

F.E.A.R. Project (Fully Encrypted Anonymous Routing)



F.E.A.R. — это кроссплатформенный мессенджер с открытым исходным кодом, разработанный для обеспечения максимальной конфиденциальности и безопасности. Все сообщения проходят сквозное шифрование (End-to-End Encryption, E2EE), а маршрутизация реализована с использованием алгоритмов, обеспечивающих анонимность отправителя и получателя.

License **MIT** Platform **Linux | Windows**

Ключевые особенности

- **Сквозное шифрование (E2EE):** Все сообщения шифруются на устройстве отправителя и расшифровываются только на устройстве получателя. Даже сервер не имеет доступа к содержимому ваших переписок.
- **Анонимная маршрутизация:** Используются передовые методы для сокрытия IP-адресов и метаданных, затрудняя отслеживание и анализ трафика.
- **Кроссплатформенность:** Одинаково работает на всех популярных desktop и mobile операционных системах.
- **Открытый исходный код:** Прозрачность и безопасность. Каждый может проверить код, аудировать его и внести свой вклад в развитие проекта.
- **Самохостинг:** Вы можете развернуть свой собственный сервер для полного контроля над инфраструктурой.

Быстрый старт

Алгоритм работы

User1 User2 договариваются начать общение в чате.

1. User1 запускает fear.exe и генерирует пароль для входа в комнату. `./fear genkey`

Ты увидишь строку вида:

Room key (base64 urlsafe, save/share securely): `z6aK3_k9I7rmpy6Sn-84QZ9Yc0p3T7VhzReWCKE0x4I`

2. User1 запускает key-exchanger.exe, выбирает отправить ключ и генерирует `p, g, pub_key1`.
3. User1 сообщает `p, g, pub_key1` User2.
4. User2 запускает key-exchanger.exe, выбирает сгенерировать публичный ключ, вводит полученные `p, g, pub_key1` и генерирует свой `pub_key2`.
5. User2 сообщает `pub_key2` User1.
6. User1 вводит `pub_key2` и пароль для входа в комнату. В итоге он получает зашифрованный пароль.
7. User1 сообщает зашифрованный пароль User2.
8. User2 запускает key-exchanger.exe, выбирает расшифровать ключ, вводит полученные `p, g, pub_key1, secret key`, зашифрованный пароль. В итоге он получает зашифрованный пароль для входа в комнату.
9. Один из User или кто то другой должен запустить сервер fear.exe и сообщить IP-адрес и порт User'ам. ВНИМАНИЕ! сервер должен быть проверенным и ему должно быть можно доверять! Не использовать неизвестные сервера! `./fear server --port 7777`
10. User1 создает комнату на сервере серверу с именем testroom, паролем для входа и указывает свой ник. `./fear client --host 127.0.0.1 --port 7777 --room testroom --key z6aK3_k9I7rmpy6Sn-84QZ9Yc0p3T7VhzReWCKE0x4I --name User1`
11. User2 подключается к комнате на сервере серверу с именем testroom, паролем для входа и указывает свой ник. `./fear client --host 127.0.0.1 --port 7777 --room testroom --key z6aK3_k9I7rmpy6Sn-84QZ9Yc0p3T7VhzReWCKE0x4I --name User2`
12. Общение.
13. В конце просто отключаются от программы `Ctrl + C`.

Предварительные требования

Перед компиляцией убедитесь, что у вас установлены:

- **Git**
- **CMake** (версия 3.15 или выше)
- **Компилятор с поддержкой C++17** (GCC, Clang, MSVC)
- **Qt Framework** (версия 6.2 или выше) - для графического интерфейса
- **vcpkg** или **Conan** (рекомендуется для управления зависимостями)

Сборка проекта из исходного кода

1. **Клонируйте репозиторий:**

```
git clone https://github.com/shchuchkin-pkims/fear.git
cd fear
```

2. Соберите проект

```
make
```

Собранные исполняемые файлы появятся в папках.

