Санкт-Петербургский государственный университет

Афанасов Артем Константинович

Выпускная квалификационная работа

Извлечение данных Android-приложения WhatsApp

Уровень образования: бакалавриат

Направление *02.03.03 "Математическое обеспечение и администрирование информационных систем"*

Основная образовательная программа *СВ.5006.2017 "Математическое обеспечение и администрирование информационных систем"*

Профиль *Системное программирование*

Научный руководитель:
доцент кафедры СП, к.т.н. Ю. В. Литвинов

Консультант:
инженер-программист ООО «Белкасофт» А. И. Цой

Рецензент:
генеральный директор ООО «КОДА-технологии» А. Р. Ханов

Санкт-Петербург
2021

Saint Petersburg State University

Afanasov Artem

Bachelor's Thesis

WhatsApp backup acquisition and decryption for Android platform

Education level: bachelor

Specialty *02.03.03 "Software and Administration of Information Systems"*

Programme *CB.5006.2017 "Software and Administration of Information Systems"*

Profile *System Engineering*

Scientific supervisor:
Docent, C.Sc. Yurii Litvinov

Consultant:
Software Engineer LLC «Belkasoft» Arthur Tsoy

Reviewer:
CEO LLC «CODA technology» Artur Khanov

Saint Petersburg
2021

Оглавление

Введение

Используя приложения для обмена сообщениями, человек может передавать любую информацию. Это могут быть как обыденные сообщения, так и данные, связанные с преступлениями. Разработчики мессенджеров обеспечивают безопасность пользовательских данных путем шифрования. Но шифрование данных затрудняет для экспертов в области цифровой криминалистики анализ данных подозреваемого, которые могут содержать потенциальные улики. То есть эксперты должны иметь возможность анализировать полученные данные. Им требуются инструменты для извлечения данных с целью получения потенциальных улик. Данные могут находиться на различных источниках: внешние накопители; внутренняя память устройств; облачные хранилища.

Компания «Белкасофт» занимается разработкой инструмента Belkasoft X, одной из задач которого является получение доступа к данным из облачных хранилищ. Компанией была поставлена задача исследовать возможности извлечения данных приложения WhatsApp для платформы Android. На момент января 2021 года WhatsApp является самым популярным в мире мессенджером [12].

WhatsApp реализует безопасность данных пользователей, шифруя резервные копии их сообщений. Резервная копия базы данных содержит: тексты сообщений, контакты получателя/отправителя сообщений, информацию о пересланных файлах, список чатов, время прочтения сообщения, упомянутые в сообщении аккаунты и другую информацию. Приложение WhatsApp создает зашифрованные резервные копии данных на устройстве пользователя и дублирует их в облако (Google Drive для платформы Android). Извлекать резервные копии возможно методами «физическим» и «облачным». «Физическим» методом возможно выгрузить зашифрованные и не зашифрованные базы данных, а также ключ расшифровки. Для «физического» метода необходим доступ к Android-устройству. А без доступа к Android-устройству работает «облачный» метод извлечения данных, для которого необходимы логин и пароль от Google Drive и SMS-код от сервера WhatsApp. В случае «об-

лачного» метода ключ расшифровки необходимо генерировать.

В метод получения данных без Android-устройства входят следующие шаги.

- Выгрузка зашифрованных баз данных из Google Drive. Требуется логин/пароль от учетной записи Google Drive. В рамках настоящей работы логин/пароль являются заранее известными.
- Генерация ключа расшифровки путем имитации запросов к серверу WhatsApp. Требуется код, отправляемый сервером WhatsApp в SMS-сообщении на привязанный к аккаунту WhatsApp номер телефона.
- Расшифровка базы данных.

Существуют проекты, реализующие «физическое» и «облачное» извлечение данных Android-приложения WhatsApp. Это как коммерческие продукты с закрытым исходным кодом, так и открытые решения по выгрузке баз данных из облака и их последующей расшифровке имеющимся ключом. Но для получения доступа к данным приложения WhatsApp без использования Android-устройства необходим важный шаг — генерация ключа расшифровки. Однако проектов с открытым доступом, осуществляющих генерацию ключа расшифровки для актуальных версий WhatsApp, не существует. WhatsApp-Key-Generator — старый проект по генерации ключа, который уже не актуален и не способен генерировать ключ для расшифровки настоящих баз данных. В данной работе исследован алгоритм генерации ключа расшифровки резервных копий данных приложения WhatsApp платформы Android.

По причине отсутствия открытых решений по генерации ключа расшифровки необходимо изучать работу самого приложения, проводя обратную разработку[1]. Для этого использовались атака посредника[2] на

[1]Обратная разработка (также называемая реверс-инжинирингом) – это исследование работы существующего приложения.

[2]Атака посредника – это вид доступа к трафику в криптографии, при которой атакующий ретранслирует и (или) изменяет связь между клиентом и сервером.

HTTPS-трафик и декомпиляция APK-файла[3], проводилась модификация исходного кода приложения WhatsApp и анализировались файлы WhatsApp в системе Android. Целью модификации был анализ XMPP-трафика[4]. Для полного доступа к системе Android использовались рут-права[5].

В данной работе исследованы «физический» и «облачный» методы доступа к данными пользователя WhatsApp. Особое внимание уделено второму методу, для исследования которого требуется проводить обратную разработку. Также реализовано «облачное» извлечение данных, которое интегрировано в Belkasoft X. Доклад по данной работе был представлен на LII международной научной конференции аспирантов и студентов «Процессы управления и устойчивость» (CPS'21), а статья готовится к публикации в журнале «Процессов управления и устойчивости».

[3]Декомпиляция APK — это получение исходного кода Android-приложения, архивированного в APK-файл.

[4]XMPP — протокол связи, основанный на XML и непосредственно на TCP-сокетах.

[5]От англ. root – традиционное имя суперпользователя в UNIX-подобных системах.

1. Постановка задачи

Целью данной работы является реализация способов получения доступа к резервным копиям Android-приложения WhatsApp. Для ее достижения необходимо решить следующие задачи.

- Провести обзор существующих решений.
- Получить доступ к резервным копиям базы данных приложения WhatsApp непосредственно с Android-устройства путем:
  - рутирования устройства,
  - понижения версии приложения WhatsApp.
- Изучить генерацию ключа расшифровки резервной копии базы данных приложения WhatsApp и имитировать WhatsApp-клиент для получения данных ключа.
- Реализовать и интегрировать в продукт Belkasoft X получение доступа к резервной копии базы данных приложения WhatsApp из облака Google Drive.
- Провести апробацию.

2. Обзор

2.1. Существующие решения по получению доступа к резервным копиям приложения WhatsApp

2.1.1. Проприетарные решения

Следующие продукты являются коммерческими проектами по получению доступа к резервным копиям приложения WhatsApp.

- Оксиджен Софтвер – «Мобильный Криминалист».
- Cellebrite – «UFED».
- Magnet – «AXIOM».
- Elcomsoft – «Elcomsoft Explorer for WhatsApp».

Согласно сравнительной таблице (см. Приложение А) были сделаны следующие выводы:

- выгрузка данных (зашифрованные базы данных, ключ расшифровки) непосредственно из Android-устройства не ново: оно есть во всех перечисленных продуктах [18] [4] [9] [5];
- выгрузка данных из Google Drive также есть у всех [3] [8] [5] [18];
- у двух из четырех продуктов («AXIOM», «Мобильный Криминалист») имеется возможность выгрузки данных посредством сканирования QR-кода [17] [10];
- генерация ключа расшифровки резервной копии баз данных приложения WhatsApp имеется у двух из четырех продуктов [16] [5] («Мобильный Криминалист», «Elcomsoft Explorer for WhatsApp»).

Исходя из имеющихся решений, необходимые данные для расшифровки резервных копий получаются посредством регистрации на сервере WhatsApp через SMS-код. Также исходя из статьи «Как дешифровать сообщения WhatsApp из резервной копии на Android и iOS» [2], можно сделать вывод о том, что ключ расшифровки отправляется клиенту с сервера WhatsApp. Следовательно, при реверс-инжиниринге нужно обратить внимание на клиент-серверное взаимодействие на предмет передачи данных ключа расшифровки. Также необходимо изучить алгоритм работы приложения WhatsApp посредством декомпилирования APK-файла.

2.1.2. Открытые решения

Для извлечения баз данных из Google Drive используется WhatsApp GD Extractor [13]. Требуется логин/пароль для входа в Google-аккаунт. Существующие закрытые программные продукты имеют те же требования для извлечения данных из Google Drive. Для извлечением баз данных и ключа расшифровки непосредственно из устройства используется проект WhatsApp Key/DB Extractor [14]. Расшифровывать базы данных позволяет инструмент WhatsApp Crypt12 Database Decrypter [6]. В настоящей версии WhatsApp используется алгоритм шифрования *AES-GCM*.

2.2. Расшифровка трафика в новых версиях WhatsApp

Авторы статьи «WhatsApp network forensics: Decrypting and understanding the WhatsApp call signaling messages» [7] дешифровали интернет-трафик WhatsApp. Но для повтора их эксперимента необходим файл */data/data/com.whatsapp/files/pw*. Этот файл отсутствует в файловой системе актуальной версии приложения. Попытка установить пониженную версию не приносит нужных результатов, поскольку отсутствует возможность пройти регистрацию, и по этой причине в файловой системе мало полезных файлов, относящихся к

WhatsApp. Поэтому сейчас данное исследование не может помочь в расшифровке трафика актуальной версии приложения.

2.3. Содержание резервной копии базы данных

В статье «Forensic analysis of WhatsApp Messenger on Android smartphones» [1] описано содержание расшифрованной базы данных *msgstore.db*: таблиц *messages* и *chat_list*, содержащих разнообразную информацию для анализа. Но в актуальной базе в интересующую таблицу *messages* на данный момент входит большее число атрибутов. Полезную информацию содержат следующие:

- *media_caption* (заголовок отправленного сообщения);
- *read_device_timestamp* (время прочтения сообщения на устройстве);
- *played_device_timestamp* (время воспроизведения сообщения);
- *mentioned_jids* (упомянутые в сообщении аккаунты).

Аналогично с таблицей *chat_list*:

- *subject* (название беседы);
- *creation* (время создания);
- *last_read_receipt_send_message_table_id* (порядковый номер записи в таблице *messages*, соответствующей последнему прочитанному сообщению пользователем, у которого извлечена данная база).

2.4. Используемые инструменты и технологии

2.4.1. Рутирование, jailbreak, Root Explorer

Рутирование устройства – это получения прав суперпользователя (от англ. root – традиционное имя суперпользователя в UNIX-подобных

ситемах). Одна из целей рутирования – это получения полного доступа к файловой системе Android.

jailbreak – это повышение привилегий на устройстве Apple, позволяющее получить root-доступ в операционной системе и возможность устанавливать программное обеспечение, недоступное в iOS App Store.

Для управления и просмотра файлов приложения WhatsApp, находящихся в области памяти, доступной только на рутированном устройстве, недостаточно предустановленного файлового менеджера. Задачу просмотра таких файлов удобно решать с помощью файлового менеджера Root Explorer. Приложение также позволяет получить информацию о времени создания файлов, что помогает при анализе генерации ключа шифрования.

2.4.2. Android Debug Bridge

В проекте WhatsApp Key/DB Extractor используется Android Debug Bridge – это инструмент командной строки, позволяющий общаться с устройством. Например, устанавливать приложения на телефон, обмениваться данными между устройством и компьютером, создавать резервные копии приложений. Данный инструмент позволяет автоматизировать методы «физического» извлечения данных.

2.4.3. Снифферы Burp Suite Community, Wireshark

Для анализа HTTPS-трафика используется Burp Suite Community. Инструмент позволяет проводить атаку посредника путем перехвата и расшифровки пакетов через прокси, их изменения и отбрасывания. Это помогает анализировать трафик, например, понимать, какие пакеты относятся к клиент-серверному общению, связанному с генерацией ключа. Также был использован снифер Wireshark вместе с *sslkeylog*-файлом для расшифровки HTTPS-траффика.

2.4.4. Протокол XMPP

XMPP — протокол связи, основанный на XML и непосредственно на TCP-сокетах. Это протокол для мгновенного обмена сообщениями. Используется приложением WhatsApp после регистрации аккаунта на сервере WhatsApp. Снифферы Burp Suite Community, Fiddler не спроектированы для перехвата XMPP-трафика. Перехват XMPP-пакетов возможен с помощью Wireshark и tcpdump [6].

2.4.5. Библиотека Cydia Substrate и система сборки theos

Для модификации iOS-версии WhatsApp использовались система сборки theos и библиотека Cydia Substrate. Cydia Substrate в рантайме определяет необходимые методы и подменяет их вызовы на модифицированные. Система сборки theos подключает написанную динамическую библиотеку к приложению WhatsApp.

2.4.6. Декомпилятор jadx

Для декомпиляции APK-файла приложения WhatsApp использовался декомпилятор jadx. По сравнению со связкой программ для декомпиляции APK-файлов jd-gui & dex2jar (графическая утилита, отображающая исходный код Java и принимающая на вход файл с jar-расширением, и утилита, преобразующая APK-файл в jar-файл, соответственно) jadx декомпилирует больше кода. Более того, он позволяет показывать тот код, который декомпилятор посчитал некорректным, непоследовательным (а именно в этом коде оказалась существенная логика программы).

[6]Консольный инструмент для перехвата трафика

3. Получение доступа к резервным копиям базы данных приложения WhatsApp непосредственно с устройства

В данной работе рассмотрены два способа «физического» извлечения данных: рутирование и понижение версии. Рутирование устройства позволяет извлекать данные как из разблокированного, так и заблокированного устройства. Понижение версии работает с разблокированным устройством, которое может быть как рутировано, так и не рутировано. В настройках Android-устройства необходимо разрешить отладку по USB.

3.1. Рутирование устройства и файлы приложения WhatsApp

Данные приложения WhatsApp хранятся как в приватной, так и общедоступной области памяти. В файловой системе находятся базы данных (зашифрованные и не зашифрованные), ключ расшифровки, идентификационные данные пользователя. Получать доступ к данным приложения WhatsApp возможно непосредственно с Android-устройства.

Недоступная область памяти – */data/data/com.whatsapp/*, доступная – *.../notPrivateMemory/sdcard/WhatsApp/Databases/*. Используя Root Explorer, на рутированном устройстве были найдены интересующие файлы приложения WhatsApp:

- /data/data/com.whatsapp/
  - databases/
    - msgstore.db (не зашифрованная база данных, содержит информацию о переданных сообщениях)
  - files/
    - key (ключ расшифровки)

  - shared_prefs/
    * whatsapp_preferences_light.xml (содержит узел *edge_routing_info*, используемый для авторизации на сервере WhatsApp)

- .../notPrivateMemory/sdcard/WhatsApp/Databases/
  - msgstore*.db.crypt12 (зашифрованные базы данных)

Когда телефон рутирован, извлечь любые файлы файловой системы возможно с использованием Android Debug Bridge вне зависимости от того, заблокировано или разблокировано устройство. В скрипте по извлечению, например, ключа расшифровки из рутированного устройства нужно использовать следующие команды.

```
adb shell \
''su -c 'cp /data/data/com.whatsapp/files/key /mnt/sdcard'''

adb pull /mnt/sdcard/key C:\Artyom
```

Для исполнения первой команды необходимы права суперпользователя рутованного устройства, что не всегда может быть обеспечено. Поэтому возможность получения данных без прав суперпользователя также представляет интерес.

3.2. Понижение версии приложения WhatsApp

В старых версиях возможно существование уязвимостей, которые позволяют извлекать данные приложений актуальных версий. Проект с открытым исходным кодом «WhatsApp Key/DB Extractor» [14] предоставляет решение для извлечения следующих файлов: *key*, расшифрованные базы данных *msgstore.db*, *wa.db*, *axolotl.db*. Это shell-скрипт, который использует Android Debug Bridge. Для разблокированного телефона без прав суперпользователя возможно использовать метод понижения версии приложения. Скрипт содержит следующие шаги, работающие как для рутованного, так и не рутованного Android-устройства.

