**Задание 1.** Обзор программных решений для беспроводных самоорганизующихся сетей

**1.1. B.A.T.M.A.N. (Better Approach To Mobile Ad-hoc Networking)**

**Описание:**  
B.A.T.M.A.N. представляет собой протокол маршрутизации для Mesh-сетей, где каждый узел выполняет роль клиента и маршрутизатора. Его основная цель — децентрализованное построение оптимальных маршрутов между узлами.

**Технические особенности:**

* Поддержка динамического подключения и отключения узлов.
* Использует собственные маршруты вместо таблиц маршрутизации операционной системы.
* Работает на уровне канального взаимодействия (L2), что делает его независимым от IP-адресации.

**Характерные черты:**

* Автоматическая маршрутизация сохраняет связь даже при отключении промежуточных узлов.
* Активно применяется в локальных сообществах для беспроводного интернета, таких как Freifunk в Германии.

**Платформы и установка:**

* Работает на Linux.
* Может быть интегрирован в прошивки OpenWrt.

**1.2. OLSR (Optimized Link State Routing Protocol)**

**Описание:**  
OLSR — это проактивный протокол маршрутизации для ад-хок сетей. Он регулярно обновляет информацию о состоянии соединений, поддерживая актуальность маршрутов.

**Технические особенности:**

* Для оптимизации передаёт только часть топологической информации, уменьшая нагрузку на сеть.
* Совместим с фиксированными и мобильными узлами.

**Характерные черты:**

* Эффективен в сетях с большим числом узлов благодаря минимальному объёму служебного трафика.
* Подходит для устройств с ограниченными ресурсами.

**Платформы и установка:**

* Доступен для Windows, Linux и macOS.

**1.3. Serval Mesh**

**Описание:**  
Serval Mesh — это приложение для создания локальной Mesh-сети между устройствами без интернета. Оно позволяет обмениваться сообщениями, звонить и отправлять файлы через Wi-Fi.

**Технические особенности:**

* Использует технологию Wi-Fi Direct для построения локальной сети.
* Не требует центрального маршрутизатора или точки доступа.
* Обеспечивает шифрование для защиты данных.

**Характерные черты:**

* Простое в установке и использовании на мобильных устройствах.
* Полезно для связи в чрезвычайных ситуациях, например, в условиях стихийных бедствий.

**Платформы и установка:**

* Поддерживается на Android.
* Установка возможна через Google Play или с помощью APK-файлов.

**1.4. Briar**

**Описание:**  
Briar — это программное обеспечение с открытым исходным кодом, предназначенное для создания безопасных сетей через Bluetooth, Wi-Fi или интернет. Оно обеспечивает обмен сообщениями, передачу файлов и создание форумов без центральных серверов.

**Характерные черты:**

* **Децентрализация:** сообщения хранятся исключительно на устройствах пользователей.
* **Шифрование:** данные защищены с помощью сквозного шифрования.
* **Работа без интернета:** синхронизация возможна при установлении локальной связи.

