

что это такое, как работает, технический анализ и конфигурация

 rt-solar.ru/solar-4rays/blog/6472

February 1, 2001



ClayRat — это семейство вредоносного программного обеспечения для Android, относящееся к классу Spyware / RAT (Remote Access Trojan). ВПО предназначено для скрытого наблюдения за пользователем и удаленного управления зараженным устройством.

Вредонос способен перехватывать SMS-сообщения и журнал вызовов, получать доступ к контактам, делать фотографии, запись экрана, а также отправлять команды с сервера управления (C2). Распространяется преимущественно через фишинговые сайты и под видом легитимных приложений.

История ClayRat во многом повторяет судьбу Gorilla: после громкого старта и быстрого масштабирования инфраструктуры проект стремительно угас. Вредонос, который привлек к себе широкое внимание в октябре 2025 года, к декабрю того же года потерял все командные серверы. По информации из открытых источников, активность ClayRat, вероятно, прекратилась после задержания в Краснодаре студента, подозреваемого в его разработке. Как и в случае с Gorilla, «сгорел» проект ярко, но быстро во многом из-за фундаментальных просчетов его автора.

В статьях Zimreium ([1](#) и [2](#)) приводится технический анализ нагрузок и примеры массового распространения вредоноса. Мы сконцентрируемся на анализе серверной части и подведем общий итог активности ClayRat.

Резюме:

- ClayRat — Android Spyware/RAT с функциональностью перехвата SMS, звонков, контактов, скриншотов и удаленного управления устройством.
- Бэкенд написан на Go (1.24), не обфусцирован, содержит дебаг-строки и использует файловое хранение (JSON) вместо СУБД.
- Конфигурация хранится в plaintext: пароли пользователей, токены Telegram/SMS, C2-домены, — что создает дополнительные риски при компрометации панели.
- Фильтрация SMS реализует многоуровневый парсинг: поиск по ключевым словам банков, МФО, маркетплейсов, поиск номеров карт по регулярным выражениям и т. д.
- APK-билдер генерирует вредоносные приложения на основе шаблона, подставляя конфигурируемые параметры.
- Инфраструктура использовала домен krmall[.]su (также фигурирует в индикаторах DCRAT) и массив C2-серверов (~100 IP).
- Финал: к декабрю 2025 года все C2-серверы стали недоступны, в открытых источниках появилась информация о задержании в Краснодаре студента, подозреваемого в создании вредоноса ClayRat.

Технический анализ

Мы обнаружили бэкенд-модуль в процессе анализа инфраструктуры злоумышленников. Он находился внутри архива с именем server.zip (6227833177b3e4390759802c303cb8d51168d1698e113dc700a97b9e43f88ba8), размещенного в открытом доступе, где помимо самого исполняемого файла присутствовали также вспомогательные утилиты для билда и подписи APK, а также шаблон APK-файла, который использовался для генерации вредоносных APK файлов.

Структура проекта

MD5	e99ed610f69eef88046e5a52a638fc3c
-----	----------------------------------

SHA1	6f9920f11bf4bd0b6e08b578201a8a37fff1283c
------	--

SHA256	d6b5ebce1531484f384868c68c1639acc351954b09b7fbb989d2060f2adc0c45
--------	--

name	webpanel_linux_amd64
------	----------------------

Исполняемый файл webpanel_linux_amd64 представляет собой необфусцированное веб-приложение (ELF), написанное на языке Go 1.24.

```
C:/Users/krypton/Desktop/webpanel/server/internal/builders/apk_builder.go
C:/Users/krypton/Desktop/webpanel/server/internal/infofiltering/filtering.go
C:/Users/krypton/go/pkg/mod/github.com/go-telegram-bot-api/telegram-bot-api/v5@v5.5.1/log.go
C:/Users/krypton/go/pkg/mod/github.com/go-telegram-bot-api/telegram-bot-api/v5@v5.5.1/bot.go
C:/Users/krypton/go/pkg/mod/github.com/go-telegram-bot-api/telegram-bot-api/v5@v5.5.1/params.go
C:/Users/krypton/go/pkg/mod/github.com/go-telegram-bot-api/telegram-bot-api/v5@v5.5.1/types.go
C:/Users/krypton/go/pkg/mod/nhooyr.io/websocket@v1.8.17/wsjson/wsjson.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/websocket_commands.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/api.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/auth.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/builder.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/dashboard.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/pages.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/telegram_notifications.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/websocket.go
C:/Users/krypton/Desktop/webpanel/server/internal/handlers/websocket_crypto.go
C:/Users/krypton/go/pkg/mod/github.com/gin-contrib/sessions@v1.0.3/cookie/cookie.go
C:/Users/krypton/Desktop/webpanel/server/main.go
C:/Users/krypton/Desktop/webpanel/server/internal/middleware/auth.go
```

Дебаг-строки

Интересным является то, что пользователь, под которым компилировали данный сервер, пренебрегает дебаг-строками систематически. В открытых коллекциях семплов можно обнаружить различные дропперы данного автора начиная с 2023 года (данный злоумышленник предположительно занимался также распространением вредоноса DCRAT, например c5ae9f00de305435ad8a1f17cd825a576a237c5a116b469d40b994101a7c02a2).

Также приложение использует библиотеку `embed`, предназначенную для хранения вспомогательных файлов внутри исполняемого файла. В данном случае это js-скрипты и html-шаблоны страниц. Для извлечения файлов можно использовать утилиту [Gembe](#).

```

  ✓ static
    ✓ css
      # app.css
      # dashboard.css
    ✓ js
      JS app.js
      JS dashboard.js
      JS devices-page.js
      JS ui-utils.js
      JS vnc-interactions.js
      JS ws-client.js
  ✓ templates
    <> apps.html
    <> builder.html
    <> calls.html
    <> cam.html
    <> contacts.html
    <> dashboard.html
    <> device_detail.html
    <> devices.html
    <> error.html
    <> index.html
    <> login.html
    <> map.html
    <> settings.html
    <> sms.html

```

Вложенные файлы

Frontend-составляющая приложения написана без использования каких-либо фреймворков. Код подробно документирован, это может говорить либо об отличной дисциплине автора данного приложения, либо же о помощи LLM при написании кода.

```

/**
 * Запрашивает список приложений
 * @param {string} clientId - ID клиента
 */
requestAppsList: function(clientId) {
    if (!this.socket || this.socket.readyState !== WebSocket.OPEN) {
        console.error('WebSocket не подключен');
        return false;
    }

    // Устанавливаем глобальный таймаут на ожидание списка приложений
    if (window.appsTimeoutId) {
        clearTimeout(window.appsTimeoutId);
    }

    window.appsTimeoutId = setTimeout(function() {
        if (window.appsRequested && !window.appsReceived) {
            window.appsRequested = false;
            console.warn('Таймаут ожидания списка приложений');
            if (window.displayErrorStatus) {
                window.displayErrorStatus('Не удалось получить данные от устройства (таймаут)');
            }
        }
    }, 60000); // Увеличиваем таймаут до 60 секунд для больших списков приложений

    // Отправляем команду 4 (get_apps)
    return this.sendCommand(clientId, 4);
},

/**
 * Запрашивает кэшированный список приложений для устройства
 * @param {string} deviceId - ID устройства
 */

```

Пример кода, модуль websocket-клиента.

