера - распознавать, когда какое-либо устройство было подключено или отключено от USB-порта, или когда был включен или выключен эмулятор Android.

ADB является частью Android Software Development Kit (SDK). Чтобы использовать ADB для работы с устройством, подключенным через USB, необходимо включить отладку USB в настройках системы устройства в разделе «Параметры разработчика».

При запуске ADB клиента, он запускает сервер, если тот не был запущен раньше. Когда сервер запускается, он привязывается к локальному TCP-порту 5037 и прослушивает команды, отправленные от ADB-клиентов - все клиенты используют порт 5037 для связи с сервером. Затем сервер устанавливает соединения со всеми присоединенными устройствами. Он находит эмуляторы, сканируя нечетные порты в диапазоне от 5555 до 5585 (диапазон для первых 16 эмуляторов). Когда сервер находит ADB-демона, он устанавливает соединение с этим портом.

2.7. Существующие решения

В данном разделе описаны некоторые известные существующие на данный момент решения.

2.7.1. Magnet Acquire (Magnet Forensics)

Это бесплатный инструмент для судебной экспертизы, который используется для извлечения данных так же и с Android устройств.

При извлечении данных с мобильного устройства Android Magnet Acquire выполняет следующие действия [8]:

1) Создает резервную копию этого устройства.

2) Устанавливает приложение-агент на устройство.

3) Использует это приложение для извлечения некоторых видов данных и копирования файлов с SD-карты (при ее наличии).

4) Возвращает все извлеченные данные и записывает в один файл.

Android Magnet Acquire принадлежит компании Magnet Forensics и не находится в открытом доступе.

2.7.2. Oxygen Forensic Suite

Oxygen Forensic Suite — программное обеспечение для мобильной судебной экспертизы, используемое для логического анализа смартфонов и других мобильных устройств, разработанных Oxygen Software. Данный продукт может извлекать информацию об устройстве, контакты, события календаря, SMS-сообщения, журналы событий и файлы. Кроме того, поставщик утверждает, что пакет может извлекать метаданные, связанные с этими файлами. Подробная схема работы программы доступна по ссылке [10].

2.7.3. Dr.Fone

Dr.Fone — продукт компании Wondershare Software, который служит для восстановления данных, системы, передачи данных, резервного копирование и восстановление файлов и т.д. На компьютер устанавливается программа, которая устанавливает агент на подключенное устройство и производит с ним запрошенные действия. Недостатком Dr.Fone является то, что для работы с устройством необходимо рутирование устройста.

3. Архитектура

В целом реализуемая система состоит из двух основных объектов — ADB-клиента (далее - клиента), который необходим для автоматизации работы с агентом и приёма данных с устройства, а также агента, который служит для сбора и отправки данных на ПК. Для организации взаимодействия частей системы используется ADB-сервер, который перенаправляет команды с клиента на устройство. Далее будут подробно описаны основные компоненты системы.

3.1. Схема взаимодействия клиента и ADB-сервера

Данный раздел представляет схему взаимодействия клиента с ADB-сервером. В общем виде, взаимодействие выглядит следующим образом: клиент отправляет команды ADB-серверу, сервер обрабатывает команды, отвечает клиенту и передает эту команду ADB-демону на устройстве. Опишем этот процесс более подробно.

Для начала работы необходимо запустить ADB-сервер на компьютере. Сразу после запуска клиент отправляет операционной системе команду для запуска ADB-сервера. Тот в свою очередь подключается к порту 5037 и готов принимать команды от клиента.

ADB-сервер умеет распознавать 2 типа подключенных устройств — Andoid-эмуляторы, запущенные на компьютере, и физические устройства, подключенные к USB-порту. В рамках данной работы исследовался метод извлечения данных с физического устройства, поэтому после включения сервера клиент отправляет запрос для того, чтобы сервер выбрал для работы устройство, присоединенное с помощью USB. Сервер подключается у устройству со статусом ”ONLINE” — то есть сервер не только определил, что устройство соединено с компьютером, но также смог подключиться к ADB-демону внутри устройства. После успешного подключения сервер передает клиенту результат выполнения запроса — сообщение ”OKAY”.

Следующим действием клиента является отправление команды ”forward”. С ее помощью можно перенаправить запросы с конкретного порта на

компьютере на определенный порт на устройстве.