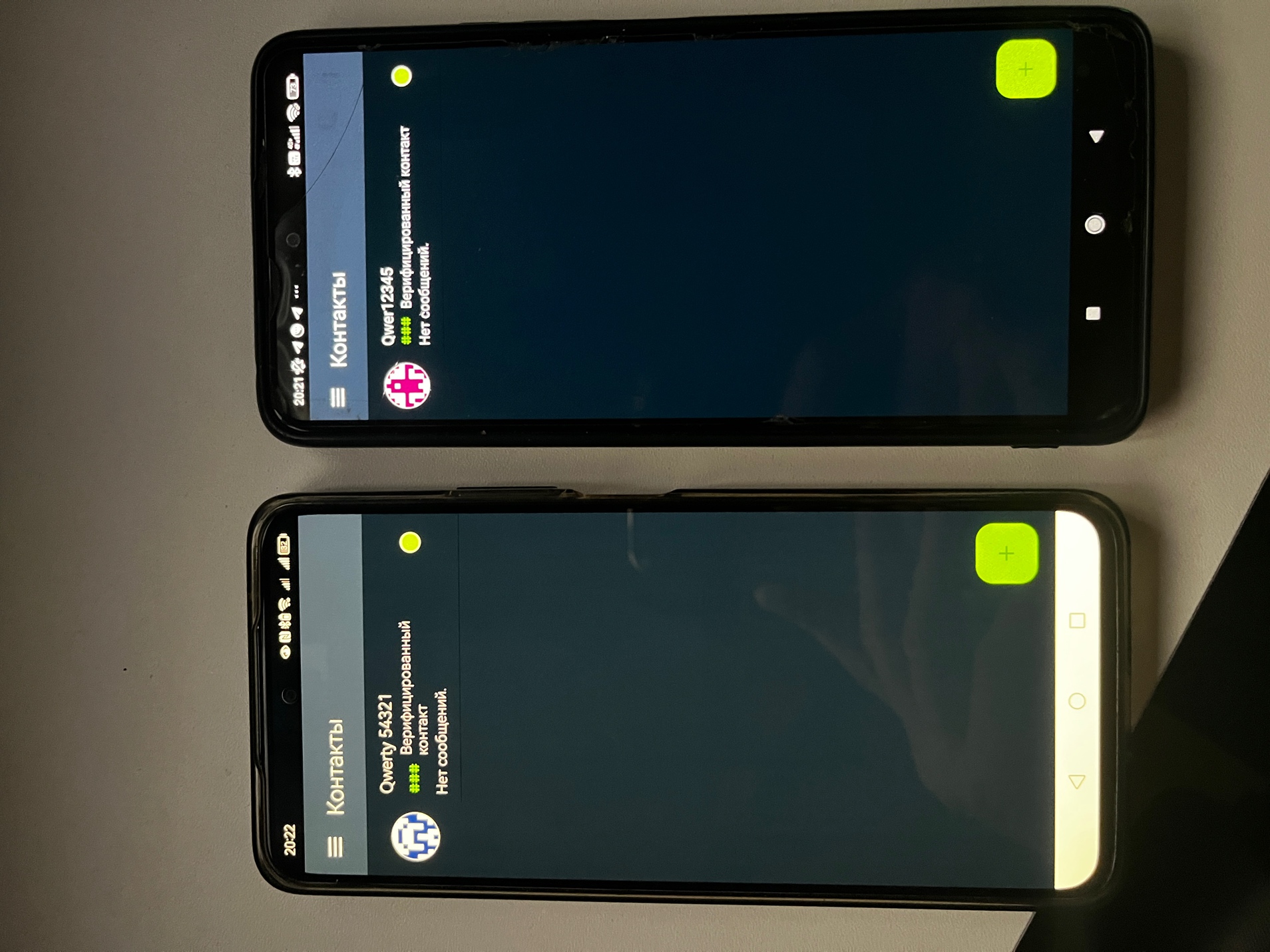
**Платформы и установка:**

* Работает на Android.
* Установка доступна через Google Play или альтернативные источники.

**Задание 2**. Выбрать один из этих пакетов и установить у себя на телефоне,

попробовать связаться с соседями из группы.

**Был использован Briar**.



**Задание 3**. Выбрать один из этих пакетов и установить у себя на телефоне,

попробовать связаться с соседями из группы.

**Обзор API для Интернета вещей на macOS**

**1. HomeKit (Apple)**

**Описание:**  
HomeKit — это API от Apple для работы с умными устройствами в экосистеме "умного дома". Используется для управления IoT-устройствами, таких как умные лампы, термостаты, замки и камеры, через приложение "Дом" на macOS, iOS и других устройствах Apple.

**Особенности:**

* Простая интеграция с macOS через Swift и Objective-C.
* Поддерживает автоматизацию и управление через Siri.
* Высокий уровень безопасности благодаря сквозному шифрованию и проверке устройств.

**Пример сценария использования:**  
Разработка приложения для управления "умным домом", которое может включать/выключать свет или проверять состояние замка.

**2. MQTT (Message Queuing Telemetry Transport)**

**Описание:**  
MQTT — это легковесный протокол обмена сообщениями, популярный в IoT. Позволяет macOS-приложениям подключаться к устройствам через брокеры MQTT для отправки и получения данных.

**Особенности:**

* Работает на macOS через библиотеки на Python, JavaScript (Node.js), Swift и других языках.
* Поддерживает сценарии реального времени, такие как мониторинг датчиков или управление устройствами.
* Энергоэффективный, подходит для работы с устройствами с ограниченными ресурсами.

**Пример сценария использования:**  
Написание программы, которая собирает данные с датчиков температуры и отправляет уведомления, если показатели выходят за пределы нормы.

**Библиотеки:**

* Python: paho-mqtt
* Node.js: mqtt.js

**3. AWS IoT Core**

**Описание:**  
AWS IoT Core — это облачный сервис, который позволяет подключать IoT-устройства к облаку для управления, сбора данных и аналитики. На macOS доступен через SDK для Python, Swift и JavaScript.

**Особенности:**

* Поддержка управления устройствами и обработки данных.
* Интеграция с другими сервисами AWS (например, Lambda, DynamoDB).
* Возможность масштабирования IoT-проектов.