- Удаляется исходная версия WhatsApp, но сохраняются данные приложения (*adb shell pm uninstall -k com.whatsapp*).
  - Если версия Android 7.0 или выше, то необходимо перезагрузить Android-устройство (иначе встречается ошибка «INSTALL_FAILED_VERSION_DOWNGRADE»).
- Устанавливается пониженная версия WhatsApp 2.11.431 (*adb shell pm install -r -d whatsapp.apk*).
- Выполняется резервное копирование приложения (*adb backup com.whatsapp*), для которого необходимо разблокировать устройство и подтвердить действие. Результатом копирования является файл *backup.ab*, который затем распаковывается в файл с расширением *.tar*.
- Из архива *.tar* распаковываются файлы: *msgstore.db, wa.db* (данные о контактах пользователя), *axolotl.db* (идентификационные данные пользователя), *whatsapp.cryptkey* (key-файл), *chatsettings.db*.

Аналогичная выгрузка резервной копии приложения WhatsApp актуальной версии командой *adb backup com.whatsapp* не дает нужных результатов. Для рутированного устройства из актуальной версии извлекается лишь один файл *_manifest* с информацией об Android-приложении. Файлы манифеста новой и пониженной версии отличаются лишь номером версии. Для нерутированного устройства результатом выгрузки резервной копии актуальной версии приложения WhatsApp является архив, который не содержит файлов.

4. Генерация ключа расшифровки резервной копии базы данных приложения WhatsApp и имитация WhatsApp-клиента для получения данных ключа

Приложение WhatsApp предоставляет возможность сохранения резервных копий данных пользователя в облаке Google Drive. Эти данные возможно извлечь из облака даже после удаления баз данных с устройства либо удаления приложения.

Используя проект «WhatsApp Google Drive Extractor» [13] были извлечены файлы из Google Drive: зашифрованный файл с сообщениями *msgstore.db.crypt12* и медиа-файлы (файлы папок *Wallpaper*, *WhatsApp Animated Gifs, WhatsApp Audio, WhatsApp Documents, WhatsApp Images, WhatsApp Stickers, WhatsApp Voice Notes*, выгружаемые приложением WhatsApp в облако из директории *.../notPrivateMemory/sdcard/WhatsApp/media/*). Необходимы логин/пароль от Google-аккаунта и номер телефона пользователя (используется для определения аккаунта в URI-строке HTTPS-запроса).

Файл *msgstore.db.crypt12* расшифровывается при помощи *key*-файла. Расшифровка базы данных, извлеченной из Google Drive проводилась с помощью инструмента с открытым исходным кодом «WhatsApp Crypt12 Database Decrypter» [6]. Для чтения файла базы данных использовался DB Browser for SQLite. Для создания «облачного» решения извлечения данных необходимо генерировать key-файл.

4.1. Исследование генерации ключа расшифровки путем реверс-инжиниринга WhatsApp

Возможны следующие способы появления ключа на устройстве: получение ключа в чистом или зашифрованном виде от сервера, генерация ключа на устройстве. Исследование генерации ключа в данной работе показало, что WhatsApp генерирует ключ на устройстве. В начале ра-

боты приложения WhatsApp скачивается зашифрованная база данных. Далее ключ расшифровки генерируется на устройстве: часть ключа генерируется самостоятельно, часть ключа берется из зашифрованной базы, часть получается от сервера. В процессе исследования был получен алгоритм расшифровки резервной копии (см. рис 1).

Рис. 1: Алгоритм расшифровки резервной копии

4.1.1. Перехват HTTPS-трафика

Необходимость общения с сервером WhatsApp показало изучение файловой системы Android-устройства с правами суперпользователя: ключ создается после появления баз данных из Google Drive в файловой системе. Также были выполнены следующие действия на Android-устройстве.

- Открытие приложения WhatsApp, отключение от интернета, удаление key-файла из файловой системы, открытие чатов WhatsApp. Результат – файл */data/data/com.whatsapp/files/key* на устройстве не создан.

- Открытие приложения WhatsApp, отключение от интернета, удаление key-файла из файловой системы, открытие чатов WhatsApp, подключение к интернету. Файл */data/data/com.whatsapp/files/key* создан.

Сначала был проанализирован HTTPS-трафик.

Анализ трафика снифферoм Packet Capture

Сначала для анализа трафика было использовано Android-приложение Packet Capture (его аналог, HttpCanary, не позволяет выгружать pcap файлы для чтения в Wireshark).

Судя по времени в логе пакетов приложения Packet Capture, а также данным из Root Explorer о создании key-файла, ключ дешифровки создаётся в файловой системе не более чем через две секунды после последнего пакета в логе.

Генерация ключа происходит по нескольким сценариям.

- Приложение WhatsApp включено (либо в фоне, либо основным приложением), ключ дешифрования из файловой системы удален. Key-файл создаётся в файловой системе после переподключения к интернету.

- Интернет включен; key-файл удалён; WhatsApp либо включен, либо нет. Ключ создаётся после перезагрузки WhatsApp.

- Интернет включен, WhatsApp включен, ключ удалён. Включение сниффинга через Packet Capture влечет создание ключа в файловой системе.

С помощью Packet Capture можно увидеть, что после передачи пакетов между клиентом и сервером в файловой системе появляется key-файл: сначала отправляется запрос *GET /generate_204 HTTP/1.1.*, затем приходит код ответа *The HTTP 204 No Content*; после этого происходит общение между клиентом и сервером, трафик которого зашифрован. Выгруженные с помощью этой программы pcap файлы

можно прочитать в Wireshark, но все пакеты оказываются без тела. Поэтому информации недостаточно и необходимо использовать снифферы на персональном компьютере.

Анализ трафика через эмуляцию Android

Изначально использование на персональном компьютере анализаторов трафика проходило через эмуляцию Android-устройства. Эмуляция проводилась при помощи инструмента Genymotion (версия Android – 9.0), который легко позволяет получить root-доступ. В качестве сниффера использовался Wireshark. Начальный анализ трафика показал, что используются протоколы TLSv1.2, TLSv1.3. Следовательно, необходимо производить подмену SSL-сертификатов на телефоне.

Анализ трафика сниффером Burp Suite Community

Для атаки посредника телефон необходимо настроить на подключение к интернету через прокси Burp Suite Community, а также установить сертификат от Burp Suite Community. Был перехвачен ряд HTTPS-запросов.

- Три GET-запроса на регистрацию клиента на сервере WhatsApp *https://v.whatsapp.net*:
  - */v2/exist?ENC=EncryptedQueryStringParameters1*;
  - */v2/code?ENC=EncryptedQueryStringParameters2*;
  - */v2/register?ENC=EncryptedQueryStringParameters3.*
    - Сервер отправляет параметр «*edge_routing_info*»
- Запросы на выгрузку базы данных из Google Drive с сервера *https://backup.googleapis.com*:
  - */v1/clients/wa/backups/digitsOfPhone?mode=restore*;
  - */v1/clients/wa/backups/digitsOfPhone/files ?mode=restore&pageSize=5000.*

С каждым запросом на регистрацию передаются идентификационные параметры пользователя. В трафике они шифрованы и переданы как один параметр *ENC*. Первый запрос проверяет, существует ли аккаунт для данного телефона. Далее клиент запрашивает SMS-код от сервера для регистрации. Третьим запросом клиент регистрируется, используя SMS-код. В ответ на третий запрос сервер отправляет параметр *edge_routing_info*, требуемый для идентификации пользователя при авторизации.

Анализ трафика снифферoм Wireshark

В конфигурацию Wireshark был добавлен файл *sslkeylog*, генерируемый браузером Firefox, и была расшифрована часть трафика WhatsApp. Это позволило увидеть, что при клиент-серверном взаимодействии используется протокол Диффи-Хеллмана на эллиптических кривых[7], который входит в основу Noise Protocol Framework [11]. Noise используется при авторизации на сервере WhatsApp и шифровании передаваемых данных.

4.1.2. Структура ключа расшифровки

С использованием декомпилятора jadx приложение WhatsApp было исследовано для определения структуры *key*-файла. В ходе исследования оказалось, что для расшифровки базы данных достаточно только последнего блока из 32 байт *key*-файла. Ключ расшифровки имеет следующую структуру (см. рис. 2) и соответствующие декомпилированные названия.

- Первый раздел.
  - Байты 1-27 – константная метаинформация о файле ключа расшифровки: *key*-файл содержит массив байт длины 0x83.
  - Байты 28-29 (*cipherVersion*) – версия шифрования. На момент весны 2021 версия 0x0001.

[7]Протокол Диффи-Хеллмана – это метод безопасного обмена общими криптографическими ключами по общедоступному каналу

  - Байт 30 (*keyVersion*) – версия ключа. На момент весны 2021 версия 0x02.