На следующем изображении показаны первоначальные действия клиента, не захватывающие процесс передачи команд. Из него видно, что клиент и ADB - сервер общаются через порт 5037. В то же время сервер подключен к ADB - демону, работающему на устройстве, подключенном через USB-кабель.

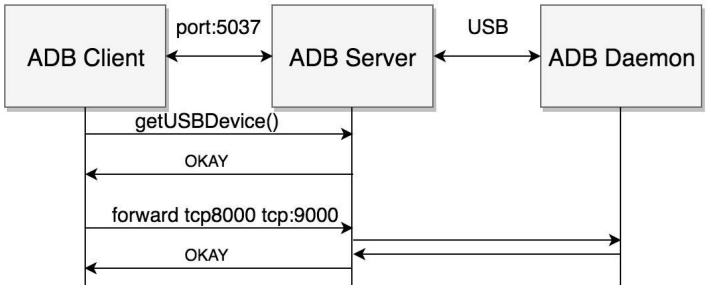

Рис. 1: Начало работы клиента с ADB-сервером

После настройки взаимодействия клиента и ADB-сервера необходимо загрузить, установить, а также запустить приложение-агент на целевом устройстве. Для этого используется ADB-демон, работающий на целевом устройстве.

В клиенте реализованы отдельные функции, которые реализуют необходимое взаимодействие клиента с ADB - сервером, который в свою очередь обеспечивает общение с устройством.

На следующей схеме изображены шаги, необходимые для запуска приложения-агента на устройстве.

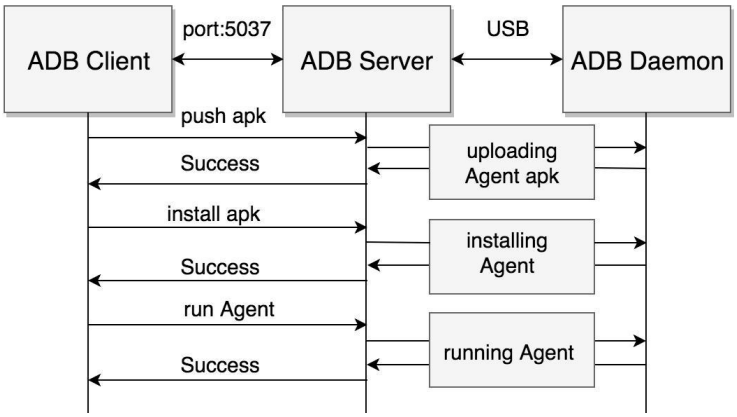

Рис. 2: Загрузка приложения

3.2. Схема работы клиента с агентом

Данный раздел представляет схему взаимодействия клиента с агентом. В общем виде взаимодействие выглядит следующим образом: клиент отправляет команды агенту, агент считывает команды, собирает запрошенные данные и отправляет их обратно клиенту. Опишем эти действия более подробно.

После того, как агент запущен на целевом устройстве, клиент начинает общаться напрямую с агентом, при этом, больше не обращаясь к серверу для передачи команд ADBD. Процесс диалога клиента с приложением изображен на следующей схеме:

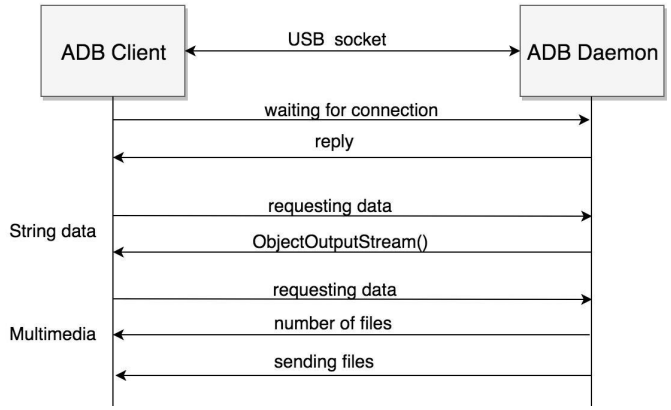

Рис. 3: Общение клиента с агентом

Как только агент начал работу, необходимо проверить, успел ли он начать прослушивать заданный порт. Стрелка ”waiting for connection” на изображении обозначает процесс проверки состояния приложения на готовность слушать команды клиента. Агенту отправляются запросы и, в случае получения запроса, он передает положительный ответ на компьютер.

