Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет
имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ

КАФЕДРА СИСТЕМЫ ОБРАБОТКИ ИНФОРМАЦИИ И УПРАВЛЕНИЯ

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К НАУЧНО-ИССЛЕДОВАТЕЛЬСКОЙ РАБОТЕ

НА ТЕМУ:

Технология реализации модели информационно-аналитической системы безопасности защиты речевой информации

Студент ИУ5-35М _____ **О.И. Козинов**
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель _____ **Ю.Е. Гапанюк**
(Подпись, дата) (И.О.Фамилия)

2023 г.

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

УТВЕРЖДАЮ
Заведующий кафедрой ___ИУ5____
(Индекс)
_____ ___В.И. Терехов__
(И.О.Фамилия)
«_04_» _____сентября_____ 2023 г.

З А Д А Н И Е
на выполнение научно-исследовательской работы

по теме _____Технология реализации модели информационно-аналитической системы безопасности защиты речевой информации_____
_____

Студент группы _ИУ5-35М__

_____Козинов Олег Игоревич_____
(Фамилия, имя, отчество)

Направленность НИР (учебная, исследовательская, практическая, производственная, др.)
_____ИССЛЕДОВАТЕЛЬСКАЯ_____
Источник тематики (кафедра, предприятие, НИР) _____КАФЕДРА_____

График выполнения НИР: 25% к __._ нед., 50% к _._ нед., 75% к _._ нед., 100% к __._ нед.

Техническое задание ____Разработать информационно-аналитическую систему безопасности для защиты речевой информации. Определить архитектуру, разработать программный комплекс, методику работы, проведение экспериментального исследования с анализом результатов_____
_____
_____
_____
_____

Оформление научно-исследовательской работы:

Расчетно-пояснительная записка на __41__ листах формата А4.
Перечень графического (иллюстративного) материала (чертежи, плакаты, слайды и т.п.)
_____
_____
_____
_____

Дата выдачи задания «_04_» ____сентября_____ 2023 г.

Руководитель НИР _____ _____Ю.Е. Гапанюк_____
(Подпись, дата) (И.О.Фамилия)

Студент _____ _____О.И. Козинов_____
(Подпись, дата) (И.О.Фамилия)

Примечание: Задание оформляется в двух экземплярах: один выдается студенту, второй хранится на кафедре.

Оглавление

ВВЕДЕНИЕ

В результате стремительного роста и цифровизации коммерческой сферы информационная безопасность стала решающим фактором успеха для любой организации. В связи с этим компании создают целый арсенал передового программного обеспечения и оборудования, выстраивают защиту и мониторинг инфраструктуры. Однако эффективность этих систем возможна только в том случае, если обеспечена защита каждого из модулей и видов информации во всей инфраструктуре организации в целом.

Для обеспечения информационной безопасности важное значение имеет анализ защищенности каналов и среды передачи речевой информации, содержащей чувствительные данные, в том числе для служебного пользования и с грифом секретности.

Задачи защиты речевой информации от утечки по акустическим каналам, порождаемым речевой деятельностью человека, занимают ведущее место в области безопасности информации. При этом ряд аспектов, влияющих на эффективность защиты речевой информации, зачастую остается за пределами внимания при организации системы информационной безопасности объектов, разработке и производстве средств защиты речевой информации, их практическом применении. Для того чтобы обезопасить себя от рисков потери информации необходимо предпринять ряд мероприятий организационного и технического характера, направленных на построение системы защиты информации.

Цель данной работы – создание условий для оптимизации процесса выявления угроз речевой информации посредством использования информационно-аналитической системы безопасности защиты речевой информации. Основной вопрос данного исследования заключается в том, что разработка модулей модели информационно-аналитической системы безопасности защиты речевой информации может ускорить процесс обнаружения, расследования и реагирования на инциденты нарушения физического периметра и нарушение законодательства.

Согласно цели и исследуемому вопросу, составлены следующие задачи исследования:

– Разработать архитектуру модели информационно-аналитической системы безопасности защиты речевой информации,

– Разработать методику работы с программной реализацией модели информационно-аналитической системы безопасности защиты речевой информации,

– Провести экспериментальное исследование модулей модели информационно-аналитической системы безопасности защиты речевой информации, агрегировать результаты, сделать выводы.

Технология реализации модели информационно-аналитической системы безопасности защиты речевой информации

Архитектура модели информационно-аналитической системы безопасности защиты речевой информации

Типовая информационно-аналитическая система безопасности защиты речевой информации имеет следующий вид:

1. Модуль channel ident,
2. Модуль channel security,
3. Модуль журналирования,
4. Модуль администрирования,
5. Модуль взаимодействия с журналом,
6. Модуль GUI.