**Пример сценария использования:**  
Создание приложения для macOS, которое собирает данные с устройств, передает их в AWS IoT Core для анализа и отображает результат на панели управления.

**4. Google Cloud IoT**

**Описание:**  
Google Cloud IoT предоставляет инструменты для подключения и управления устройствами IoT. На macOS вы можете использовать его через библиотеки Python, Node.js или Java.

**Особенности:**

* Поддержка устройств через MQTT и HTTP.
* Интеграция с BigQuery и другими сервисами Google для анализа данных.
* Высокая производительность и масштабируемость.

**Пример сценария использования:**  
Мониторинг состояния умных устройств с возможностью анализа данных в режиме реального времени через Google BigQuery.

**5. Bluetooth Low Energy (BLE)**

**Описание:**  
BLE — это технология, широко используемая в IoT для подключения устройств с низким энергопотреблением. На macOS можно взаимодействовать с BLE-устройствами через **Core Bluetooth** API.

**Особенности:**

* Поддерживает сканирование и подключение к BLE-устройствам.
* Позволяет управлять датчиками, маячками и другими BLE-устройствами.
* Полная интеграция в экосистему macOS через Swift и Objective-C.

**Пример сценария использования:**  
Разработка macOS-приложения для сбора данных с фитнес-браслетов или медицинских устройств.

**6. Node-RED**

**Описание:**  
Node-RED — это визуальная среда для разработки IoT-приложений, которая поддерживает macOS. Она позволяет создавать потоки данных и управлять устройствами через графический интерфейс.

**Особенности:**

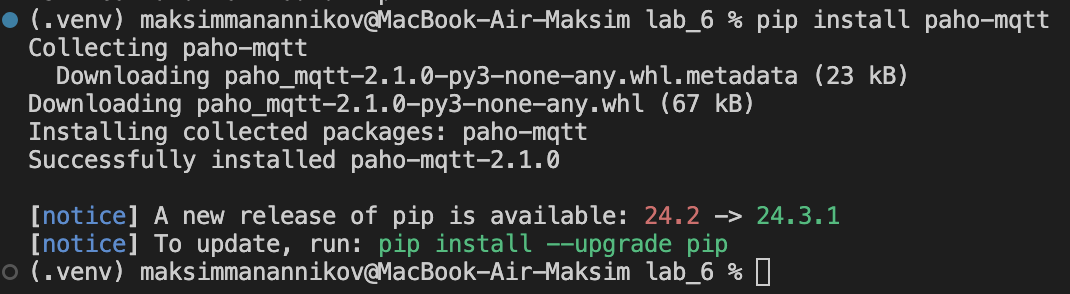
* Простая настройка и разработка без глубоких знаний программирования.
* Поддерживает MQTT, HTTP, WebSocket и другие протоколы.
* Может использоваться для управления IoT-устройствами и обработки данных.

**Пример сценария использования:**  
Создание визуального дашборда для управления IoT-устройствами в сети.

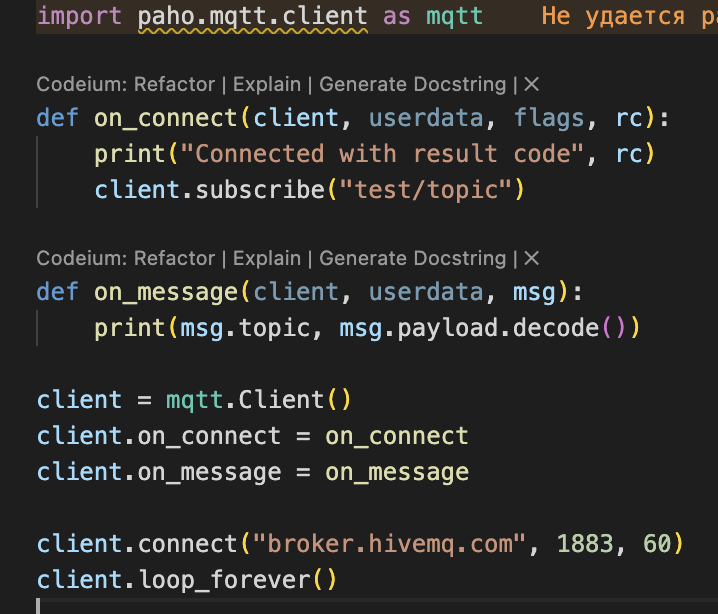
**Задание 4**. Попытаться инсталлировать один из API на своем рабочем месте

В качестве примера установим MQTT API.

В Python установим библиотеку Paho-MQTT командой: **pip install paho-mqtt**



Далее представлен простой код подключения к публичному MQTT-брокеру и отправки/получения сообщений для демонстрации работы API:



Результат выполнения:

