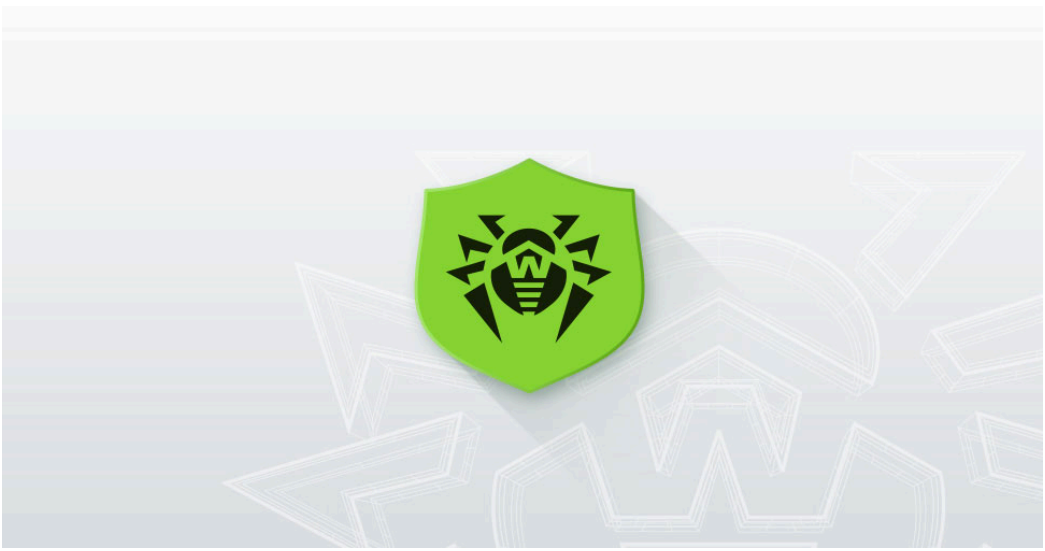
vms.drweb.ru /virus/

# Android.Backdoor.Baohuo.1.origin



sha1:

- 4410f69099a037a25e5976df04a91cee7dbfac14 (org.thunderdog.challegram)

### Описание

Бэкдор для устройств на базе ОС Android, который злоумышленники встроили в копию оригинальной версии мессенджера Telegram X. Он выполняет команды атакующих, позволяет похищать конфиденциальные данные жертв и получать полный контроль над их учетными записями Telegram. **Android.Backdoor.Baohuo.1.origin** распространяется через вредоносные сайты, он также был найден в ряде сторонних каталогов Android-программ.

### Принцип действия

Существует несколько версий **Android.Backdoor.Baohuo.1.origin**, которые отличаются способом внедрения в Telegram X. Основные типы модификаций:

- бэкдор встроен в главный исполняемый DEX-файл мессенджера;
- бэкдор в виде патча динамически внедряется в исполняемый DEX-файл при помощи утилиты LSPatch;
- бэкдор находится в отдельном DEX-файле в каталоге с ресурсами программы и загружается динамически.

Во всех модификациях вызов метода инициализации вредоносного кода расположен в классе ApplicationLoader, поэтому **Android.Backdoor.Baohuo.1.origin** запускается непосредственно при запуске мессенджера. При этом сама программа сохраняет работоспособность и для пользователя выглядит безобидной.

**Взаимодействие с Telegram X**

**Android.Backdoor.Baohuo.1.origin** способен менять функциональность Telegram X на уровне кода, используя фреймворк Xposed и подготовленные злоумышленниками зеркала методов мессенджера. Когда бэкдору необходимо выполнить нестандартное для программы действие (например, скрыть в ее интерфейсе определенные чаты или авторизованные устройства), применяется фреймворк, который динамически меняет функциональность методов.

Если же действие не требует вмешательства в логику работы программы, используются только одни зеркала.

Пример зеркала:

```
com.ucreator.tgjar.reflect.mirror.org.telegram.tgnet.TLRPC.TL_inputChannel
```

Для вызова нужного метода **Android.Backdoor.Baohuo.1.origin** формирует его имя, используя следующий алгоритм:

1. Считывается имя пакета зеркала, которое идет после mirror (для примера выше результатом будет org.telegram.tgnet);
2. Считывается имя зеркалируемого класса (для примера выше результатом будет TLRPC.TL_inputChannel);
3. Итоговое имя возвращается при помощи метода getName() зеркала путем добавления второй строки к первой: org.telegram.tgnet.TLRPC.TL_inputChannel.

Затем с помощью рефлексии создается объект этого метода (выполняется вызов метода).

**Организация управления**

Команды отдаются бэкдору двумя способами:

- через C2-сервер;
- через базу данных Redis.

В более ранних версиях **Android.Backdoor.Baohuo.1.origin** управление выполнялось только через C2-сервер.

Команды и ответы на них передаются в формате JSON.