Данные компоненты предназначены для получения первичной информации для анализа, такой как: результат анализа канала передачи, идентификация необходимых операций по безопасности. Это позволяет собирать следующую информацию:

– какие каналы связи обладают высоким уровнем доверия,

– насколько информация, передаваемая по каналу, защищена,

– вероятность воздействия на канал связи на физическом уровне.

Модуль channel ident предназначен для определения активных каналов связи, интенсивности потока данных, загрузки и подключения к каналу, получения первичной конфигурационной информации, такой как частота передачи, количество усилителей, протоколы.

Модуль channel security отвечает за оценку безопасности организации связи, спецификацию настроек канала передачи, оценку рисков и вероятности неправомерного доступа к информации на физическом уровне, а также для нормализации и предотвращения нелегитимного воздействия – приведения канала передачи к высокому уровню доверия и безопасности.

Для обновления данных о каналах связи и передаваемой информации, сбора информации о работе системы, состоянии ее модулей и контроля качества работы используется модуль администрирования.

Модуль журналирования несет функционал обновления, удаления и изменения базы данных с результатами оценки и нормализации каналов. Его можно создать на основе одной из открытых баз данных PostgreSQL.

Модуль взаимодействия с журналом предназначен для обмена актуальной информацией между модулем администрирования и модулем хранилища. Обновление происходит по заданным промежуткам от 30 секунд до 3 минут, в зависимости интенсивности взаимодействия.

Модуль GUI предназначен для работы в пользовательском интерфейсе системы.

На рисунке 1 представлена архитектура информационно-аналитической системы безопасности защиты речевой информации.

В ходе анализа архитектуры информационно-аналитической системы безопасности защиты речевой информации определены ее основные компоненты: модуль channel ident, модуль channel security, модуль администрирования, модуль журналирования, модуль взаимодействия с журналом и модуль графического интерфейса.

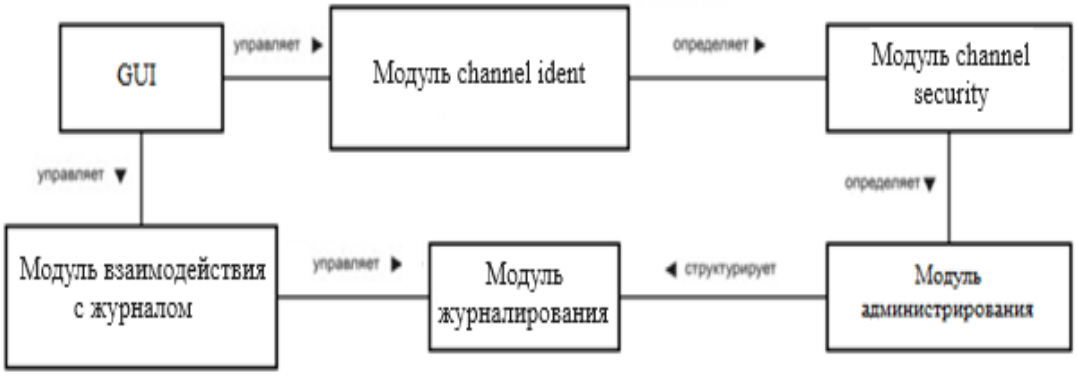

Рисунок 1 – Архитектура информационно-аналитической системы безопасности защиты речевой информации

Методика работы с программным комплексом реализации модели информационно-аналитической системы безопасности защиты речевой информации

Для начала работы с программным комплексом реализации модели информационно-аналитической системы безопасности защиты речевой информации, необходимо запустить комплекс и получить интерфейс администрирования.

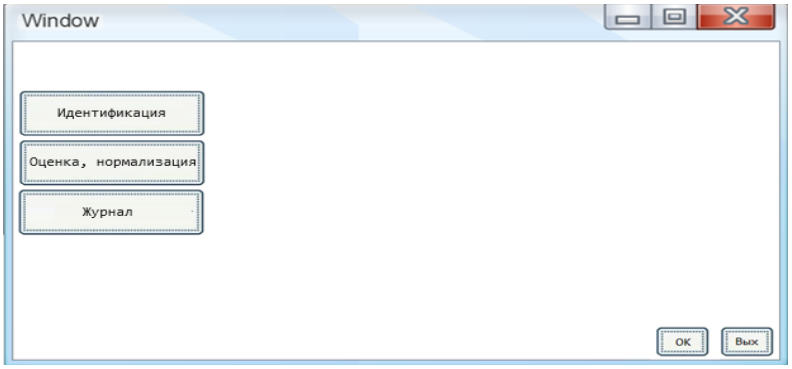

Рисунок 2 – Экранная копия интерфейса администрирования

Для запуска процесса идентификации и обнаружения активных каналов связи необходимо нажать кнопку «Идентификация», после чего запустится интерфейс и модуль channel ident, отвечающий за сканирование активных каналов связи, сбор параметров и определение объемов и характеристик информации, передаваемой по линии связи.