Код разбит на следующие модули:

Модуль	Описание
app.js	Данный модуль отвечает за логику получения информации о приложениях, установленных на устройстве
dashboard.js	Логика управления дашбордом
device-page.js	Логика управления устройством
ui-utils.js	Вспомогательные функции пользовательского интерфейса
vnc-interactions.js	Модуль для получения изображения экрана. Работает поверх протокола веб-сокет

Конфигурация

При первом запуске веб-панель генерирует JSON-файлы конфигурации, списки аккаунтов операторов, а также списки устройств и создается папка для работы с билдами.

```
2026/02/23 08:27:57 Учетные записи пользователей загружены
2026/02/23 08:27:57 Токен Telegram бота не указан в конфигурации. Уведомления в Telegram не будут работать.
[GIN-debug] [WARNING] Running in "debug" mode. Switch to "release" mode in production.
- using env:   export GIN_MODE=release
- using code:  gin.SetMode(gin.ReleaseMode)

2026/02/23 08:27:57 Настроены доверенные прокси для Cloudflare. Правильная обработка IP адресов клиентов через прокси включена.
2026/02/23 08:27:57 [GIN-debug] [WARNING] Creating an Engine instance with the Logger and Recovery middleware already attached.
2026/02/23 08:27:57
2026/02/23 08:27:57 [GIN-debug] [WARNING] Running in "debug" mode. Switch to "release" mode in production.
2026/02/23 08:27:57 - using env:   export GIN_MODE=release
2026/02/23 08:27:57 - using code:  gin.SetMode(gin.ReleaseMode)
2026/02/23 08:27:57
2026/02/23 08:27:57 Ошибка чтения конфигурации билда build_1768922313899: open builds/build_1768922313899/config.json: no such file or directory
2026/02/23 08:27:57 Ошибка чтения конфигурации билда build_1768922354961: open builds/build_1768922354961/config.json: no such file or directory
2026/02/23 08:27:57 Ошибка чтения конфигурации билда build_1768922711128: open builds/build_1768922711128/config.json: no such file or directory
2026/02/23 08:27:57 Ошибка чтения конфигурации билда build_project_687t678t687_1768922354961_1768922354: open builds/build_project_687t678t687_1768922354961_1768922354/config.json: no such file or directory
2026/02/23 08:27:57 Ошибка чтения конфигурации билда build_project_unknown_1768922313899_1768922313: open builds/build_project_unknown_1768922313899_1768922313/config.json: no such file or directory
2026/02/23 08:27:57 Ошибка чтения конфигурации билда build_project_unknown_1768922711128_1768922711: open builds/build_project_unknown_1768922711128_1768922711/config.json: no such file or directory
2026/02/23 08:27:57 Ошибка чтения конфигурации билда com.android.project_687t678t687_1768922354961: open builds/com.android.project_687t678t687_1768922354961/config.json: no such file or directory
2026/02/23 08:27:57 Ошибка чтения конфигурации билда com.android.project_unknown_1768922313899: open builds/com.android.project_unknown_1768922313899/config.json: no such file or directory
2026/02/23 08:27:57 Ошибка чтения конфигурации билда com.android.project_unknown_1768922711128: open builds/com.android.project_unknown_1768922711128/config.json: no such file or directory
2026/02/23 08:27:57 Загружено 0 существующих билдов
[GIN-debug] GET    /static/*filepath --> github.com/gin-gonic/gin.(*RouterGroup).createStaticHandler.func1 (2 handlers)
[GIN-debug] HEAD   /static/*filepath --> github.com/gin-gonic/gin.(*RouterGroup).createStaticHandler.func1 (2 handlers)
[GIN-debug] GET    /                  --> webpanel/internal/handlers.IndexHandler (2 handlers)
```

Запуск происходит по умолчанию в дебаг-режиме

База данных для хранения аккаунтов операторов и сущностей устройств при этом не используется, все происходит через файлы формата .json. Ниже приведен пример конфигурации (файл config.json):

```

{
  "server": {
    "http_port": 8081,
    "auto_start": true,
    "log_all_actions": true
  },
  "security": {
    "username": "admin",
    "password": "admin",
    "enable_2fa": false,
    "lock_after_fails": true
  },
  "relay": {
    "phone_number": "",
    "enable_retransmission": false
  },
  "telegram": {
    "token": "",
    "user_id": "",
    "enable_notify": false,
    "bot_token": "",
    "chat_id": 0
  },
  "sms": {
    "token": "",
    "user_id": "",
    "enable_sms": false
  }
}

```

В конфигурации можно заметить опцию `enable_2fa`, предназначенную для управления двухфакторной аутентификацией, однако в рассматриваемой версии веб-панели она не использовалась.

Пользователи также хранятся в виде JSON-объектов, причем пароли лежат в файле `user_accounts.json` в незашифрованном виде (plaintext). Это говорит о пренебрежении элементарными нормами безопасности при разработке панели.

Интересна и логика прав доступа: хотя в модели есть поле `role` (с ролями `admin`, `operator`, `viewer`), реальное влияние на доступ к вкладкам оказывает объект `permissions`. По сути, поле `role` здесь бесполезно, так как вся авторизация строится на перечислении конкретных разрешений.