- Второй раздел.
  - Байты 31-62 (*serverSalt*) – байты из зашифрованной базы данных.
  - Байты 63-78 (*googleIdSalt*) – байты из зашифрованной базы данных.
  - Байты 79-110 (*hashedGoogleId*) – результат SHA-256 от байтов 63-78.
  - Байты 111-126 – последовательность нулевых байт.

- Третий раздел.
  - Байты 127-157 (*cipherKey*) – байты, отправляемые от сервера в ходе XMPP-общения. Являются ключом, расшифровывающим *msgstore.db.crypt12*.

00000000	ACED	0005	7572	0002	5B42	ACF3	17F8	0608
00000010	54E0	0200	0078	7000	0000	83 00	01 02	C033
00000020	86DB	3C41	EEC5	78CA	2FCF	0B41	FE86	BC81
00000030	69C0	B292	D43F	C0DE	FFF2	5F13	9DF6	2892
00000040	EFA6	3AA4	CB0D	4C2D	77AA	C50D	BDA2	C632
00000050	B2FD	F5E6	0755	1D31	16B4	99C4	F533	C8B2
00000060	24E8	56CD	552F	4AC7	A096	D725	B9ED	0000
00000070	0000	0000	0000	0000	0000	0000	0000	7535
00000080	D059	3574	BFEF	1A6E	7321	6EAD	3B02	339E
00000090	119B	A730	DB5B	2A5D	1833	C26A	4532	

Рис. 2: Структура *key*-файла

При декомпиляции приложения был обнаружен код с названиями версий *.crypt12, .crypt13, .crypt14*. На данный момент WhatsApp для

шифрования использует версию под названием *.crypt12*. В данной версии используется алгоритм шифрования *AES-GCM*.

Изучения декомпилированного кода APK привело к следующему шагу – исследованию XMPP-общения с сервером WhatsApp: приложение формирует и обрабатывает XMPP-пакеты.

4.1.3. Логирование iOS-версии WhatsApp и перехват XMPP-трафика

Burp Suite Community позволяет перехватывать пакеты благодаря режиму *Intercept*, в котором возможно пересылать, отбрасывать и изменять пакеты устройства, подключенному к прокси. Рассмотрим следующие исходные данные во время штатной работы приложения WhatsApp: устройство подключено к интернету через прокси; *key*-файл отсутствует в файловой системе (то есть WhatsApp необходимо делать запросы серверу для генерации ключа расшифровки); включен режим *Intercept*, не пропускающий пакеты через прокси. Но *key*-файл всё равно генерируется в файловой системе, хотя в логах анализатора трафика присутствуют только пакеты с запросами, которые не были перенаправлены.

WhatsApp использует протокол XMPP, который не удалось перехватить ни Burp Suite Community, ни Fiddler. XMPP-пакеты были перехвачены с помощью Wireshark и tcpdump, но их не удалось расшифровать. Из-за сложностей в перехвате XMPP-трафика была проведена модификация приложения WhatsApp с целью получения информации о XMPP-общении.

Модификация Android-версии

Модифицировать Android-версию WhatsApp невозможно по причине проверки подписи APK-файла на стороне сервера WhatsApp. Если самостоятельно подписать APK-файл, собранный с помощью APKtool или APK studio, то при регистрации по номеру телефона возникает ошибка: «похоже, с вашей версией WhatsApp возникли проблемы, скачайте последнюю версию приложения с нашего сайта». Если сохранить старую подпись приложения и пересобрать APK,

то приложение невозможно установить на Android-устройство. Модифицировать старые версии WhatsApp невозможно по причине проверки версии приложения сервером WhatsApp и возникновения ошибки: «устаревшая версия приложения, установите новую». Способ удаления актуальной версии WhatsApp с сохранением данных, а затем установкой самостоятельно подписанной APK также невозможен, так как возникает ошибка: «новая подпись не совпадает с подписью предыдущей установленной версии».

Модификация iOS-версии

Использовалось iOS-устройство, на котором проведен jailbreak. Код библиотеки CoreFramework, используемой iOS-приложением WhatsApp, содержит методы по работе с XMPP. Используя библиотеку Cydia Substrate (пакет Cydia Substrate необходимо поставить на устройство), были подменены вызовы XMPP-методов. Таким образом, функция NSLog записывала в общий лог информацию, используемую при генерации XMPP-сообщений: сериализуемые перед отправкой данные и десериализуемые ответы сервера.

Для подмены оригинальных методов WhatsApp была написана библиотека, содержащая код модифицированных методов для подстановки вместо оригинальных и использующая библиотеку Cydia Substrate. Для подключения написанной библиотеки к WhatsApp, необходимо сконфигурировать специальный plist-файл[8].

Модификация оригинального метода на примере *serialize* класса *XMPPBinaryCoder* библиотеки *CoreFramework*:

```
%hook XMPPBinaryCoder

-(void *)serialize:(void *)arg2 options:(unsigned long long)arg3 {
        NSLog(@"HACKXMMPP serialize: %@",arg2);
        return %orig;
}
```

Для системы сборки theos, которая устанавливает пакет с обновлен-

[8]plist-файл – это список свойств в xml-формате, пары «ключ-значение» которого определяют информацию о конфигурации времени выполнения для приложения.

ными методами на устройство, необходимо написать makefile. Makefile должен содержать названия plist-файла и файла с модифицированными методами, а также команду для перезагрузки SpringBoard[9]. Перезагрузка необходима для активации Cydia Substrate.

Благодаря модификации методов, связанных с обработкой XMPP, удалось перехватить XMPP-трафик WhatsApp iOS-версии. Это позволило увидеть, какие параметры XMPP-запроса на получение данных ключа расшифровки используются, и какой XMPP-ответ отправляет сервер WhatsApp. Следующим этапом было изучение декомпилированного APK приложения WhatsApp на предмет генерации параметров и формирования XMPP-запроса. В ходе исследования оказалось, что в Android-версии WhatsApp XMPP-общение происходит так же, как и в iOS-версии.

4.1.4. Параметры XMPP-запроса Android-версии WhatsApp

00000000	ACED	0005	7572	0002	5B42	ACF3	17F8	0608
00000010	54E0	0200	0078	7000	0000	83 00	01 02	C033
00000020	86DB	3C41	EEC5	78CA	2FCF	0B41	FE86	BC81
00000030	69C0	B292	D43F	C0DE	FFF2	5F13	9DF6	2892
00000040	EFA6	3AA4	CB0D	4C2D	77AA	C50D	BDA2	C632
00000050	B2FD	F5E6	0755	1D31	16B4	99C4	F533	C8B2

Рис. 3: Байты ключа, извлекаемые из *msgstore.db.crypt12*

После анализа XMPP-общения iOS-версии WhatsApp, декомпилированный код Android-версии был исследован на предмет генерации параметров и формирования XMPP-запроса. Стало понятно, что для генерации параметров используются байты зашифрованной базы данных *msgstore.db.ctypt12*:

[9]SpringBoard – системное приложение главного экрана iOS.

00000000	0001	02 C0	3386	DB3C	41EE	C578	CA2F	CF0B
00000010	41FE	86BC	8169	C0B2	92D4	3FC0	DEFF	F25F
00000020	139D	F6 28	92EF	A63A	A4CB	0D4C	2D77	AAC5
00000030	0DBD	A2 C5	5940	7062	17FC	DF61	A746	D87E
00000040	EFA6	C371	4037	B483	7EF4	8D55	3076	13B6
00000050	FBAE	...						

Рис. 4: Байты *msgstore.db.crypt12*

4.2. Имитация WhatsApp-клиента для генерации ключа расшифровки

Последний фрагмент *key*-файла отправляется от сервера в ходе XMPP-общения. Поэтому для того, чтобы сгенерировать ключ расшифровки, требуется имитировать XMPP-запрос на получение данных ключа. Для имитации XMPP-запроса сначала необходимо задать и получить идентификационные данные пользователя, чтобы авторизоваться на сервере. Они генерируются при регистрации пользователя на сервере WhatsApp. Поэтому также необходимо имитировать HTTPS-запросы на регистрацию. Способы генерации параметров для регистрации клиента были найдены в проекте yowsup.