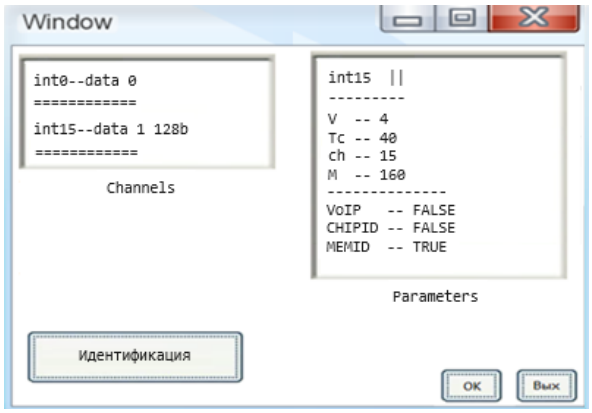

Рисунок 3 – Экранная копия интерфейса анализа

Для анализа безопасности полученного объема информации и соотношения параметров канала связи используется модуль channel security, доступный по кнопке «Оценка, нормализация». В открывшемся окне пользователь увидит два вложенных окна, одно из которых содержит перечень

базовых параметров канала связи или интерфейса, которые использовались при передаче информации. При нажатии кнопки «Оценка и нормализация» произойдет принудительная смена параметров на наиболее безопасные и появится параметр SECURE, равный 10, что означает активную защиту канала связи.

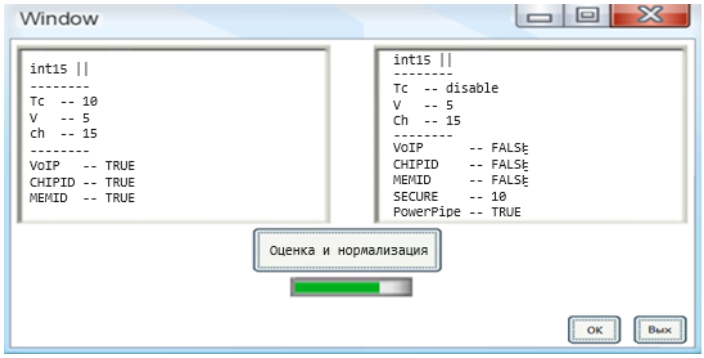

Рисунок 4 – Экранная копия интерфейса нормализации

Если пользователю необходимо ознакомиться с журналом, необходимо нажать «Журнал», после чего откроется окно интерфейса просмотра и редактирования журнала. В представленном окне пользователь имеет возможность ввести доступный активный канал связи и затем добавить в базу - внести его принудительно, нажав кнопку «добавить», или же убрать канал связи в связи с неактуальностью, нажав кнопку «исключить». Ознакомиться с базой и произвести редактирование полученного списка пользователь может нажатием кнопки «редактировать».

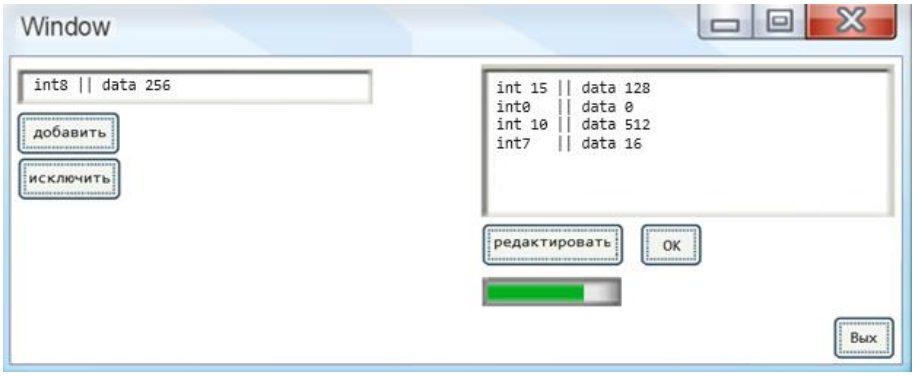

Рисунок 5 – Экранная копия интерфейса журналирования

Таким образом, была составлена методика работы с программным комплексом реализации модели информационно-аналитической системы безопасности защиты речевой информации. Разработанная модель позволяет

детально разобрать, какое назначение имеет тот или иной элемент интерфейса и какие действия необходимо выполнить для достижения результата.

Выводы по разделу 1

В ходе анализа архитектуры информационно-аналитической системы безопасности защиты речевой информации определены ее основные компоненты: модуль channel ident, модуль channel security, модуль администрирования, модуль журналирования, модуль взаимодействия с журналом и модуль графического интерфейса.

Таким образом, была составлена методика работы с программным комплексом реализации модели информационно-аналитической системы безопасности защиты речевой информации. Разработанная модель позволяет детально разобрать, какое назначение имеет тот или иной элемент интерфейса и какие действия необходимо выполнить для достижения результата.

Экспериментальное исследование модели информационно-аналитической системы безопасности защиты речевой информации

План проведения экспериментального исследования модели информационно-аналитической системы безопасности защиты речевой информации

В данном разделе представлен план проведения экспериментального исследования информационно-аналитической системы безопасности защиты речевой информации. В соответствии с обозначенными целями, были утверждены следующие этапы исследования:

1. Подготовка к тестированию:
    а) Определение целей и задач,
    б) Определение функциональных возможностей продукта и перечня модулей,
    в) Определение результатов тестирования и критериев оценки,
    г) Разработка примеров.
2. Функциональное тестирование:
    а) Проверка функциональности каждого модуля на соответствие,
    б) Тестирование работы с базой данных PostgreSQL.
3. Тестирование производительности:
    а) Проверка быстродействия системы и модулей.
4. Тестирование пользовательского интерфейса:
    а) Проверка корректности отображения и функционирования элементов интерфейса системы,
    б) Тестирование GUI.
5. Оформление отчета о тестировании:
    а) Описание результатов,
    б) Детальное описание ошибок,
    в) Оценка качества,
    г) Составление рекомендаций по исправлению.

Этапы эксперимента, исходные данные и прогнозируемые результаты экспериментального исследования описаны в таблице 1.

Таблица 1 – Этапы экспериментального исследования

Экспери-мент	**Задачи**	**Исходные данные**	**Прогноз результата**	**Условие**
1	Обнаружить небезопасные конфигурации канала передачи, нормализовать соединение	Параметры подключения, канал связи	Обнаружение небезопасных настроек, нормализация	Использование штатных средств системы передачи информации
2	Обнаружить небезопасные конфигурации канала передачи, нормализовать соединение посредством использования ИАС	Параметры подключения, канал связи	Обнаружение небезопасных настроек, нормализация	Использование строго функционала ИАС
3	Обнаружить небезопасные конфигурации канала передачи, нормализовать соединение	Параметры подключения, канал связи, НУПы	Обнаружение небезопасных настроек, нормализация	Использование штатных средств системы передачи информации
4	Обнаружить небезопасные конфигурации канала передачи, нормализовать соединение посредством использования ИАС	Параметры подключения, канал связи, НУПы	Обнаружение небезопасных настроек, нормализация	Использование строго функционала ИАС

Продолжение таблицы 1

Экспери-мент	**Задачи**	**Исходные данные**	**Прогноз результата**	**Условие**
5	Обнаружить небезопасные конфигурации канала передачи, нормализовать соединение	Параметры подключения, канал связи, НУПы	Обнаружение небезопасных настроек, нормализация	Использование штатных средств системы передачи информации старого образца
6	Обнаружить небезопасные конфигурации канала передачи, нормализовать соединение посредством использования ИАС	Параметры подключения, канал связи, НУПы	Обнаружение небезопасных настроек, нормализация	Использование строго функционала ИАС в системах старого образца

Разработан план проведения экспериментального исследования модели информационно-аналитической системы безопасности защиты речевой информации. Разработанный план позволяет провести исследования на основе экспериментального метода, структурировать и сравнить полученные результаты.

Реализация экспериментального исследования модели информационно-аналитической системы безопасности удаленного доступа

Цель исследования – оценка эффективности процесса выявления небезопасной конфигурации на основе анализа параметров подключения и интенсивности потока информации.

Задачи исследования:

1. Сформировать среду передачи;
2. Определить уровень безопасности каналов связи;
3. Собрать необходимые метаданные, статистику, сформировать отчет.

Методика исследования:

1. Подготовка оконечных устройств к установлению связи;
2. Исследование параметров конфигурации, обнаружение аномалий;
3. Агрегация статистики и полученных результатов, формирование отчета;
4. Оценка времени выполнения задачи.

Экспериментальное исследование №1.

1. Настройка и подготовка канала связи к работе.

Для старта эксперимента необходимо сконфигурировать среду передачи между двумя точками без использования НУП. Затраченное время – 15 минут.

2. Исследование параметров конфигурации каналов связи, поиск точек усиления сигналов.

В процессе анализа обнаружена недостаточно безопасная частотная модуляция в канале связи, что позволяет потенциальному злоумышленнику, встроив устройство в канал связи, получить передаваемую конфиденциальную информацию в слабо защищенном виде. Затраченное время – 25 минут.

3. Консолидация и составление отчета.

Для формирования отчета учитывается статистика работы канала связи и чистота среды передачи, а также скорость передачи информации по каналу, его параметры конфигурации и оценка защищенности. Затраченное время – 30 минут.

4. Суммарная оценка трудозатрат.

Исходя из объективной оценки, время, затраченное на проведение данного экспериментального исследования, составляет 70 минут (1 час, 10 минут).

Экспериментальное исследование №2.

1. Настройка и подготовка канала связи к работе.

Для старта эксперимента необходимо сконфигурировать среду передачи между двумя точками без использования НУП. Затраченное время – 15 минут.