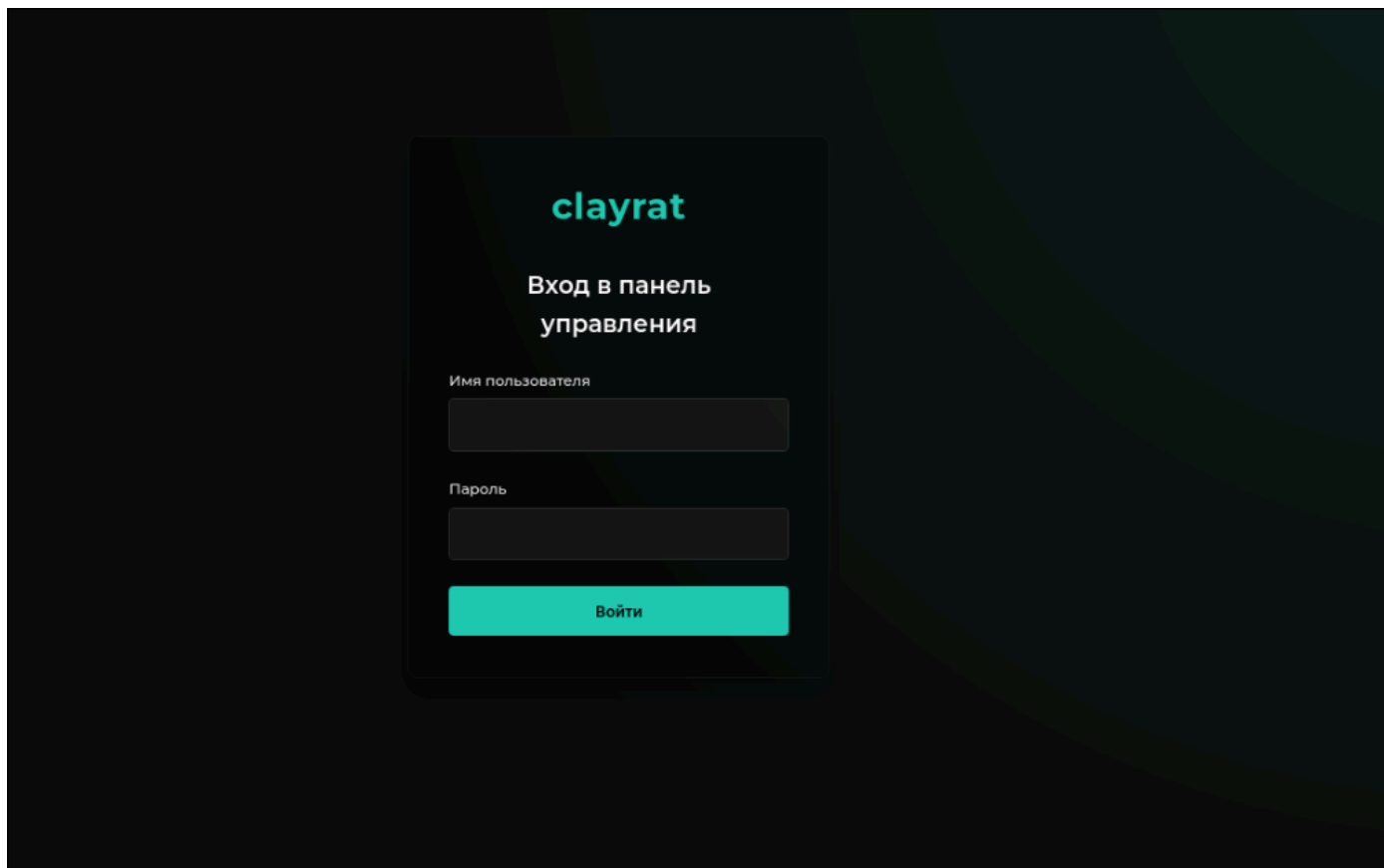
```

{
  "users": [
    {
      "id": "admin",
      "username": "admin",
      "password": "admin",
      "role": "admin",
      "permissions": {
        "access_devices": true,
        "access_builder": true,
        "access_settings": true
      },
      "created_at": "2026-02-23T11:16:01.218837633-06:00",
      "last_login": "2026-02-23T11:57:04.459720899-06:00"
    },
    {
      "id": "user_20260223112514",
      "username": "test",
      "password": "test",
      "role": "operator",
      "permissions": {
        "access_devices": false,
        "access_builder": false,
        "access_settings": false
      },
      "created_at": "2026-02-23T11:25:14.042824508-06:00",
      "last_login": "0001-01-01T00:00:00Z"
    }
  ]
}

```

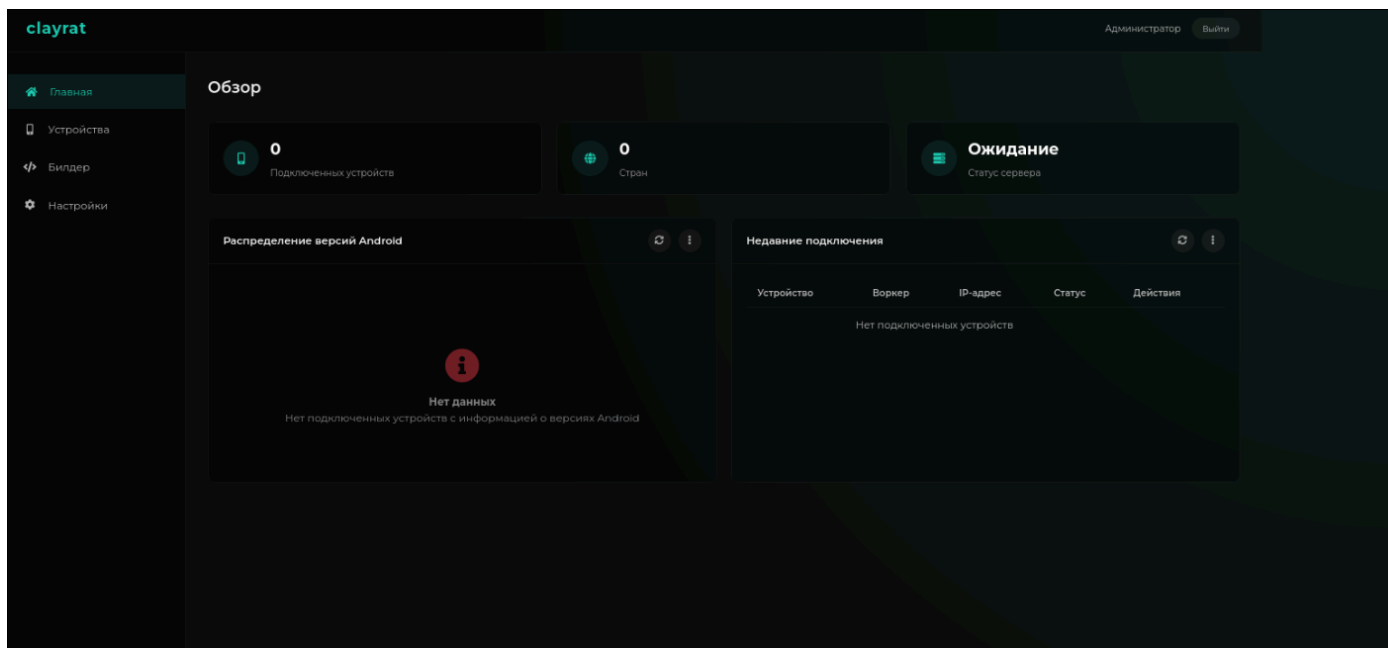
Страница логина и дашборд

Панель управления встречает нас окном логина.



