Система генерации ключей и защищённой коммуникации на основе WiFi CSI

Описание

Данный проект реализует систему генерации ключей на физическом уровне (PLKG), которая использует информацию о состоянии канала WiFi (CSI) от устройств ESP32 для установления криптографических ключей между двумя сторонами (Алиса и Боб). Система демонстрирует устойчивость к пассивному прослушиванию со стороны злоумышленника (Ева) за счёт генерации ключей на основе CSI и реализует протокол Cascade для согласования ключей.[1]

Ключевые возможности

- **Генерация ключей на основе CSI**: Извлечение криптографических ключей из характеристик WiFi-канала
- **Согласование ключей по протоколу Cascade**: Интерактивный протокол коррекции ошибок для выравнивания ключей между легитимными сторонами
- **Шифрование AES-GCM**: Безопасный обмен сообщениями с использованием сгенерированных ключей
- **Симуляция пассивного прослушивателя**: Демонстрация защищённости от атак типа "человек посередине"
- **Визуализация в реальном времени**: Построение графиков амплитудных характеристик канала для всех участников
- **Многопоточная архитектура**: Одновременный сбор CSI и обработка сообщений

Аппаратные требования

- 3× платы разработки ESP32 с поддержкой CSI
- USB-подключения для последовательной связи
- WiFi-окружение 2.4 ГГц

Программные зависимости

```
numpy
matplotlib
pycryptodome  # Шифрование AES
pyserial      # Связь с ESP32
```

Установка зависимостей:

```
pip install numpy matplotlib pycryptodome pyserial
или
pip install -r requirements.txt
```

Архитектура системы

Участники

- **Алиса**: Инициатор, отправляет зашифрованные сообщения
- **Боб**: Получатель, легитимный партнёр по коммуникации
- **Ева**: Пассивный прослушиватель, пытающийся восстановить ключ

Параметры конфигурации

```
ALICE_PORT = "/dev/ttyUSB0"
BOB_PORT = "/dev/ttyUSB1"
EVE_PORT = "/dev/ttyUSB2"
BAUD_RATE = 460800
WIFI_CHANNEL = 6
WIFI_BANDWIDTH = 40  # МГц
PHASE_DURATION = 15  # секунд
```

Принцип работы

Фаза 1: Сбор CSI (30 секунд)

1. Боб и Ева слушают, пока Алиса передаёт ping-пакеты в течение 15 секунд
2. Алиса и Ева слушают, пока Боб передаёт ping-пакеты в течение 15 секунд
3. Все устройства собирают CSI-измерения из принятых пакетов[1]

Фаза 2: Генерация ключа

Система извлекает ключи из CSI-данных с использованием алгоритма квантования:

- Выбирает 16 основных поднесущих из валидных CSI-диапазонов
- Применяет голосование соседей с ±2 соседними поднесущими
- Использует квартильные пороги (25%, 50%, 75%) для 2-битного квантования на поднесущую
- Генерирует исходные ключи размером 32 байта (256 бит)[1]

Фаза 3: Согласование ключей по протоколу Cascade

Алиса и Боб используют протокол Cascade для исправления битовых ошибок в своих ключах:

- Множественные проходы с увеличивающимся размером блоков (8 × 2^проход)
- Обнаружение ошибок на основе чётности
- Интерактивный бинарный поиск для исправления ошибок
- Ева пассивно слушает, но не может активно участвовать[1]

Фаза 4: Защищённая коммуникация

- Финальные ключи хешируются с помощью SHA-256
- Сообщения шифруются с использованием AES-GCM
- Nonce и тег аутентификации передаются вместе с шифротекстом[1]

Использование

1. Прошивка ESP32

Загрузите в устройства ESP32 прошивку с поддержкой сбора CSI и кастомных протокольных команд:

- `radio_init -c <канал> -b <ширина_полосы> -m <mac> -s below --restart`
- `recv -t <время> -m <mac>`
- `ping -t <время> -r <частота сообщений> -m <mac>`
- `msg_listen`
- `msg_send -m <mac_получателя> <сообщение>`

2. Запуск Python-скрипта

```
python CSI_key_final.py
```

3. Выполнение фаз

Скрипт автоматически выполняет:

- Настройку и инициализацию радио
- Фазу сбора CSI
- Генерацию ключей
- Согласование по Cascade
- Интерактивный защищённый чат

4. Режим интерактивного чата

После установления ключей введите сообщения для отправки зашифрованных коммуникаций:

```
You: Привет, Боб!
Bob: Сообщение получено безопасно
```

Введите `exit` или `quit` для завершения.

Файлы данных

Система генерирует CSV-файлы для последующего анализа:

- `alice.csv`: CSI-измерения и события Алисы
- `bob.csv`: CSI-измерения и события Боба
- `eve.csv`: CSI-измерения и события Евы[1]

Формат CSV: `метка_времени, тип, данные`

Анализ безопасности

Успешное согласование ключей

Алиса и Боб получают идентичные ключи после согласования по Cascade, что подтверждается сравнением хешей ключей.[1]

Недостатки Евы

Визуализация демонстрирует, что CSI-измерения Евы значительно отличаются от измерений Алисы и Боба из-за:

- Различных физических характеристик канала
- Пространственной декорреляции беспроводных каналов
- Взаимности, применимой только к каналу Алиса-Боб[2]

Ева не может успешно восстановить ключ путём простого пассивного наблюдения.[1]

Детали протокола Cascade

Роль Алисы

- Отправляет информацию о чётности для перемешанных блоков битов
- Отвечает на запросы бинарного поиска Боба
- Ожидает сигналов завершения[1]

Роль Боба

- Сравнивает полученные чётности с локальным ключом
- Инициирует бинарный поиск при обнаружении несовпадений
- Инвертирует ошибочные биты
- Отправляет сигнал завершения после каждого прохода[1]

Ограничения Евы

- Может только слушать протокольные сообщения
- Не может активно запрашивать Алису
- Должна угадывать позиции битов по ограниченной информации
- Успешно исправляет мало битов или не исправляет вовсе[1]

Метрики производительности

- **Генерация исходного ключа**: ~32 байта (256 бит)
- **Проходы Cascade**: 6 (настраивается)
- **Типичные исправления**: 0-15 бит для Боба (в зависимости от качества канала)
- **Успешность Евы**: Близка к нулю для хорошо разнесённых позиций

Будущие улучшения

- Адаптивный выбор размера блоков на основе оценочной вероятности битовой ошибки
- Усиление конфиденциальности для удаления частичной информации Евы
- Поддержка диапазона 5 ГГц

- Оптимизация производительности для приложений реального времени

Ссылки

- Протокол Cascade: G. Brassard and L. Salvail, "Secret-Key Reconciliation by Public Discussion"
- Генерация ключей на основе CSI: Литература по безопасности физического уровня
- ESP32 CSI: Документация Espressif ESP-IDF

Лицензия

Авторы: **Последнее обновление**: Декабрь 2025