2. Исследование параметров конфигурации каналов связи, поиск точек усиления сигналов.

В процессе анализа обнаружена недостаточно безопасная частотная модуляция в канале связи, что позволяет потенциальному злоумышленнику, встроив устройство в канал связи, получить передаваемую конфиденциальную информацию в слабо защищенном виде, а также обнаружена потеря несущего колебания на дистанции передачи информации, что чревато потерей чувствительных данных на уровне помех и шумов. Затраченное время – 8 минут.

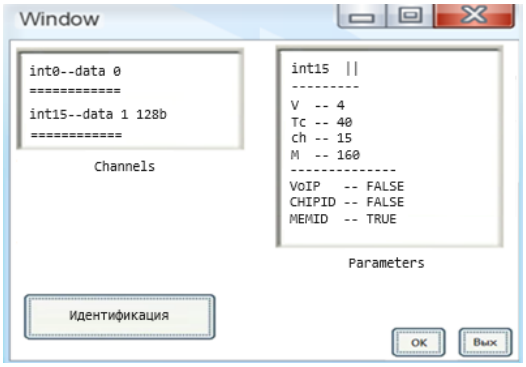

Рисунок 6 – Анализ канала передачи

3. Консолидация и составление отчета.

Для формирования отчета учитывается статистика работы канала связи и чистота среды передачи, а также скорость передачи информации по каналу, его параметры конфигурации и оценка защищенности. Затраченное время – 2 минуты.

```
==================================
      GEN REPORT FOR I15       ||
==================================
int            ||      15       ||
==================================
Tc/V/cH        ||   10,5,15     ||
==================================
MEMID          ||     TRUE      ||
==================================
MODULATION     ||     TRUE      ||
==================================
SECURITY       ||      70%      ||
==================================
==================================
```

Рисунок 7 – Отчетность

4. Суммарная оценка трудозатрат.

Исходя из объективной оценки, время, затраченное на проведение данного экспериментального исследования, составляет 25 минут.

Экспериментальное исследование №3.

Настройка и подготовка канала связи к работе.

Для старта эксперимента необходимо сконфигурировать среду передачи между двумя точками с использованием НУП. Затраченное время – 17 минут.

Исследование параметров конфигурации каналов связи, поиск точек усиления сигналов.

В процессе анализа обнаружена недостаточно безопасная частотная модуляция в канале связи на втором участке (НУП1), а также было обнаружено частотно-демодулирующее устройство, включенное в разрыв линии связи, которое позволяет потенциальному злоумышленнику получить передаваемую конфиденциальную информацию в слабо защищенном виде. Затраченное время – 45 минут.

Консолидация и составление отчета.

Для формирования отчета учитывается статистика работы канала связи и чистота среды передачи, а также скорость передачи информации по каналу, его параметры конфигурации и оценка защищенности. Затраченное время – 40 минут.

Суммарная оценка трудозатрат.

Исходя из объективной оценки, время, затраченное на проведение данного экспериментального исследования, составляет 102 минуты (1 час, 42 минуты).

Экспериментальное исследование №4.

1. Настройка и подготовка канала связи к работе.

Для старта эксперимента необходимо сконфигурировать среду передачи между двумя точками с использованием НУП. Затраченное время – 17 минут.

2. Исследование параметров конфигурации каналов связи, поиск точек усиления сигналов.

В процессе анализа обнаружена недостаточно безопасная частотная модуляция в канале связи на втором участке (НУП1), а также было обнаружено

частотно-демодулирующее устройство, включенное в разрыв линии связи, которое позволяет потенциальному злоумышленнику получить передаваемую конфиденциальную информацию в слабо защищенном виде. Затраченное время – 13 минут.

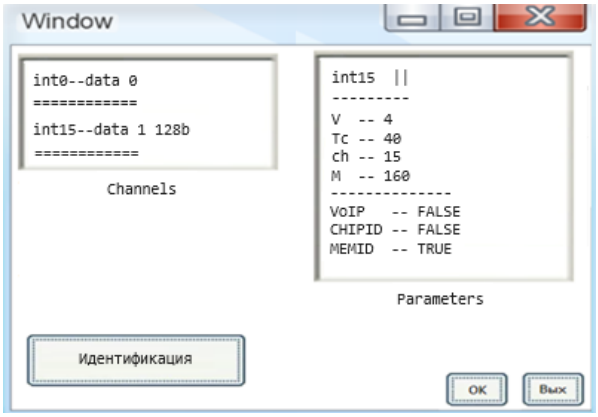

Рисунок 8 – Анализ канала передачи

3. Консолидация и составление отчета.

Для формирования отчета учитывается статистика работы канала связи и чистота среды передачи, а также скорость передачи информации по каналу, его параметры конфигурации и оценка защищенности. Затраченное время – 3 минуты.

```
================================
      GEN REPORT FOR I17       ||
================================
int           ||      17       ||
================================
Tc/V/cH       ||   10,5,15     ||
================================
MEMID         ||     TRUE      ||
================================
MODULATION    ||     TRUE      ||
================================
=============================================
SECURITY      ||      20%      || critical ||
=============================================
VULNERABILITY ||      YES      ||
================================
EXPLOATATION  ||  V-DEMODULATI ||
================================
NORMALISATION ||    need 70%   ||
================================
```

Рисунок 9 – Отчетность

4. Суммарная оценка трудозатрат.

Исходя из объективной оценки, время, затраченное на проведение данного экспериментального исследования, составляет 25 минут.

Экспериментальное исследование №5.

1. Настройка и подготовка канала связи к работе.

Для старта эксперимента необходимо сконфигурировать среду передачи между двумя точками с использованием НУП на базе устройств старого образца. Затраченное время – 24 минуты.

2. Исследование параметров конфигурации каналов связи, поиск точек усиления сигналов.

В процессе анализа обнаружена недостаточно безопасная частотная модуляция в канале связи на третьем участке (НУП2), что позволяет потенциальному злоумышленнику получить передаваемую конфиденциальную информацию в слабо защищенном виде, подключив демодулирующее устройство в разрыв. Затраченное время – 52 минуты.

3. Консолидация и составление отчета.

Для формирования отчета учитывается статистика работы канала связи и чистота среды передачи, а также скорость передачи информации по каналу, его параметры конфигурации и оценка защищенности. Затраченное время – 43 минуты.

4. Суммарная оценка трудозатрат.

Исходя из объективной оценки, время, затраченное на проведение данного экспериментального исследования, составляет 119 минут (1 час, 59 минут).

Экспериментальное исследование №6.

1. Настройка и подготовка канала связи к работе.

Для старта эксперимента необходимо сконфигурировать среду передачи между двумя точками с использованием НУП на базе устройств старого образца. Затраченное время – 24 минуты.

2. Исследование параметров конфигурации каналов связи, поиск точек усиления сигналов.

В процессе анализа обнаружена недостаточно безопасная частотная модуляция в канале связи на третьем участке (НУП2), что позволяет потенциальному злоумышленнику получить передаваемую конфиденциальную

информацию в слабо защищенном виде, подключив демодулирующее устройство в разрыв. Также системой было обнаружено несоответствие скорости передачи данных, уровня шумов в канале рекомендуемым значениям. Обнаружено использование устаревшего оборудования с истекшими сертификатами соответствия. Затраченное время – 22 минуты.

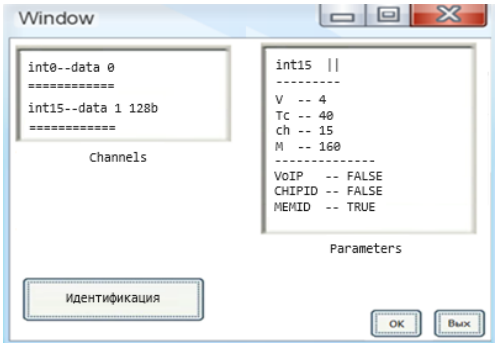

Рисунок 10 – Анализ канала передачи

3. Консолидация и составление отчета.

Для формирования отчета учитывается статистика работы канала связи и чистота среды передачи, а также скорость передачи информации по каналу, его параметры конфигурации и оценка защищенности. Затраченное время – 5 минут.

```
================================
      GEN REPORT FOR I17       ||
================================
int           ||      17       ||
================================
Tc/V/cH       ||   4,3,8     ||
================================
MEMID         ||     TRUE      ||
================================
MODULATION    ||     TRUE      ||
================================
SPEED         ||     LOW       ||
=============================================
SECURITY      ||      40%      ||  MEDIUM  ||
=============================================
VULNERABILITY ||      YES      ||
================================
NORMALISATION ||    need 70%   ||
================================
DETECTED OLD PROT & SWITCHES - ||
NEED UPGRADE!!!                ||
================================
```

Рисунок 11 – Отчетность

5. Суммарная оценка трудозатрат.

Исходя из объективной оценки, время, затраченное на проведение данного экспериментального исследования, составляет 51 минуту.

В результате экспериментального исследования было выявлено, что система обеспечивает оптимальный уровень работы. Рассматриваемая модель позволяет значительно сократить время анализа параметров соединения и обнаружения несоответствий требованиям безопасности, а наличие пользовательского интерфейса обеспечивает удобство при работе.

2.3 Анализ результатов экспериментального исследования модели информационно-аналитической системы безопасности защиты речевой информации

Для анализа результатов построена гистограмма, которая отражает затраченное время при проведении экспериментов.