Страница логина

После логина оператор попадает на страницу дашборда, который содержит общую телеметрию о подключенных устройствах, а также статус сервера. В левой части панели располагается набор функциональных вкладок. Всего их три: устройства, билдер, настройки.



Дашборд

Устройства

На данной вкладке оператор может посмотреть информацию обо всех зараженных устройствах. Здесь также доступны действия для управления устройствами, среди которых есть запрос подробной информации об устройстве, запрос списка приложений, доступ к камере, журнал звонков, SMS.

В случае с запросом SMS на стороне сервера есть хендлер для анализа пришедших сообщений. Он осуществляет поиск по регулярным выражениям и ключевым словам сообщений от банков, маркетплейсов, номеров банковских карт.

Устройства

Поиск устройства...

Все устройства

Устройство	Тег	Воркер	IP	Android	Подключено	Статус	Экран	Действия
Samsung Galaxy S21 SM-G991B alb2c3d4e5f6g7h8	Тестовое устройство	Worker-01	192.168.1.105	13	15.01.2024 14:32	ОНЛАЙН	Включен	
Samsung Galaxy S21 SM-G991B alb2c3d4e5f6g7h8	Тестовое устройство	Worker-01	192.168.1.105	13	15.01.2024 14:32	ОНЛАЙН	Включен	

Управление устройствами

Во вкладке журнала звонков предусмотрена функциональность совершения звонка с указанного оператора на заданный номер без перехвата аудиопотока. Потенциально такая функциональность могла бы использоваться для так называемого toll fraud (совершение звонков на платные номера со списанием средств) или для одного из мошеннических сценариев, когда сначала с телефона жертвы инициируется звонок одному из контактов (например, родственнику), а затем происходит взаимодействие посредством текстового общения.

```

        // Устанавливаем класс selected для первого оператора
        firstOption.classList.add('selected');

        console.log("Автоматически выбран оператор:", firstOption.textContent);
    }
}

// Обработчик для кнопки звонка
const callButton = document.getElementById('make-call-btn');
const phoneNumberInput = document.getElementById('call-to-number');
const operatorInput = document.getElementById('call-operator');

if (callButton && phoneNumberInput && operatorInput) {
    callButton.addEventListener('click', () => {
        const number = phoneNumberInput.value.trim();
        const operatorName = operatorInput.value;

        if (!number) {
            CallsClient.showToast('Введите номер телефона', 'error');
            return;
        }

        CallsClient.makeCall(number, operatorName);
    });
}
});

```

Совершение звонков

При переходе на вкладку детальной информации об устройстве можно получить подробности. Также можно сделать скриншот экрана, просмотреть экран, получить информацию об устройстве, обновить информацию об устройстве, отключить устройство от сервера.

[← Вернуться к списку устройств](#)

Детали устройства



Pixel 7 Pro

GP4BC

ID: 8f3a9b2c-1d4e-5f6g-7h8i-9j0k1l2m3n4o

● ОНЛАЙН

TAG

📷👁️🔍🔄⚙️

Общая информация

Системная информация

Информация о соединении

📱 Информация об устройстве

Название устройства	Pixel 7 Pro
Модель	GP4BC
Производитель	Google
SIM 1	Tele2 - +79001112233
SIM 2	Megafon - +79004445566

🖥️ Системная информация

Android	14
Заряд батареи	42%

🌐 Информация о сети

IP-адрес	10.0.0.55
Тип подключения	Mobile Data
Статус	Подключено

Подробная информация об устройстве

Общая информация

Системная информация

Информация о соединении

⚙️ Дополнительная информация

ID устройства	8f3a9b2c-1d4e-5f6g-7h8i-9j0k1l2m3n4o
Номер сборки	AP2A.240605.024

👤 Информация о билде

Воркер	worker-msk-03
ТГ Воркера	@worker_example
Вебвью	https://google.com

Вкладка «Системная информация» (тестовые данные)

12/23

 История подключений

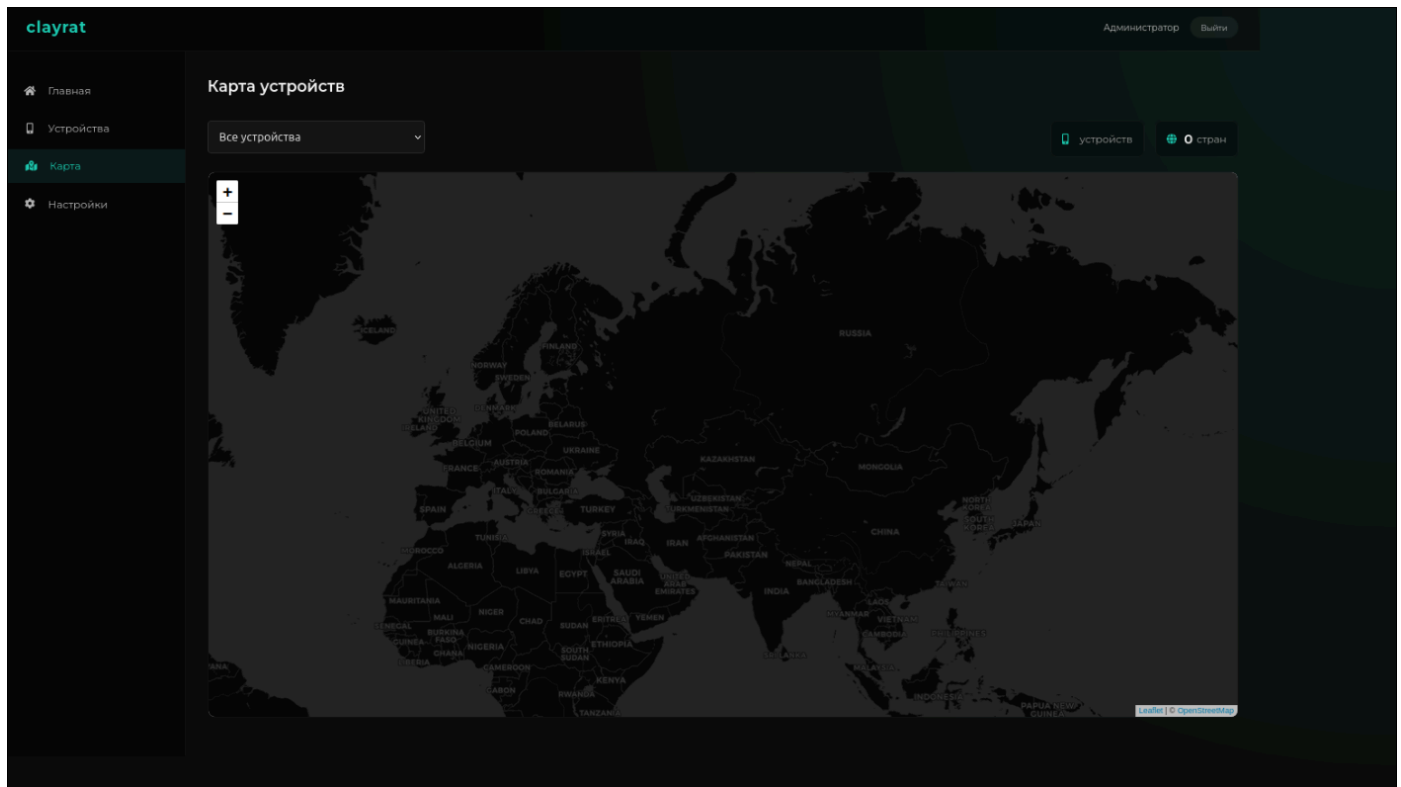
Первое подключение	15.01.2024 09:30:00
--------------------	---------------------