Запуск сервера

Серверная часть должна быть запущена на машине, доступной всем клиентам. Это может быть персональное устройство или выделенный сервер. ВНИМАНИЕ! сервер должен быть проверенным и ему должно быть можно доверять! Не используйте неизвестные сервера для общения!

1. **Соберите серверный проект** (он находится в поддиректории `server/`). Процесс сборки аналогичен основному, целевой исполняемый файл — `fear-server`.
2. **Запустите сервер:**

```
# Перейдите в директорию с исполняемым файлом
cd fear/bin

# Запустите исполняемый файл
# На Linux/macOS
./fear server --port 7777

# На Windows
./fear.exe server --port 7777
```

Вы можете указать свой порт с помощью аргумента командной строки `--port` (например, `./fear --port 54321`).

Подключение клиента

1. Сначала нужно сгенерировать криптографический ключ.

```
# Перейдите в директорию с исполняемым файлом
cd fear/bin

# Запустите исполняемый файл
# На Linux/macOS
./fear genkey

# На Windows
./fear.exe genkey
```

Программа выдаст ключ. Например, z6aK3_k9I7rmpy6Sn-84QZ9Yc0p3T7VhzReWCKE0x4I

2. Запустите клиент, укажите настройки подключения к серверу и сгенерированный ключ.

```
# Перейдите в директорию с исполняемым файлом
cd fear/bin

# Запустите исполняемый файл
# На Linux/macOS
./fear client --host 127.0.0.1 --port 7777 --room testroom --key
z6aK3_k9I7rmpy6Sn-84QZ9Yc0p3T7VhzReWCKE0x4I --name User1

# На Windows
./fear.exe client --host 127.0.0.1 --port 7777 --room testroom --key
z6aK3_k9I7rmpy6Sn-84QZ9Yc0p3T7VhzReWCKE0x4I --name User1
```

Для общения в одной комнате всем участникам нужен один и тот же ключ.

3. Для безопасного получения ключа рекомендуется использовать программу key-exchange из пакета F.E.A.R. Project.

Структура проекта



```
F.E.A.R.-Project/
├── client-console/      # Исходный код клиент-серверной части
├── gui/                 # Исходный код приложения GUI
├── key-exchange/        # Исходный код приложения для обмена ключом
├── updater/            # Исходный код приложения менеджера обновлений
├── doc/                 # Дополнительная документация
├── CMakeLists.txt       # Главный файл конфигурации CMake
└── README.md           # Этот файл
```

Технологии

- **C++17** - Основной язык разработки
- **Qt 6** - Кроссплатформенный фреймворк для GUI
- **CMake** - Система сборки
- **Libsodium / OpenSSL** - Криптографические примитивы
- **Curl** - Сетевые взаимодействия

Дорожная карта

Текущая разработка

-  Разработка базовой архитектуры
-  Правки GUI интерфейса

- 🛠 Разработка update-сервиса
- 🐛 Исправление ошибок и доведение проекта до минимально рабочего варианта

📋 Запланировано

- Расширенный и удобный GUI интерфейс
- Передача файлов
- Аудио и видео звонки

📈 Долгосрочное видение

- 🌐 Web-версия
- 📱 Мобильное приложение для Android и iOS
- 📄 Возможность выбора протоколов шифрования

🤝 Как внести свой вклад

Мы приветствуем вклад в развитие F.E.A.R. Project!

1. Сделайте форк (Fork) репозитория.
2. Создайте ветку для вашей функции (`git checkout -b feature/AmazingFeature`).
3. Закоммитьте ваши изменения (`git commit -m 'Add some AmazingFeature'`).
4. Запушьте в ветку (`git push origin feature/AmazingFeature`).
5. Откройте Pull Request.

Пожалуйста, убедитесь, что ваш код соответствует стилю проекта и проходит все тесты.

📄 Лицензия

Этот проект распространяется под лицензией MIT. См. файл [LICENSE](#) для получения подробной информации.

Оставайтесь анонимными. Оставайтесь в безопасности.