После получения ответа, клиент может посылать запросы на передачу необходимых данных. В данной работе реализовано извлечение двух типов данных — текстовые данные (представлены в виде List<String>) и мультимедиа. Строковые данные передаются клиенту как объекты, для мультимедиа сначала на компьютер отправляется информация о количестве файлов, затем сами файлы.

4. Реализация

В данной главе будут описаны некоторые технические детали, а также рассмотрена более подробная реализация взаимодействия компонентов, описанных в главе 3, с внешним миром.

4.1. Изучение обмена данными между ПК и телефоном

Для разработки процесса общения клиента с ADB-сервером была изучена документация [2]. Для более полного понимания того, как происходит передача команд использовался Wireshark.

Wireshark — анализатор сетевых протоколов. Он используется для анализа и устранения неполадок в работе сети, разработки программного обеспечения и коммуникации. В частности, данный инструмент позволяет отслеживать трафик данных по USB.

Изучение проходило при использовании бесплатной пробной версии программы Dr.Fone, рассмотренной в обзоре, и мобильного телефона марки Sony. Рассматривалось действие программы до того момента, как на телефоне запрашивалось разрешение на получение root-прав. Основываясь на полученной информации, велась работа над разработкой клиента.

4.2. Реализация клиента. Взаимодействие клиента с ADB-сервером

По сути, существует два вида сервисов, отправляемых клиентом.

Host-сервисы: Эти сервисы работают на сервере ADB и поэтому не требуют общения с устройством. Пример используемого сервиса — host:transport-usb.

Local-сервисы: Эти команды либо выполняются на ADB-демоне, либо запускаются им на самом устройстве. Сервер ADB используется для мультиплексирования потоков между клиентом и процессом, запущен-

ном на ADBD. В этом случае его роль заключается в том, чтобы инициировать соединение, а затем передавать данные.

Дополнительно среди **local**-сервисов существуют **sync**-сервисы: работа в данном режиме отличается от обычного протокола ADB. Соединение остается в режиме синхронизации до его явного завершения.

В данной работе были реализована работа следующих сервисов

- host:devices — в результате возвращается список подключенных устройств.
- host:transport-usb — происходит соединение с устройством, подключенным по USB-кабелю.
- host-usb:forward:tcp:<local> tcp:<remote> — перенаправляет запросы с хост-компьютера на порт <local> на устройстве, а с устройства на порт <remote> на компьютере.
- local-сервис shell:<command> — отправляет команду в shell устройства.
- sync:send — используется для передачи установочного файла приложения на устройство.

После вызова этих сервисов сервер должен ответить "OKAY" или "FAIL". При реализации host-сервисов за основу был частично взят проект, доступный по ссылке [1].

Рассмотрим подробно использование перечисленных сервисов в данном проекте. С помощью команды host:devices клиент получает список подключенных устройств. В последней реализации происходит обращение к первому устройству из списка и далее работа ведется с ним. Далее вызывается команда forward:tcp:8000;tcp:9000, которая настраивает общение клиента и будущего агента. Данные команды не затрагивают общение с ADBD.

После того как связь с устройством установлена, необходимо подключиться к приложению-агенту, установленному на нем. Для этого будут использоваться вышеупомянутые local-сервисы.

Сперва клиент проверяет есть ли среди установленных приложений наш агент с помощью команды "shell: pm list packages| grep package_name", где package_name — полное имя приложения в памяти устройства. При отрицательном результате вызывается передается команда "shell:ls /sdcard | grep AndroidAgent" для получения информации о том, есть ли на устройстве установочный файл с приложением.

Если мы узнали, что на устройстве нет информации об агенте, далее необходимо выполнить несколько шагов для запуска агента. Загрузка apk-файла с агентом производится по средствам сервиса sync:send. Для установки приложения вызывается "shell:pm install -g /sdcard/AndroidAgent.ap где -g отвечает за то, чтобы приложению при установке были выданы все разрешения, перечисленные в манифесте приложения. Подробнее будет рассмотрено в разделе "Реализация агента".

Если же на устройстве уже был установлен агент, то клиент проверяет, запущено ли приложение в данный момент. Для этого используется команда "shell:pidof package_name". Она возвращает id приложения, если оно есть в списке запущенных, или возвращает пустую строку. Далее остается только запустить приложение: "shell:am start -n package_name/package_name.MainActivity".