Последнее подключение	17.01.2024 18:45:00
-----------------------	---------------------

Длительность подключения	2 дня 9 часов 15 минут
--------------------------	------------------------

Информация о соединении (тестовые данные)

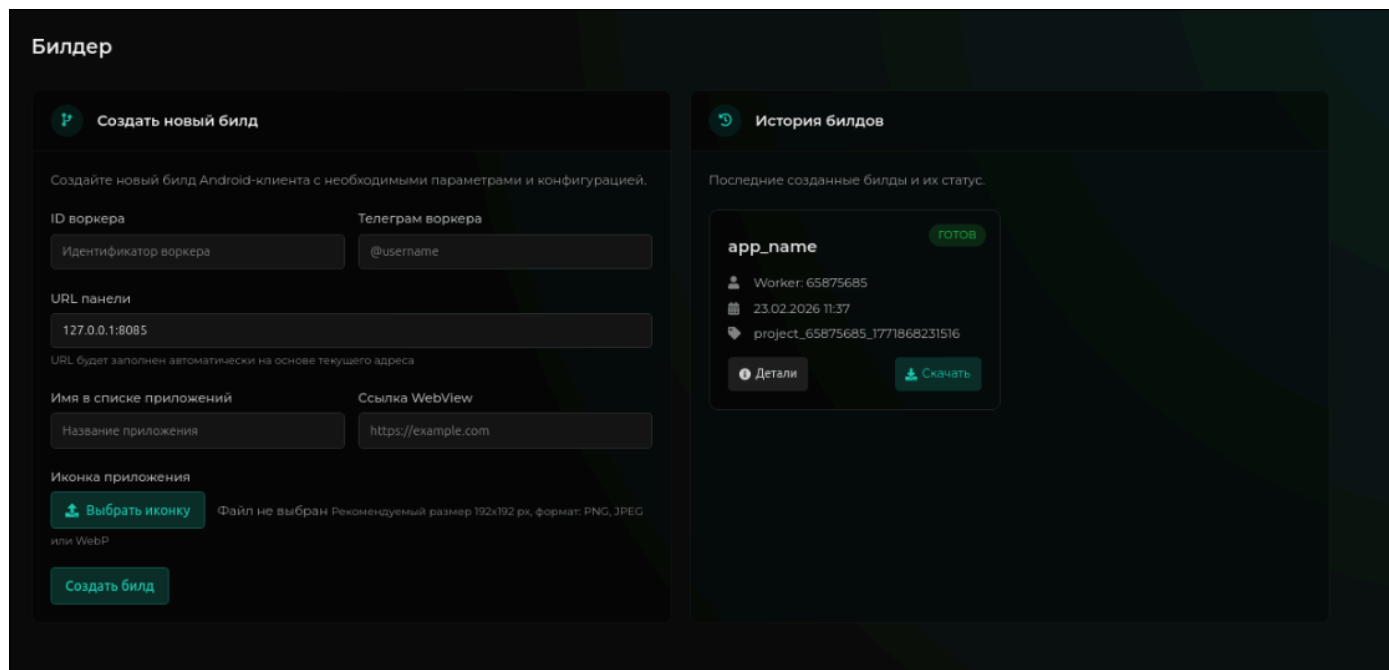
Для вкладки «Устройства» предусмотрена, но в данной версии не реализована до конца возможность просмотра «карты устройств». При переходе на нее предполагалось, что поверх карты openstreetmap будет примерно помечаться местоположение устройств в виде маркеров. При нажатии на один из маркеров должно было появляться всплывающее окно с основной информацией о данном устройстве.



Карта устройств

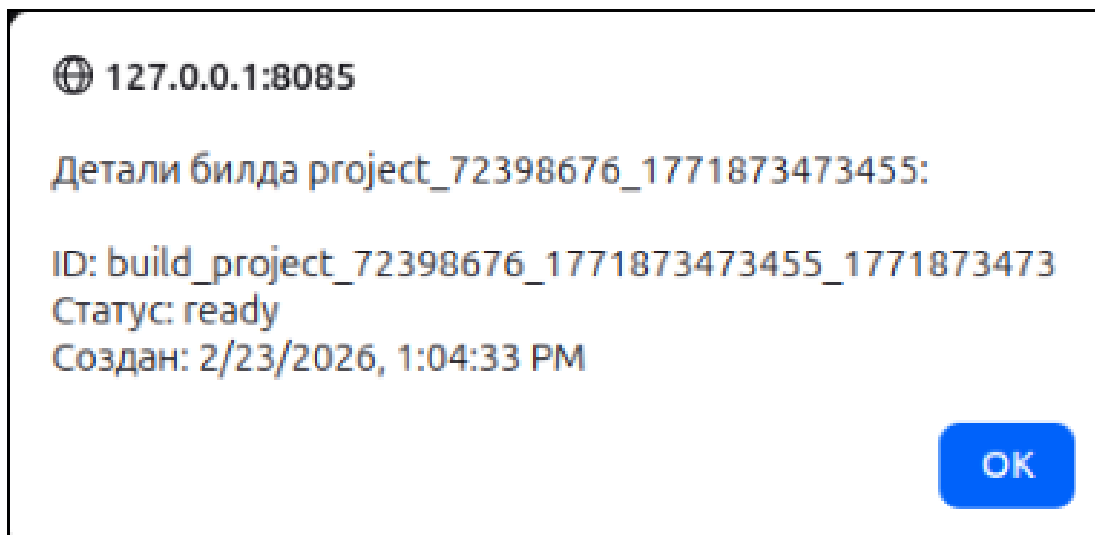
Билдер

Эта вкладка позволяет собрать вредоносный APK с нужными оператору параметрами. Среди конфигурируемых полей: идентификатор воркера, контакт воркера в Telegram, URL веб-панели, имя приложения (отображаемое), а также ссылка WebView, которая будет открыта доступным на смартфоне браузером при запуске приложения. Необходимо также сконфигурировать иконку приложения.