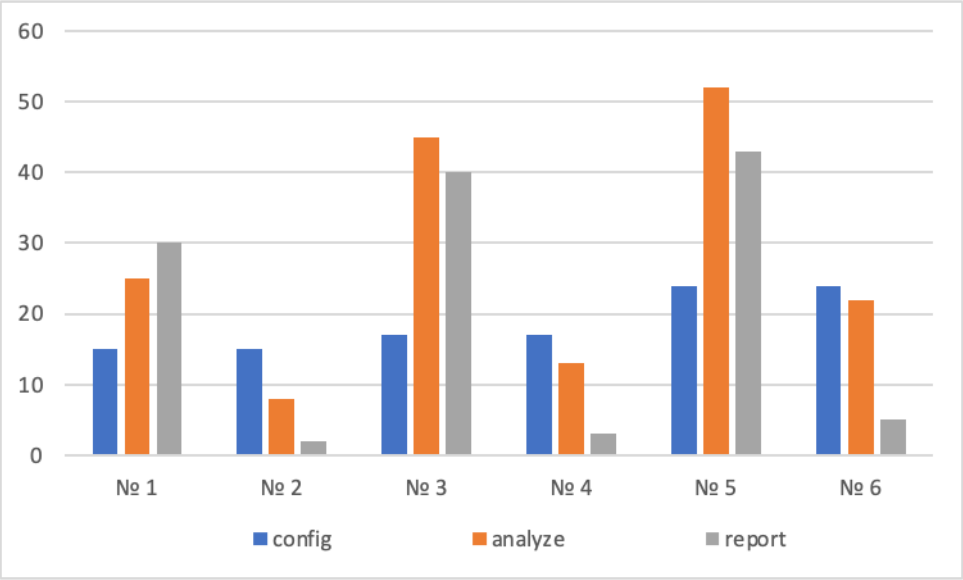

Рисунок 12 – Статистика затраченных ресурсов

Сформированная диаграмма подтверждает, что эксперименты без использования ИАС требуют существенно больших временных трудозатрат на каждом из этапов.

В таблице ниже приведены результаты экспериментов.

Таблица 2 – Результаты исследований

	Подготовка дампа, мин	Исследование и анализ, мин	Отчет, мин	Общее время, мин	Успешность
Эксперимент № 1	15	25	30	70	Да
Эксперимент № 2	15	8	2	25	Да

Эксперимент № 3	17	45	40	102	Да
Эксперимент № 4	17	13	3	33	Да
Эксперимент № 5	24	52	43	119	Да
Эксперимент № 6	24	22	5	51	Да

В результате экспериментального исследования, поставленные задачи были выполнены, цель была достигнута в соответствии с описанными этапами. Из результатов экспериментов следует, что оптимизация процессов обнаружения и поддержания соответствия требованиям безопасности процедур передачи голосовой информации реализована.

2.4 Вывод по разделу 2

1. Разработан план проведения экспериментального исследования модели информационно-аналитической системы безопасности защиты речевой информации. Разработанный план позволяет провести исследования на основе экспериментального метода, структурировать и сравнить полученные результаты.

2. В результате экспериментального исследования было выявлено, что система обеспечивает оптимальный уровень работы. Рассматриваемая модель позволяет значительно сократить время анализа параметров соединения и обнаружения несоответствий требованиям безопасности, а наличие пользовательского интерфейса обеспечивает удобство при работе.

3. В результате экспериментального исследования, поставленные задачи были выполнены, цель была достигнута в соответствии с описанными этапами. Из результатов экспериментов следует, что оптимизация процессов обнаружения и поддержания соответствия требованиям безопасности процедур передачи голосовой информации реализована.

ЗАКЛЮЧЕНИЕ

В ходе анализа архитектуры информационно-аналитической системы безопасности защиты речевой информации определены ее основные компоненты: модуль channel ident, модуль channel security, модуль администрирования, модуль журналирования, модуль взаимодействия с журналом и модуль графического интерфейса.

Таким образом, была составлена методика работы с программным комплексом реализации модели информационно-аналитической системы безопасности защиты речевой информации. Разработанная модель позволяет детально разобрать, какое назначение имеет тот или иной элемент интерфейса и какие действия необходимо выполнить для достижения результата.

Разработан план проведения экспериментального исследования модели информационно-аналитической системы безопасности защиты речевой информации. Разработанный план позволяет провести исследования на основе экспериментального метода, структурировать и сравнить полученные результаты.

В результате экспериментального исследования было выявлено, что система обеспечивает оптимальный уровень работы. Рассматриваемая модель позволяет значительно сократить время анализа параметров соединения и обнаружения несоответствий требованиям безопасности, а наличие пользовательского интерфейса обеспечивает удобство при работе.

В результате экспериментального исследования, поставленные задачи были выполнены, цель была достигнута в соответствии с описанными этапами. Из результатов экспериментов следует, что оптимизация процессов обнаружения и поддержания соответствия требованиям безопасности процедур передачи голосовой информации реализована.