4.2.1. Регистрация клиента

Сначала на сервер вместе с запросом на регистрацию отправляется информация о номере телефона, идентификационные параметры пользователя на основе эллиптической кривой Curve25519, служебные параметры (см. проект yowsup[15]). В случае корректно предоставленных данных сервер отправляет SMS-код для регистрации на телефонный номер аккаунта. Далее клиент для завершения регистрации отправляет полученный код вместе с идентификационной информацией о себе. По завершению регистрации сервер высылает подтверждение и дополнительный идентификационный параметр – *edge_routing_info*.

Запрос с передачей SMS-кода и ответа при окончании регистрации

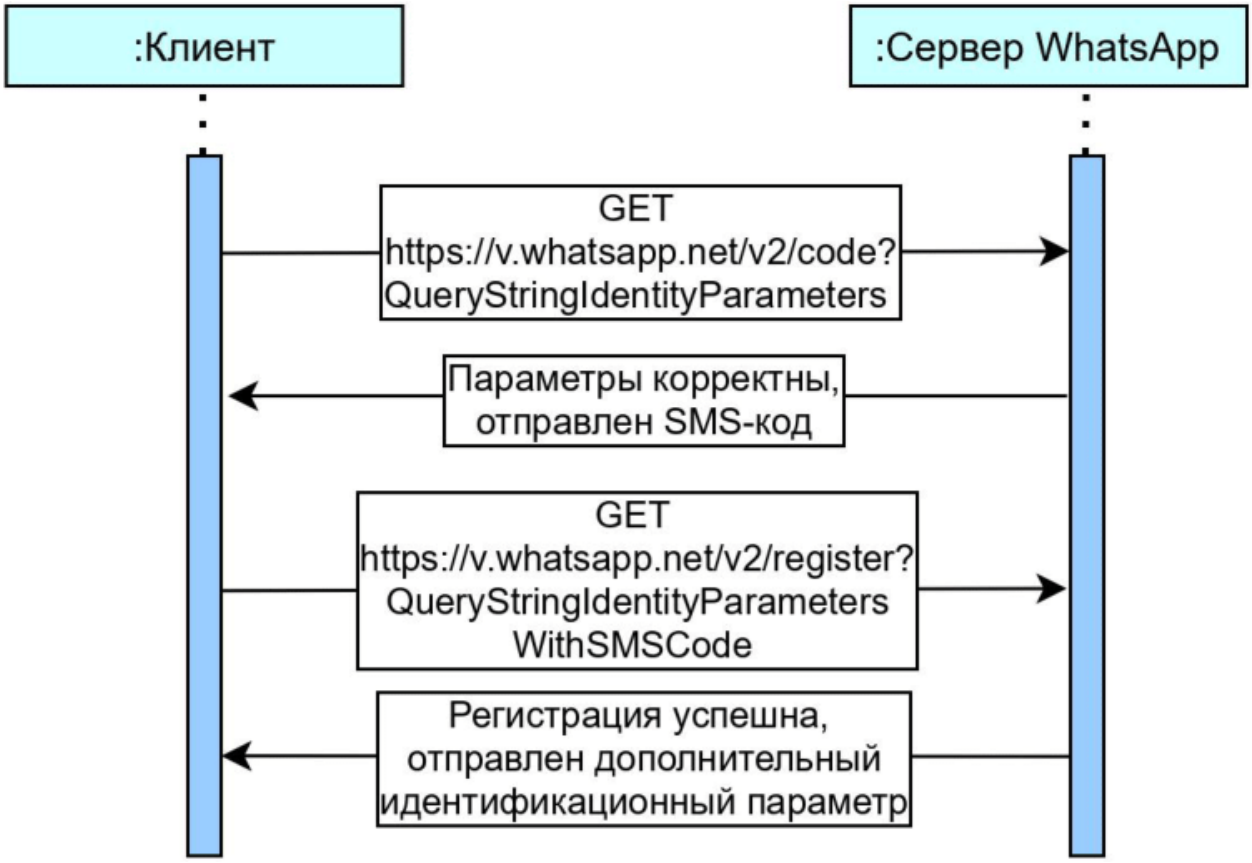

Рис. 5: Сценарий регистрации на сервере WhatsApp по сгенерированными параметрам

выглядит следующим образом.

- URI-строка для запроса клиента *https://v.whatsapp.net/v2/register* с параметрами *cc* (код страны), *in* (номер телефона), *authkey* (публичный ключ в кодировке base64, входящий в пару ключей, которая сгенерированна на основе эллиптической кривой Curve25519; приватный ключ данной пары сохраняется на устройстве и используется в Noise Protocol Framework при авторизации), *code* (6-значный SMS-код).

- Ответ сервера:

```
{
    ''status'' : ''ok'',
    ''login'' : ''phoneNumber'',
    ''type'' : ''existing'',
```

```
        ''edge_routing_info'' : ''base64data'',
        ''security_code_set'' : false
    }
```

4.2.2. Авторизация клиента и XMPP-запрос на получение данных ключа расшифровки

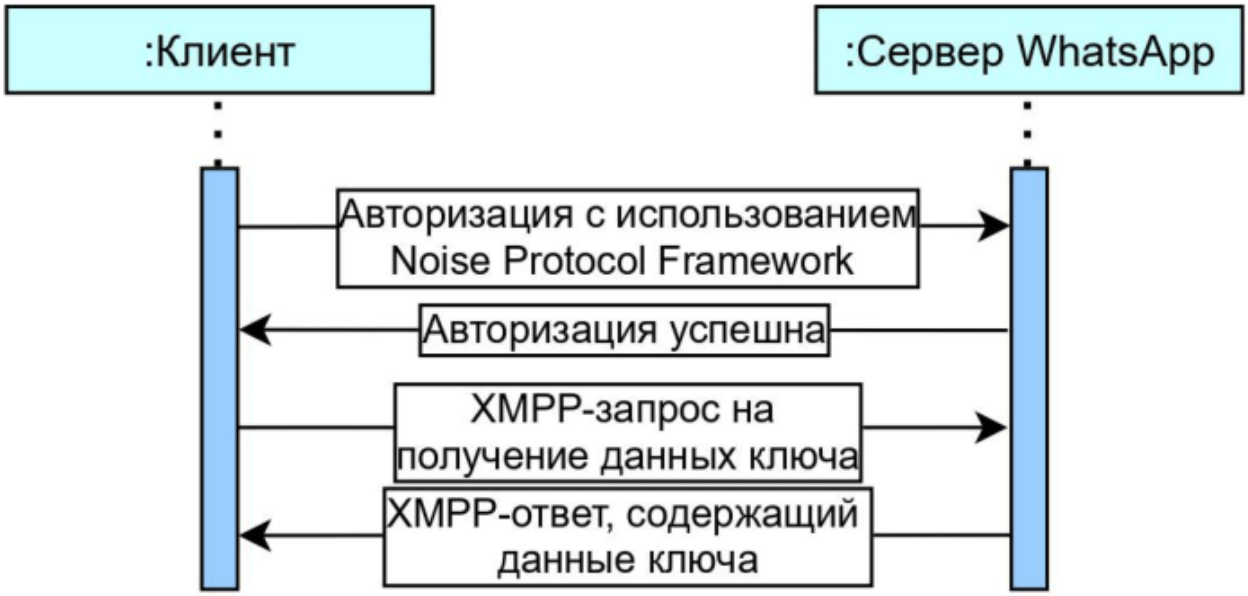

Рис. 6: Сценарий получения данных ключа шифрования по XMPP

После регистрации и формирования идентификационных данных клиента следует авторизация на сервере с использованием протокола Noise, основанном на соглашении о ключах Диффи-Хеллмана. По Noise Protocol Framework шифруются и расшифровываются передаваемые данные. При шифровании Noise использует данные приватного ключа из пары, которая генерируется на стороне клиента на основе эллиптической кривой Curve25519 при регистрации. В итоге, сервер WhatsApp отправляет последний фрагмент *key*-файла длиной 32 байта.

4.2.3. Алгоритм генерации ключа расшифровки без участия устройства

1. Выгрузить зашифрованную базу данных из Google Drive с целью использования ее байтов при генерации параметров XMPP-

запроса.

2. Сформировать пару из публичного и приватного ключей, используемых библиотекой Noise Protocol Framework в ходе XMPP-общения, и идентификационные данные аккаунта.

3. Зарегистрироваться на сервере, используя SMS-код, идентификационные данные аккаунта и публичный ключ из пары.

4. Авторизоваться на сервере, используя Noise Protocol Framework, приватный ключ из пары и идентификационные данные аккаунта, тем самым установив безопасное соединение согласно протоколу Диффи-Хеллмана.

5. Используя Noise Protocol Framework и данные зашифрованной базы данных, отправить XMPP-запрос на получение последнего фрагмента ключа расшифровки по установленному безопасному соединению.

5. Архитектура прототипа по генерации ключа расшифровки без участия устройства

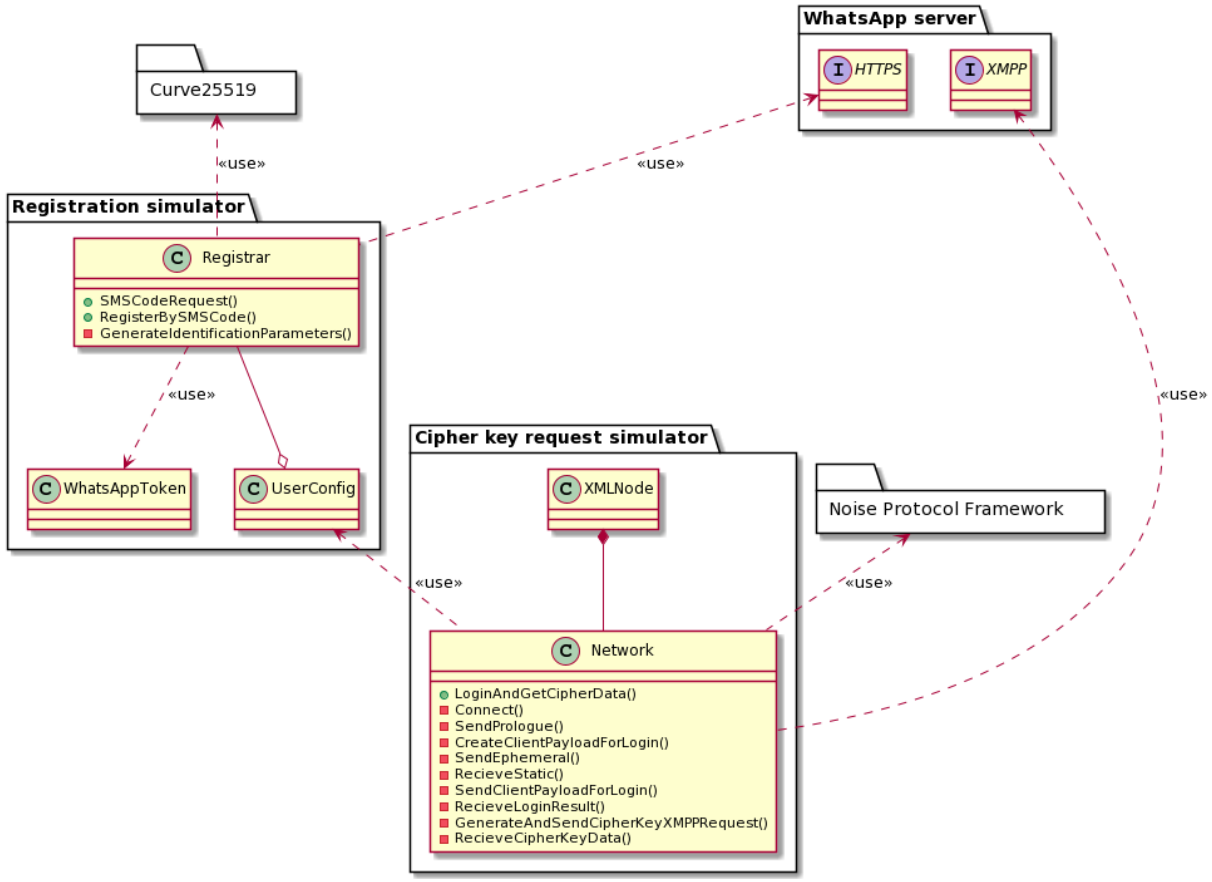

Рис. 7: Архитектура прототипа по генерации ключа расшифровки без участия Android-устройства. Реализовано на C#

Был реализован прототип по генерации ключа для расшифровки *msgstore.db.crypt12* на языке C# (языки реализации Belkasoft X: C# и C++). В его архитектуру входят пять компонентов: *Curve25519* (библиотека эллиптической кривой), *Noise Protocol Framework* (библиотека для шифрования сообщений, основанная на соглашении о ключах Диффи-Хеллмана), *Registration simulator* (имитатор регистрации), *Cipher key request simulator* (имитатор запроса на получение ключа расшифровки), *WhatsApp server* (сервер WhatsApp, с которым происходит общение при имитации WhatsApp-аккаунта).

Реализованное решение по генерации ключа расшифровки было интегрировано в продукт Belkasoft X. Это позволило организовать конвейер для получения доступа к резервным копиям баз данных приложения WhatsApp без участия Android-устройства: «выгрузка зашифрованной базы данных из Google Drive |> генерация ключа расшифровки |> расшифровка базы данных». Выгрузка и расшифровка уже были реализованы в продукте Belkasoft X сотрудниками компании.

5.1. Компонент WhatsApp server

К серверу WhatsApp происходит подключение как по протоколу HTTPS, так и по протоколу XMPP. В случае XMPP-протокола подключение к серверу WhatsApp (*e1whatsapp.net* или *e2whatsapp.net*, или ..., или *e16whatsapp.net*) происходит по TCP-сокету. Для HTTPS-подключения формируется URI-строка, содержащая endpoint (точка входа в сервис регистрации) WhatsApp-сервера (*v.whatsapp.net*) и идентификационные параметры аккаунта.

5.2. Компонент Registration simulator

Данный компонент имитирует регистрацию аккаунта на сервере WhatsApp. Метод *SMSCodeRequest()* класса *Registrar* формирует URI-строку с идентификационными параметрами нового аккаунта и отправляет на сервер WhatsApp GET-запрос на получение SMS-кода на телефон. Идентификационные параметры генерируются методом *GenerateIdentificationParameters*. Также для генерации идентификационных параметров используется класс *WhatsAppToken*, в котором генерируется токен, зависящий от версии приложения WhatsApp. Метод *RegisterBySMSCode* формирует URI-строку с идентификационными параметрами нового аккаунта, SMS-кодом (данный шестизначный код вводится в программу пользователем) и отправляет на сервер WhatsApp GET-запрос на регистрацию аккаунта. Идентификационные параметры формируются в объект класса *UserConfig*.

5.3. Компонент Curve25519

Данная библиотека используется компонентном *Registration simulator* для генерации идентификационных параметров. При регистрации аккаунта на основе эллиптической кривой генерируются ключи, используемые в схеме согласования ключей Диффи-Хеллмана.

5.4. Компонент Cipher key request simulator

Данный компонент имитирует XMPP-запрос на получение данных ключа расшифровки. Для шифрования и расшифровки пакетов используется компонент *Noise Protocol Framework*. Методом *Connect()* происходит подключение к TCP-сокету сервера WhatsApp. Метод *SendPrologue()* отправляет байты строк-признаков начала авторизации «ED\x00\x01» и «WA\x04\x00», а также байты строки *edge_routing_info*, получаемые от сервера WhatsApp при регистрации. Метод *CreateClientPayloadForLogin()*, используя protobuf и идентификационные данные аккаунта из объекта *UserConfig*, генерирует байты для авторизации аккаунта. Метод *SendEphemeral()* отправляет временный открытый ключ серверу. Метод *ReceiveStatic()* получает от сервера его не зашифрованный временный и зашифрованный публичный ключи, а также зашифрованную полезную нагрузку, в которой содержится информация о сертификате. Метод *SendClientPayloadForLogin()* отправляет зашифрованный публичный ключ и зашифрованную полезную нагрузку, в которой содержатся данные аккаунта. Метод *ReceiveLoginResult()* получает от сервера информацию о состоянии авторизации. Метод *GenerateAndSendCipherKeyXMPPRequest()*, используя тип *XMLNode*, формирует и отправляет XML-представление XMPP-запроса на получение данных ключа расшифровки. Метод *ReceiveCipherKeyData()* получает от сервера последний фрагмент *key*-файла, длина которого 32 байта.