Когда работа агента завершена и все необходимые данные были переданы клиенту и сохранены на ПК, вызывается команда "am force-stop package_name" для завершения работы приложения. Далее "pm uninstall package_name" удаляет приложение с устройства, а "rm -f /sdcard/And удаляет установочный файл.

4.3. Реализация клиента. Взаимодействие клиента с агентом

Далее рассмотрим процесс общения клиента непосредственно с агентом, работающем на устройстве.

После того как агент был запущен, необходимо дождаться, когда приложение будет готово принимать команды клиента. Для этого клиент с заданным временным интервалом открывает сокет-соединение и

пытается передать сообщение приложению на порт, ранее настроенный командой forward. Если в процессе подключения происходит ошибка, значит приложение еще не открыло соединение со своей стороны и не смогло принять сообщение. В случает успеха клиент получает ответное сообщение от агента.

Когда подключение настроено, агент ожидает запрос от клиента на сбор необходимых данных. Всего нас интересует 8 типов данных, которые можно условно разделить на два вида: текстовые данные и мультимедиа. Клиент поочередно запрашивает все данные и, в зависимости от типа отправляемой агентом информации, принимает ее, обрабатывает и сохраняет на компьютер.

4.4. Реализация агента

При разработке агента использовалось устройство марки Samsung с версией SDK 24 (7.0 Nougat). Так как цель работы была разработать приложение-агент для сбора данных с последних версий Android, то есть начиная с версии 6.0 (Marshmallow), была указана версия SDK 23 в качестве минимальной. Почти все интересующие данные, кроме установленных приложений, затрагивают работу с информацией, которая помечена опасным уровнем доступа. Для работы с ними необходимо запрашивать разрешение у пользователя, после чего доступны будут только те данные, доступ к которым был выдан. В данной работе была выполнена обработка всех необходимых разрешений, но производить запрос на них не требуется. Ранее было сказано, что ADB может автоматически выдать все необходимые разрешения при установке приложения.

При запуске приложения в Android вызывается стандартный метод onCreate, за прием и отправку данных отвечает класс Listener. При вызове функции onCreate создается экземпляр этого класса, который отвечает на запрос клиента. Далее агент принимает информацию о данных, которые необходимо собрать и переслать клиенту. Приложение проверяет, есть ли разрешение к доступу запрашиваемых данных и в

случае положительного ответа переходит к их сбору.

В случае, когда приходит запрос на передачу текстовых данных, таких как список звонков или SMS-сообщений, информация собирается в List<String> и сразу является готовым объектом, который потом пересылается на компьютер. Передача таких данных реализуется с помощью класса ObjectOutputStream.

Если же приходит запрос на отправку мультимедиа, сначала создается список путей к искомым файлам. Далее клиенту отправляется информация о количестве отправляемых файлов, а затем они побитово передаются на компьютер.

Для работы по сбору требуемых данных использовался Android API [6].

5. Апробация

Разработка проекта велась с использованием устройство марки Samsung с установленной версией Android 7.0. Когда проект был реализован, была произведена апробация на устройстве LG Nexus с версией Android 6.0 (SDK 23) и устройстве Xiaomi с версией Android 8.0 (SDK 27).

В результате работы на компьютере создается папка, содержание которой изображено на следующем рисунке:

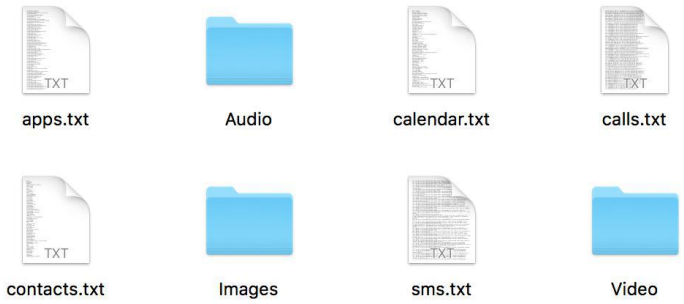

Рис. 4: Содержание папки Downloads на компьютере

Далее показан пример извлечения звонков с устройства.