Вкладка «Билдер»

При нажатии на кнопку «Детали» отображается подробная информация о билде, а в папке builds появляется созданный .apk.



Детали билда

Генерация вредоносных APK

Генерация новых .apk происходит на основе шаблона приложения. Внутри него помимо ассетов находятся уже готовые классы в виде smali-кода. При этом в приложении пытались создать примитивную обфускацию, добавив туда бесполезные классы со случайными именами.

```
✓ res / app
  > assets
  > kotlin
  > lib
  > original
  > res
✓ smali
  > A
  > a.1
  > A0
  > a0.1
  > android
  > androidx
  > B
  > b.1
  > B0
  > b0.1
  > balanda
  > bandura
  > C
  > c.1
  > C0
  > c0.1
  > com
  > D
  > d.1
  > D0
  > d0.1
  > E
  > e.1
  > E0
  > F
```

```
> f.1  
> F0  
> f0.1
```

Структура шаблона

После применения конфигурации в интерфейсе бэкэнд подставляет необходимые параметры в исходный код, после чего собирает вредоносный APK-файл с помощью утилиты apktool. Рассмотрим фрагмент класса, содержащего конфигурационные константы приложения:

```
.field public static final BASE_URL:Ljava/lang/String; = "clay.kpmail.su"  
.field public static final DEVICE_ID_KEY:Ljava/lang/String; = "device_id"  
.field public static final ID_TIMESTAMP_KEY:Ljava/lang/String; = "id_timestamp".field  
public static final NATIVE_LIB_DOWNLOAD_URL:Ljava/lang/String; = "http://clay.kpmail.su/"  
.field public static final NATIVE_LIB_NAME:Ljava/lang/String; = "wsclient"  
.field public static final VERSION_BUILD:Ljava/lang/String; = "clayrat 1.0.0"  
.field public static final WEBSOCKET_URL:Ljava/lang/String; =  
"ws://clay.kpmail.su/ws/android"  
.field public static final WEB_URL:Ljava/lang/String; = "https://mrqz.me/pomogivpoiske?  
yclid=14905483089039327231"  
.field public static final WORKER_NAME_KEY:Ljava/lang/String; = "GambllerYMB"  
.field public static final WORKER_TG_ID_KEY:Ljava/lang/String; = "6665344429"
```

Как видно из приведенного фрагмента, в исходном коде присутствуют значения по умолчанию. Так, константа VERSION_BUILD указывает на текущую версию ClayRat — в данном случае 1.0.0.

В качестве C2-сервера по умолчанию в шаблоне задан домен clay[.]kpmail[.]su. Примечательно, что домен kpmail[.]su фигурирует не только в инфраструктуре ClayRat: он также встречается в индикаторах компрометации бэкдора DCRAT, где ему соответствует поддомен dcrat[.]kpmail[.]su.

Отдельного внимания заслуживает константа WORKER_NAME_KEY, содержащая, по всей видимости, никнейм оператора ClayRat. На момент проведения анализа пользователь GambllerYMB (зарегистрированный на анонимный номер) все еще был активен в Telegram и имел следующий профиль:



Moral | UZB Traff 

 last seen today at 6:52 PM

Message

Mute

Call

More

🎵 ROUSEMEBABY – Poppin >

+888 0735 1294

Mobile

Продам трафик узб живые заявки Is

Bio

@GambllerYMB

Username 

Closed opens in 14 hours ▼

Business hours

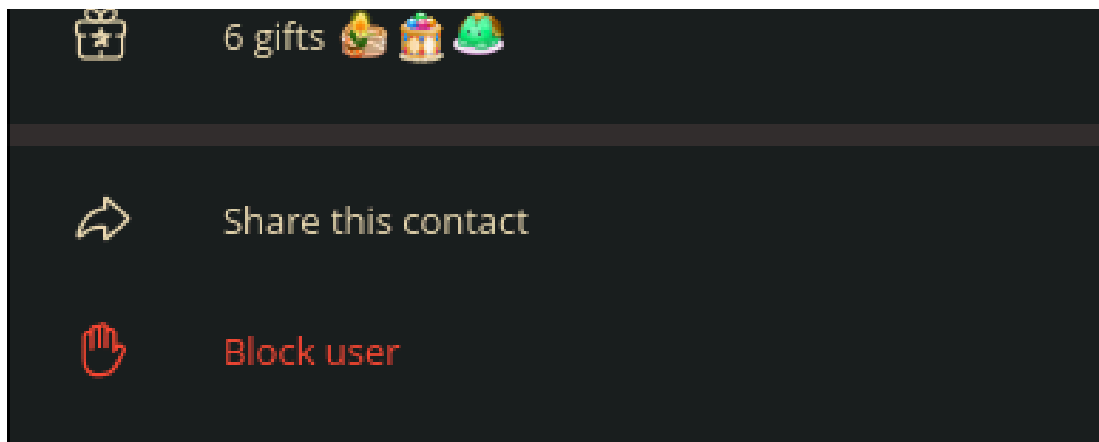
SHIS QI 13 CONJUNTO 13, Бразилиа

Location

ADD TO CONTACTS

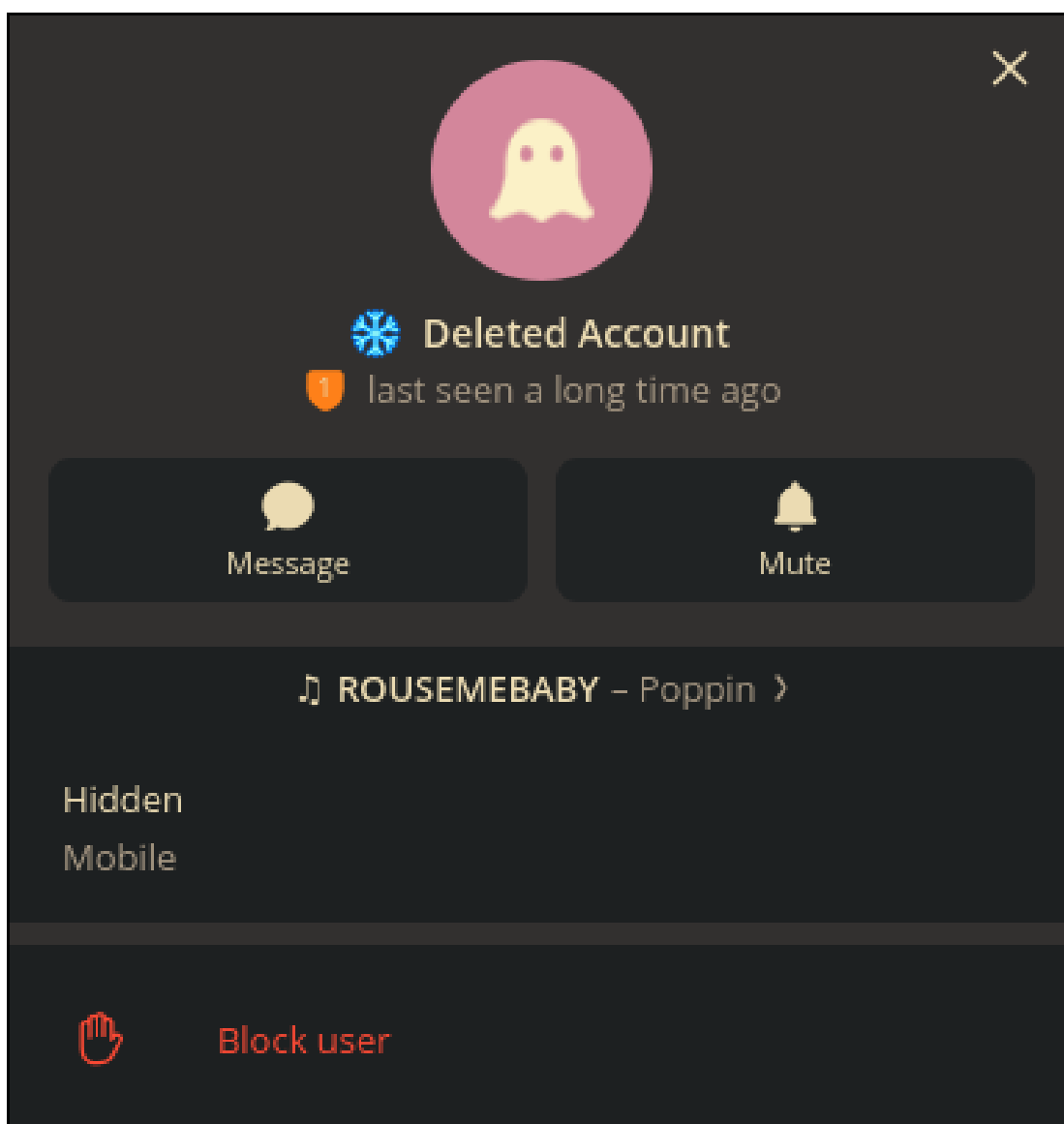


4 posts



Аккаунт оператора

Спустя некоторое время данный аккаунт был удален.



На февраль 2026 года аккаунт удален

Помимо конфигурируемых значений, в коде присутствуют также и жестко заданные. Так, в классе MainActivity задано константное значение `ERROR_REPORT_URL` (`http[:]//error[.]clayhusas[.]sbs[:]5654/error`), которое содержит URL для отправки телеметрии.

Настройки

Данная вкладка позволяет изменять настройки, хранимые в файле `config.json`. Среди них:

- Базовые настройки (порт, логирование)
- Функция ретрансляции. Позволяет осуществлять ретрансляцию SMS-сообщений на заданный номер.
- Отстук через telegram-бота
- Пересылка SMS (вероятно, подразумевается использование внешнего сервиса)
- Управление пользователями

Настройки системы

Настройки сервера

Основные настройки сервера для подключения клиентов. Изменение порта требует перезапуска сервера.

HTTP Порт

8081

Автоматически запускать сервер

Логировать все действия клиентов

Сохранить

Настройки ретранслятора

Основные настройки ретранслятора сигналов.

Номер телефона

+79001234567

Включить Ретранслятор

Сохранить

Отстук telegram

Telegram API токен

123456/ABC-DEF1234ghikl-zyx57W2v1u123ew11

Telegram User ID

12345678

Включить уведомления Telegram

Сохранить

Пересылка SMS

Настройка пересылки SMS-сообщений.