5.5. Компонент Noise Protocol Framework

Данная библиотека, основанная на соглашении о ключах Диффи-Хеллмана, используется компонентом *Cipher key request simulator* для шифрования и расшифровки пакетов, передаваемых по TCP-сокету. Для установки защищенного канала передачи данных в компоненте *Cipher key request simulator* используется паттерн *XX*, включающий в себя следующее. Первое сообщение отправляется участником А и состоит из публичного не зашифрованного ключа «e». Второе сообщение отправляется участником Б и состоит из публичного не зашифрованного ключа «e», за которым следует зашифрованный публичный ключ «s» и зашифрованная полезная нагрузка. Третье сообщение отправляется участником А и состоит из зашифрованного открытого ключа «s», за которым следует зашифрованная полезная нагрузка.

5.6. Последовательность «облачного» извлечения в Belkasoft X

Пользователь взаимодействует (см. рис. 8) с объектом *Analyzer*, который, в свою очередь, работает с объектами *GoogleDriveDownloader*, *Decrypter*, *CipherKeyGenerator*. Сначала происходит выгрузка зашифрованных баз данных из *GoogleDrive*. Затем имитируются регистрация аккаунта, авторизация и запрос на получение ключа расшифровки на *WhatsAppServer*. Последним этапом *Decrypter* расшифровывает базу данных, а *Analyzer* предоставляет ее пользователю.

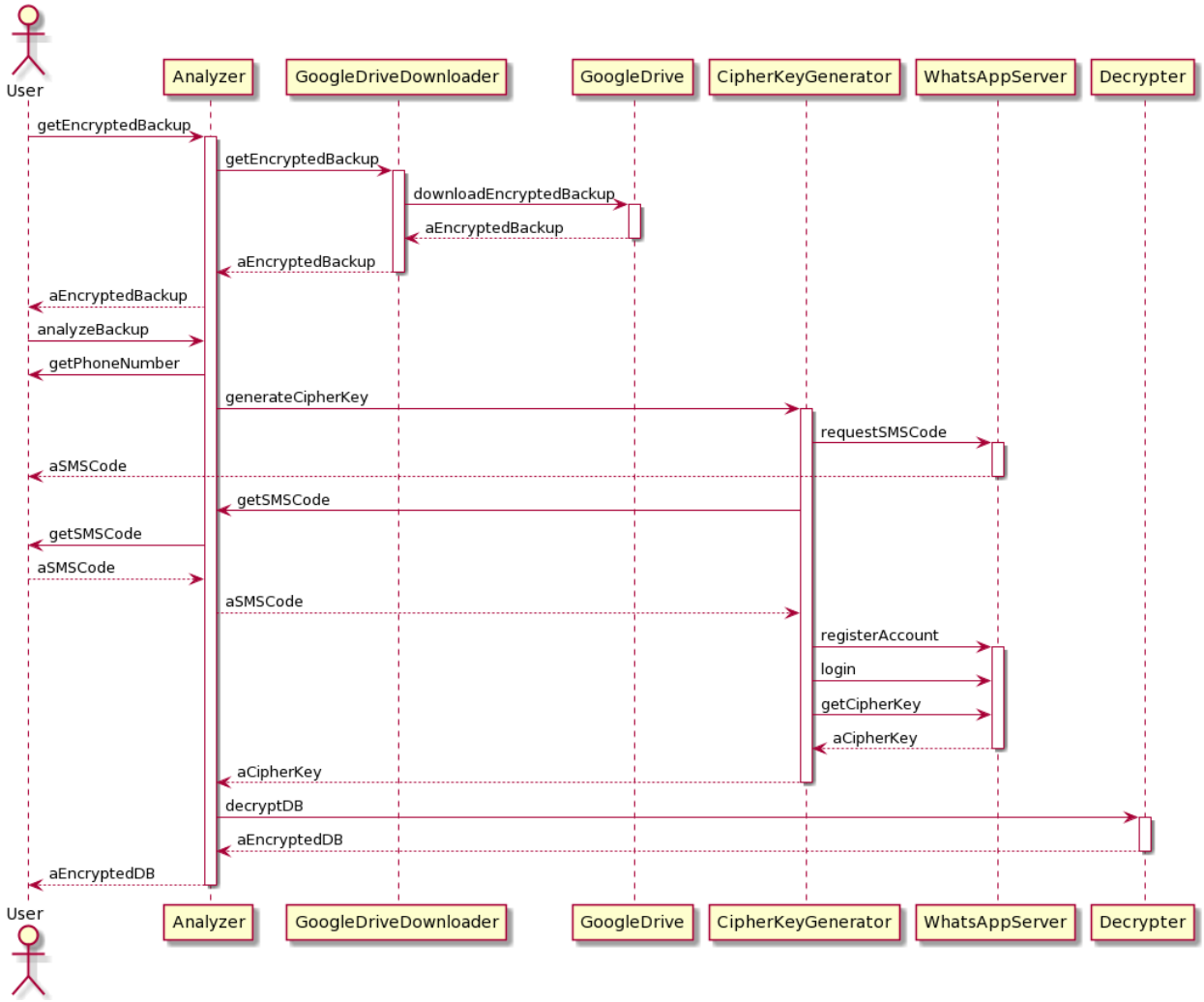

Рис. 8: Описание работы «облачного» получения доступа к данным в Belkasoft X

6. Апробация

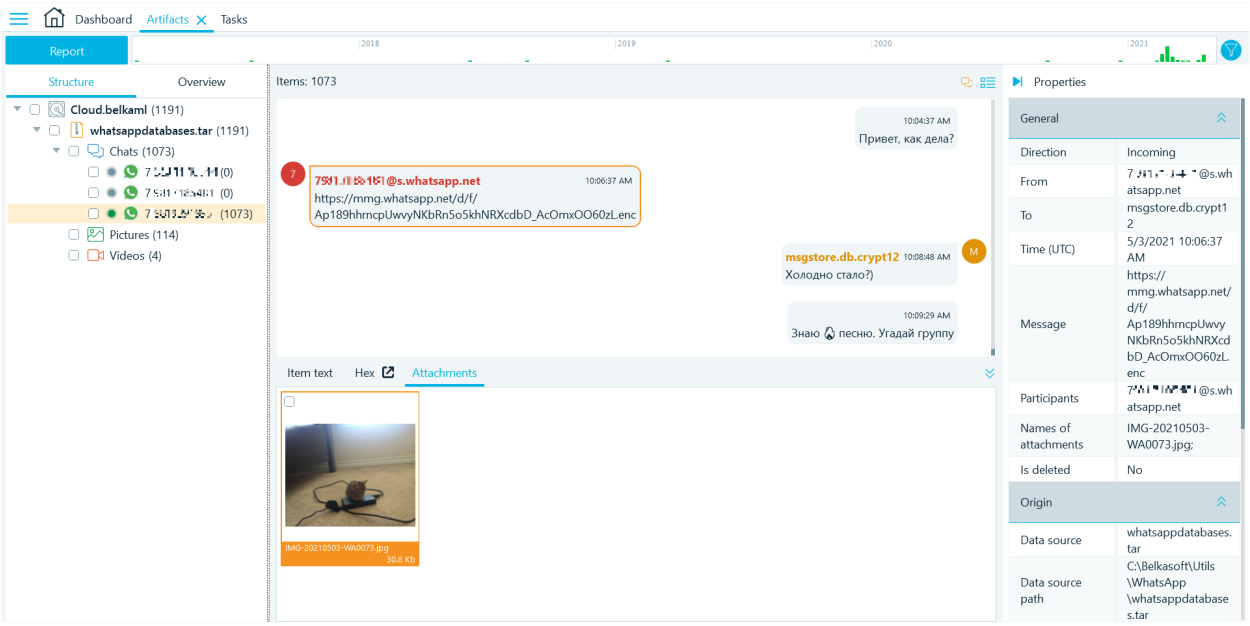

Рис. 9: Пример базы данных, расшифрованной сгенерированным ключом

Для тестового аккаунта Google Drive и тестового номера телефона, используя проект Belkasoft X и интегрированное в него решение по генерации ключа, была выполнена процедура получения доступа к резервной копии базы данных приложения WhatsApp без участия Android-устройства.