Информация на телефоне о звонках разных типов:

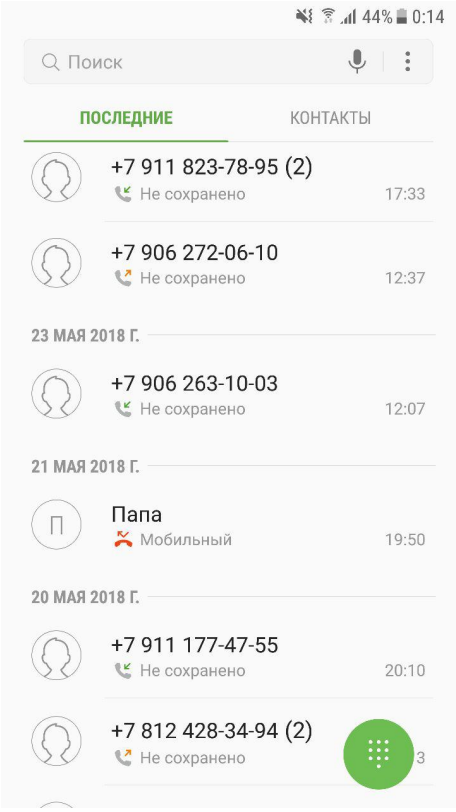

Рис. 5: Информация о звонках на телефоне

После извлечения файлов данные о звонках записываются в файл calls.txt, его содержимое изображено на следующем рисунке:

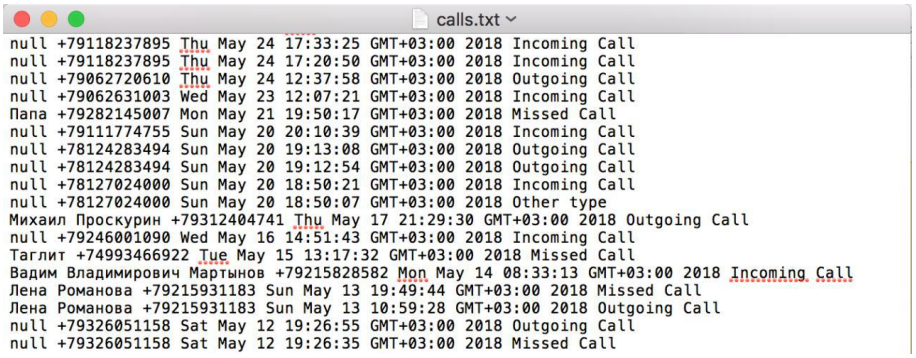

Рис. 6: Содержание файла calls.txt

Заключение

В данной работе были выполнены следующие задачи:

- Разработано приложение-агент для сбора требуемых данных на устройстве.
- Разработано клиент-приложение для управления агентом.
- Настроена связь между клиентом и агентом.
- Проведена апробация.

Список литературы

[1] URL: https://github.com/cuiyang-org/adb.

[2] ADB documentation. — URL: https://android.googlesource.com/platform/system/core/+/master/adb.

[3] Android 6.0 Changes. — URL: https://developer.android.com/about/versions/marshmallow/android-6.0-changes.html#behavior-runtime-permissions.

[4] Android 7.0 Changes. — URL: https://developer.android.com/about/versions/nougat/android-7.0-changes.html#perm.

[5] Android 8.0 Changes. — URL: https://developer.android.com/about/versions/oreo/android-8.0-changes.html#rmp.

[6] Android API. — URL: https://developer.android.google.cn/docs/.

[7] GlobalStats Statcounter. Mobile Operating System Market Share Worldwide. — URL: http://gs.statcounter.com/os-market-share/mobile/worldwide.

[8] Igor Mikhaylov Oleg Skulkin Igor Shorokhov. MOBILE FORENSICS: UFED VS MAGNET ACQUIRE. — URL: https://www.digitalforensics.com/blog/mobile-forensics-ufed-vs-magnet-acquire/.

[9] Oleg Afonin Vladimir Katalov. Mobile Forensics – Advanced Investigative Strategies. — September, 2016. — URL: https://books.google.ru/books/about/Mobile_Forensics_Advanced_Investigative.html?id=9oVcDgAAQBAJ&redir_esc=y.

[10] Oxygen Forensic Suite. — URL: http://www.oxygensoftware.ru/download/articles/MK_Getting_started.pdf.

[11] Permissions Overview. — URL: https://developer.android.com/guide/topics/permissions/overview.html.