SMS токен

Введите токен для SMS сервиса

SMS User ID

Введите идентификатор пользователя

Включить пересылку SMS

Сохранить

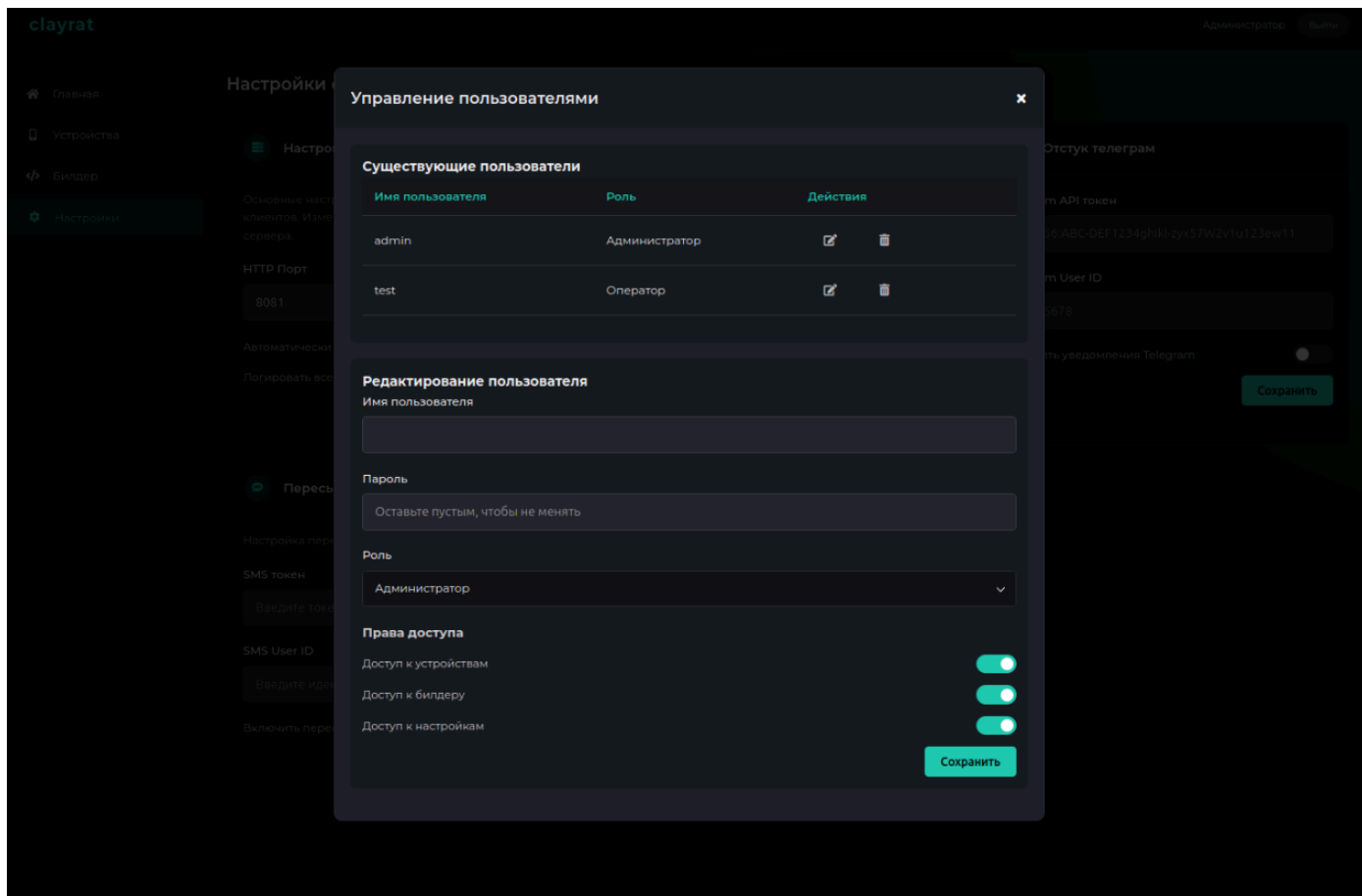
Управление пользователями

Управление пользователями системы и их правами доступа.

Управление пользователями

Вкладка настроек

При нажатии на кнопку «Управление пользователями» появляется всплывающее окно, в котором можно добавлять новых, удалять или редактировать существующих пользователей. Как упоминалось ранее, список пользователей при этом сохраняется в `.json`-файл.



Создание/редактирование пользователей

Финал истории и дальнейшие перспективы

Здесь все происходило почти в точности как с [Gorilla](#). В начале активности мы фиксировали множество новых командных серверов. В процессе стабилизации инфраструктуры злоумышленников скорость появления новых серверов снижалась, пока рост окончательно не прекратился. На данный момент все серверы недоступны.

В декабре 2025 года в публичном поле появилась новость о задержании в Краснодаре разработчика, подозреваемого в создании вредоноса ClayRat, что говорит о неутешительных для злоумышленников перспективах ClayRat.

В Краснодаре задержан разработчик вируса ClayRat для Android

Текст: Ольга Иванова

Программиста, подозреваемого в создании вредоносной программы ClayRat и управлявшего 144 доменами, задержали в Краснодаре по обвинению в киберпреступлениях.

Разработчик вредоносной программы семейства ClayRat был задержан сотрудниками УБК МВД России совместно с ОБК УМВД России по Кировской области и при содействии коллег из Краснодарского края, говорится в [Telegram](#)-канале «Вестник Киберполиции России». По версии следствия, студент-программист из Краснодара использовал свои технические навыки для создания .apk-приложения, которое распространялось как вирус для смартфонов.

Сообщается, что функционал вредоносной программы ClayRat предоставлялся через подписку на различных условиях – 90 долларов в неделю, 300 долларов в месяц или 15% от прибыли пользователя от использования программы. Для работы платформа требовала отдельную инфраструктуру, и злоумышленник зарегистрировал 144 доменных имени для администрирования процесса.

Правоохранители напомнили: любые эксперименты с разработкой или тестированием вредоносных программ могут привести к уголовной ответственности по статье 273 УК РФ – речь идет не только о реальном вреде, но и о попытке создания таких программ.

ADVERTISEMENT

 Запад оторопел: что Зеленский устроил после атаки на резиденцию Путина

 Потеряли равновесие? Разбираемся, что пошло не так

 Бойцы ВС РФ сразу не поверили: вот что бросили ВСУ в Гуляйполе

[Источник](#) WASTED

Заключение

ClayRat громко заявил о себе на старте. Однако при ближайшем рассмотрении в проекте обнаруживается множество фундаментальных недостатков: пароли в plaintext, дебаг-строки в сборке сервера, примитивная обфускация, хранение данных в JSON-файлах вместо СУБД и повторяющаяся инфраструктура. Эти просчеты выдают невысокую квалификацию разработчика и пренебрежение базовыми практиками операционной безопасности.

В итоге, несмотря на амбициозную функциональность, ClayRat повторил судьбу многих «быстрых» Android-RAT: после кратковременного всплеска активности инфраструктура деградировала, сам проект, по имеющимся данным, прекратил развитие, а его создатели теперь преследуются правоохранительными органами.

IOCs

Сетевые индикаторы

C2

999[.]claydc[.]top
armenin[.]clay[.]rest
atom[.]clayrat[.]top
babynot[.]clay[.]rest
billy[.]clayhusas[.]sbs
burger[.]clayfenrirhuy[.]top
cash[.]clayrat[.]top
cc[.]claysrat[.]top
ded[.]clayratnik[.]top
dragonball[.]cfd
dragonball[.]rest
dragonball[.]sbs
gelya[.]claysrat[.]top
igro[.]clayratnik[.]top
korobok[.]site
liker[.]clay[.]rest
mryes[.]clayfenrirhuy[.]top
n1[.]claysrat[.]top
phobia[.]clay[.]rest
pidoras[.]claysrat[.]top
pupkin[.]clayratnik[.]top
scodex[.]clayratnik[.]top
shiza[.]clayratnik[.]top
slavik[.]clay[.]rest
stepan[.]clayratnik[.]top
swaga[.]claydc[.]top
swapn[.]claydc[.]top
хаpaем[.]clayratnik[.]top

Крис Кельвин

Содержание статьи

Получать новости

-
-

Аналитический отчет по целевым кибератакам за 2025 год

[Читать](#)