- Из Google Drive выгружена зашифрованная база данных *msgstore.db.crypt12*.
- Путем имитации WhatsApp-клиента и подтверджения SMS-кода сгенерирован ключ расшифровки.
- База данных *msgstore.db.crypt12* расшифрована с использованием сгенерированного ключа.

Пример расшифрованной базы данных можно увидеть на рис. 9.

Заключение

В итоге, были достигнуты следующие результаты.

- Были рассмотрены следующие закрытые решения для получения доступа к резервным копиям приложения WhatsApp: «Мобильный Криминалист», «UFED», «AXIOM», «Elcomsoft». Были также рассмотрены открытые решения: WhatsApp-GD-Extractor-Multithread, WhatsApp-Key-DB-Extractor, WhatsApp-Crypt12-Decrypter, WhatsApp-Key-Generator. Для дальнейшей работы были выбраны следующие решения: для выгрузки зашифрованных резервных копий использовался проект WhatsApp-GD-Extractor-Multithread, для расшифровки резервных копий по имеющемуся ключу – WhatsApp-Key-Generator.

- Получен доступ к резервной копии базы данных msgstore.db.crypt12 приложения WhatsApp непосредственно с Android-устройства следующими способами.

  – Рутирование устройства: выгружены ключ расшифровки, расшифрованная база данных из приватной области памяти Android.

  – Понижение версии приложения WhatsApp на разных версиях Android рутованного и не рутованного устройства: выгружены резервные копии приложения WhatsApp пониженной версии, содержащей сохраненные файлы расшифрованной базы данных и ключа расшифровки актуальной версии приложения WhatsApp.

- Описана структура ключа расшифровки резервной копии базы данных приложения WhatsApp и имитирован WhatsApp-клиент для получения основной части данных ключа, используемой для расшифровки.

- Решение для получения доступа к резервной копии базы данных приложения WhatsApp из облака Google Drive без участия

Android-устройства реализовано на C# и интегрировано в продукт Belkasoft X. Исходный код реализации закрыт. Имеется акт о внедрении результатов дипломной работы в продукт Belkasoft X.

- Проведена апробация получения доступа к резервным копиям базы данных без участия Android-устройства: по логину/паролю от синхронизированного аккаунта Google Drive выгружена резервная копия базы данных, а затем расшифрована с использованием SMS-кода.

- Выступление с докладом на LII международной научной конференции аспирантов и студентов «Процессы управления и устойчивость» (CPS'21).

- Статья готовится к публикации в журнале «Процессы управления и устойчивость».

Приложение А: Результаты обзора решений для выгрузки и расшифровки данных WhatsApp для платформы Android

	Оксиджен Софтвер (Мобильный Криминалист)	Требования	Belkasoft (Belkasoft X)	Требования	Cellebrite (UFED)	Требования	Magnet (AXIOM)	Требования	Elcomsoft (Elcomsoft Explorer for WhatsApp)	Требования
Выгрузка данных из Android-устройства	+	Доступ к устройству	+	Доступ к устройству	+	Доступ к устройству	+	Доступ к устройству	+	Доступ к устройству
Выгрузка данных из Google Drive	+	номер телефона логин/пароль к учетной записи Google Drive	+	номер телефона логин/пароль к учетной записи Google Drive	+	номер телефона логин/пароль к учетной записи Google Drive	+	номер телефона логин/пароль к учетной записи Google Drive	+	номер телефона логин/пароль к учетной записи Google Drive
Выгрузка данных посредством установки приложения через Wi-Fi или SD-карту	+	Разблокированное устройство, андроид от 4.х Авиа-режим	-		-		-		-	
Выгрузка данных через QR-код	+	Доступ к устройству Доступ в интернет	+	Доступ к устройству Доступ в интернет	-		+	Доступ к устройству Доступ в интернет	-	
Расшифровка резервных копий баз данных приложения WhatsApp	+	key-файл	+	key-файл	+	key-файл	+	key-файл	+	key-файл
Генерация ключа расшифровки резервных копий баз данных приложения WhatsApp	+	SIM-карта	-		-		-		+	SIM-карта

Список литературы

[1] Anglano C. Forensic analysis of WhatsApp Messenger on Android smartphones // Digital Investigation. — 2014. — Vol. 11(3). — P. 201–213.

[2] Anti-Malware.ru Антон Макаров специалист по защите информации. Как дешифровать сообщения WhatsApp из резервной копии на Android и iOS // Сайт Anti Malware. URL: https://www.anti-malware.ru/practice/methods/how-data-encrypted-and-decrypted-whatsapp (дата обращения: 10.04.2021).

[3] Cellebrite. CloudAnalyzer 5.2 ReleaseNotes EN // Сайт Cellebrite. URL: https://cf-media.cellebrite.com/wp-content/uploads/2017/07/CloudAnalyzer5.2_ReleaseNotes_EN.pdf (дата обращения: 05.12.2020).

[4] Cellebrite. How Accessing Data from WhatsApp, Instagram, and Facebook on Android Can Help Solve Cases Faster // Сайт Cellebrite. URL: https://www.cellebrite.com/en/blog/how-accessing-data-from-whatsapp-instagram-and-facebook-\on-android-os-can-help-solve-cases-faster/ (дата обращения: 05.12.2020).

[5] Elcomsoft. Elcomsoft Explorer for WhatsApp // Сайт Elcomsoft. URL: https://www.elcomsoft.ru/exwa.html (дата обращения: 05.12.2020).

[6] EliteAndroidApps. WhatsApp Crypt12 Database Decrypter // Сайт GitHub. URL: https://github.com/EliteAndroidApps/WhatsApp-Crypt12-Decrypter (дата обращения: 05.12.2020).

[7] F. Karpisek I. Baggili F. Breitinger. WhatsApp network forensics: Decrypting and understanding the WhatsApp call signaling messages // Digital Investigation. — 2015. — Vol. 15. — P. 110–118.

[8] Magnet. Acquire and decrypt a WhatsApp backup using a recovered decryption key // Сайт Magnet. URL: https://support.magnetforensics.com/s/article/Acquire-and-decrypt-a-WhatsApp-backup-using-a-recovered-\decryption-key (дата обращения: 05.12.2020).

[9] Magnet. Investigating Smartphones with Magnet ACQUIRE: Webinar QA // Сайт Magnet. URL: https://www.magnetforensics.com/resources/investigating-smartphones-magnet-acquire-webinar-qa/ (дата обращения: 05.12.2020).

[10] Magnet. Updates in Magnet AXIOM 4.2 Include Support for AFF4, Skype Warrant Returns, and WhatsApp // Сайт Magnet. URL: https://www.magnetforensics.com/blog/updates-in-magnet-axiom-4-2-include-support-for-aff4-\skype-warrant-returns-and-whatsapp/ (дата обращения: 05.12.2020).

[11] Perrin Trevor. The noise protocol framework // Сайт Noise protocol. URL: https://noiseprotocol.org/ (дата обращения: 23.03.2021).

[12] Statista. Most popular global mobile messenger apps as of October 2020, based on number of monthly active users // Сайт Statista. URL: https://www.statista.com/statistics/258749/most-popular-global-mobile-messenger-apps/ (дата обращения: 02.12.2020).

[13] YuriCosta. WhatsApp Google Drive Extractor // Сайт GitHub. URL: https://github.com/YuriCosta/WhatsApp-GD-Extractor-Multithread (дата обращения: 05.12.2020).

[14] rotbuf. WhatsApp Key/DB Extractor // Сайт GitHub. URL: https://github.com/rotbuf/WhatsApp-Key-DB-Extractor (дата обращения: 05.12.2020).

[15] tgalal. yowsup // URL: https://github.com/tgalal/yowsup (дата обращения: 25.04.2021).

[16] Оксиджен Софтвер. MKDv101 Release Notes // Сайт Оксиджен Софтвер. URL: https://www.oxygensoftware.ru/download/whatsnew/MKDv101ReleaseNotes.pdf (дата обращения: 05.12.2020).

[17] Оксиджен Софтвер. «QR-код» – ключ от данных мессенджеров // Сайт Оксиджен Софтвер. URL: https://www.oxygensoftware.ru/ru/news/articles/271-whatsapp-qr (дата обращения: 05.12.2020).

[18] Оксиджен Софтвер. Угнать WhatsApp за 60 секунд // Сайт Оксиджен Софтвер. URL: https://www.oxygensoftware.ru/ru/news/articles/306-wa-60sec (дата обращения: 05.12.2020).