**Android.Backdoor.Baohuo.1.origin** имеет встроенную конфигурацию, где среди прочего заданы адреса:

- C2-сервера;
- базы данных Redis;
- NPS-сервера.

При запуске бэкдор получает обновленную конфигурацию с текущего C2-сервера, которую он затем использует для соединения с базой данных Redis злоумышленников. После подключения к нему троян получает актуальные адреса C2-сервера и NPS-сервера.

Для связи с C2-сервером и NPS-сервером используются протоколы HTTP и HTTPS.

**NPS-сервер**

NPS-сервер используется для подключения зараженных устройств к внутренней (интранет) сети злоумышленников, что позволяет превратить устройства в прокси для доступа к интернету и перенаправления трафика. Сеть построена на базе проекта https://github.com/ehang-io/nps, а в сам бэкдор добавлена соответствующая клиентская часть.

Для запуска NPS-клиента **Android.Backdoor.Baohuo.1.origin** запрашивает с сервера sdk-nps[.]ips5[.]info конфигурацию с параметрами для подключения к NPS-серверу. На момент анализа бэкдор не выполнял подключение к серверу и получал только тестовую конфигурацию:

```
{
"msg": "\u64cd\u4f5c\u6210\u529f",
"authKey": "TestAuthKey",
"password": "123456",
"code": 0,
"port": "8090",
"ip": "172[.]10.10[.]10",
"user": "user"
}
```

**C2-сервер**

Связь бэкдора с C2-сервером (hpncallback[.]gold5play[.]com) реализована через API-вызовы. С их помощью он передает злоумышленникам собираемую конфиденциальную информацию и сообщает об успешном выполнении команд.
Используются следующие API-вызовы:

- /api/AppCallback/SMS — загрузить на C2-сервер входящие СМС;
- /api/AppCallback/Contacts — загрузить на C2-сервер контакты из телефонной книги;
- /api/Callback/EncryptionData — отправить на сервер содержимое буфера обмена при сворачивании мессенджера и возврате в его окно (для отслеживания события перехватывается метод onResume

приложения);

- /api/Callback/GetLoadParams — получить от C2-сервера URL для демонстрации рекламы, а также адрес сервера для скачивания обновления трояна в виде исполняемого DEX-файла;
- /api/Callback/GetSecretKey — получить ключи шифрования, которые используются при отправке некоторых данных на C2-сервер — например, содержимого буфера обмена;
- /api/Callback/TgCheckReportDataCallback — запросить группу команд для сбора информации об установленных приложениях, об истории сообщений, о контактах из телефонной книги устройства и об устройствах, на которых выполнен вход в Telegram (запрос выполняется каждые 30 минут);
- /api/Callback/TgCheckUpdateApp — запросить у C2-сервера ссылку для скачивания обновления Telegram X. После установки обновления вызывается /api/Callback/TgInstallEventCallback для информирования об успешном выполнении задачи;
- /api/Callback/TgKeepAliveStrategyCallback — запросить у C2-сервера конфигурацию, которая сохраняется в виде JSON-файла. Пример:

```
{"switch1":false,"switch2":false,"switch3":true,"switch4":true,"switch5":false,"intervalTime":3
      0}
```

  Троян использует только переменную intervalTime, которая определяет, через какое время конфигурация будет запрошена повторно;

- /api/Callback/TgRedisStatusChange — запросить информацию о базе данных Redis;
- /api/Callback/TgRegisterPropertyCallback — загрузить на C2-сервер информацию об устройстве (выполняется каждый раз, когда мессенджер отправляет сетевые пакеты);
- /api/Xcallback/GetRobots — получить с C2-сервера список ботов, которые затем добавляются в список контактов Telegram;
- /api/callback/TgHeartCallback — вызывается каждые 3 минуты для передачи на сервер информации о текущих разрешениях приложения, состоянии устройства (включен ли или выключен экран, активно ли приложение), а также номера мобильного телефона с именем и паролем от учетной записи Telegram;
- /api/callback/TgGetTask — вызывается каждую минуту для запроса команды в формате, аналогичном командам от Redis.

**Управление через Redis**

Для получения команд через Redis **Android.Backdoor.Baohuo.1.origin** подключается к соответствующему серверу злоумышленников (159[.]138.237[.]10:33619), где регистрирует свой подканал, привязанный к зараженному устройству.
Злоумышленники подключаются к этому подканалу и публикуют в нем задания, которые бэкдор затем исполняет. Поддерживаемые команды:

- /tg/hideChats/setBlackList и /tg/hideChats/getBlackList — создать черный список чатов, которые не будут отображаться пользователю в интерфейсе Telegram X;
- /tg/hideDevice/setDeviceBlackList и /tg/hideDevice/getDeviceBlackList — скрыть от пользователя заданные устройства в списке авторизованных для его учетной записи;
- /tg/serviceNotifications/startBlock и /tg/serviceNotifications/queryBlock — заблокировать на заданное время отображение уведомлений от чатов, указанных в черном списке setBlackList;
- /tg/dialog/showUpdateApp — показать окно с информацией об обновлении приложения Telegram X — при нажатии на него пользователь перенаправляется на заданный сайт;
- /tg/query/allPackages — отправить на C2-сервер информацию обо всех установленных приложениях;
- /tg/terminated/session — сбросить на зараженном устройстве текущую сессию авторизации пользователя в Telegram;
- /tg/dialog/showInstallApp — показать окно с информацией об обновлении приложения Telegram X, где пользователю предлагается установить APK-файл (если файл отсутствует, троян предварительно скачивает его);
- /tg/hidePremium/setFlag и /tg/hidePremium/getFlag — убрать значок Telegram Premium в интерфейсе приложения у текущего пользователя;
- /tg/db/queryContactsByUsers — загрузить на C2-сервер информацию из базы данных Telegram X, в которой хранятся контакты пользователя;
- /tg/db/queryDialogsByChats — автоматически загрузить на C2-сервер информацию из базы данных Telegram X, в которой хранится история сообщений;
- /tg/db/messagesStorageRawQuery — в соответствии с указанными в команде SQL-запросами загрузить на C2-сервер информацию из базы данных Telegram X, в которой хранится история сообщений;

- /tg/channel/join — подписать пользователя на заданный Telegram-канал;
- /tg/channel/leaveChannel — покинуть заданный Telegram-канал;
- /tg/channel/addByLink — вступить от лица пользователя в Telegram-канал по указанной ссылке;
- /tg/settings/getDevices — получить список устройств, на которых выполнена авторизация в Telegram;
- /tg/captcha/token — запросить получение токена аутентификации пользователя и передать его на C2-сервер.

Пример команды:

```
{"cmd":20000,"path":"/tg/captcha/token","serial_no":"5228e35ac6834e57856a230e507b4b94","callback":"hxxps
{"key_id":"6LflQ8EqAAAAAE3JaczP-
gBVVObsFsSe2U7yZJ6O","action":"signup","currentAccount":0,"resultType":0}
```

Если значение переменной cmd отличается от 20000, команда не будет выполнена.

Значение переменной serial_no является серийным номером команды, который сохраняется перед ее исполнением. Если команда с таким номером уже приходила, устанавливается флаг duplicate, и соответствующая информация отправляется на C2-сервер через API-вызов /api/callback/TgReceptionCommandCallback вместе с информацией об устройстве. Таким образом, переменная используется для информирования об успешном получении бэкдором задания.

Значение переменной path является именем команды. Каждая команда привязана к определенному классу, и, в зависимости от этого имени, бэкдор использует нужный класс.

Значение переменной param представляет собой JSON-объект с параметрами для объекта необходимого класса.

Значение переменной callback — адрес сервера, на который после выполнения команды будет отправлен пакет, информирующий об успехе выполнения команды.

Матрица MITRE

| Этап | Техника |
|---|---|
| Первоначальный доступ | Фишинг (T1660) |
| Выполнение | Командная оболочка Unix (T1623.001) |
| | Планировщик заданий (T1603) |
| | Выполнение по событию (T1624) |
| Закрепление | Широковещательные приемники (T1624.001) |
| | Постоянное закрепление (1541) |
| | Планировщик заданий (T1603) |
| Повышение привилегий | Злоупотребление механизмом контроля повышения привилегий (T1626) |
| | Загрузка нового кода во время выполнения (T1407) |
| | Постоянное закрепление (T1541) |
| Предотвращение обнаружения | Маскировка (T1655) |
| | Использование легитимного имени или пути (T1655.001) |
| | Прокси через жертву (T1604) |
| Получение учетных данных | Доступ к уведомлениям (T1517) |
| | Данные из буфера обмена (T1414) |
| | Отслеживание местоположения (T1430) |
| | Обнаружение ПО (T1418) |
| Обнаружение | Получение информации о системе (T1426) |
| | Получение конфигурации сети системы (T1422) |
| | Получение информации об интернет-соединении (T1422.001) |
| Сбор данных | Доступ к уведомлениям (T1517) |
| | Данные из буфера обмена (T1414) |
| | Данные локальной системы (T1533) |

| Этап | Техника |
| --- | --- |
| | Отслеживание местоположения (T1430) |
| | Защищенные данные пользователя (T1636) |
| | Записи календаря (T1636.001) |
| | Список контактов (T1636.003) |
| | СМС-сообщения (T1636.004) |
| | Протокол прикладного уровня (T1437) |
| Организация управления | Веб-протоколы (T1437.001) |
| | Нестандартный порт (T1509) |
| Эксфильтрация данных | Извлечение на C2-сервер (T1646) |
| Последствия | Потеря доступа к учетной записи (T1640) |

Индикаторы компрометации

Новость о трояне