СПИСОК ЛИТЕРАТУРЫ

1) Закон РФ от 21.07.1993 № 5485-1 «О государственной тайне». ред. от 05.12.2022.

2) Федеральный закон от 27.07.2006 г. № 149-ФЗ «Об информации, информационных технологиях и о защите информации».

3) Бузов Г.А., Калинин С.В., Кондратьев А.В. Защита от утечки информации по техническим каналам // учебное пособие для вузов. - 2005.

4) СТБ ГОСТ Р 50840-2000 «Передача речи по трактам связи. Методы оценки качества, разборчивости и узнаваемости».

5) Давыдов Г.В., Каван Д.М., Шамгин Ю.В. Оценка разборчивости речи в зашумленном помещении. 2012.

6) Покровский Н.Б. 1962. Расчет и измерение разборчивости речи. б.м. : Связьиздат, 1962.

7) Железняк В.К., Макаров Ю.К., Хорев А.А. Некоторые методические подходы к оценке эффективности защиты речевой информации // Специальная техника. – М.: 2000, № 4.

8) Рашевский Я.И., Каргашин В.Л. Обзор зарубежных методов определения разборчивости речи. – Специальная техника, № 3-6, 2002

9) Steeneken, Herman J.M. Standardisation of performance criteria and assessments methods for speech communication.

10) ГОСТ 16600-72 Передача речи по трактам радиотелефонной связи.

ПРИЛОЖЕНИЕ А. КОД ПРОГРАММЫ РЕАЛИЗАЦИИ МОДЕЛИ ИНФОРМАЦИОННО-АНАЛИТИЧЕСКОЙ СИСТЕМЫ БЕЗОПАСНОСТИ ЗАЩИТЫ РЕЧЕВОЙ ИНФОРМАЦИИ

```
import datetime
import os

from PyQt5.QtWidgets import QTableWidgetItem

from micdes import Ui_Dialog_microphone
from sti import stiFromAudio, readwav
from PyQt5 import QtWidgets, QtCore, QtGui
from maindes import Ui_MainWindow
from sounddes import Ui_Dialog
from warndes import Ui_Dialog_warning
from micdes import Ui_Dialog_microphone
from datetime import date, datetime
import qtable
import sys
import soundfile as sf
import sounddevice as sd

filename = 'speech/ALLO_SC_01.WAV'
data, fs = sf.read(filename, dtype='float32')
soundChek, chekFs =sf.read('speech/pink10.wav', dtype='float32')
is_checked=0
stiCount=0
filesave=False

class micwindow (QtWidgets.QDialog):
    def  __init__(self):
        super(micwindow, self).__init__()
        self.ui = Ui_Dialog_microphone()
        self.ui.setupUi(self)

class warnwindow (QtWidgets.QDialog):
    def  __init__(self):
        super(warnwindow, self).__init__()
        self.ui = Ui_Dialog_warning()
        self.ui.setupUi(self)

class soundwindow (QtWidgets.QDialog):
    def __init__(self):
        super(soundwindow, self).__init__()
        self.ui = Ui_Dialog()
        self.ui.setupUi(self)
        self.ui.buttonBox.accepted.connect(self.btnnClicked)
        data, fs = sf.read(filename, dtype='float32')

    def btnnClicked(self):
        global is_checked
        is_checked =1

class mywindow(QtWidgets.QMainWindow):
    def __init__(self):
        super(mywindow, self).__init__()
        self.ui = Ui_MainWindow()
        self.ui.setupUi(self)
        # подключение клик-сигнал к слоту btnClicked
        self.ui.pushButton.clicked.connect(self.btnClicked)
```

```
    self.ui.pushButton_2.clicked.connect(self.checkClick)
    self.ui.pushButton_3.clicked.connect(self.runSTI)
    self.ui.tableWidget.setHorizontalHeaderLabels(('значение STI','текстовая заметка'))
    self.ui.action.triggered.connect(self.open)
    self.ui.action_2.triggered.connect(self.save)
    self.ui.action_3.triggered.connect(self.export)

def btnClicked(self):

    print(is_checked)
    if is_checked==0:

        warning_window = warnwindow()
        warning_window.exec()
    else:
        global stiCount
        stiCount=int(self.ui.spinBox.text())
        for i in range(stiCount):
            mic_window = micwindow()
            mic_window.exec()
            #global sd
            #global sf
            myrecording = sd.playrec(data, samplerate=fs, channels=1)
            sd.wait()
            sf.write('speech/recording' + str(i) + '.wav', myrecording, fs)

def save(self):
    global filesave
    filesave=True

def open(self):
    global stiCount
    stiCount = int(self.ui.spinBox.text())

def export(self):
    f=open('report '+str(date.today())+'.txt','w')
    for i in range(stiCount):
        Sti=self.ui.tableWidget.item(i, 0).text()
        coment=self.ui.tableWidget.item(i, 1).text()
        f.writelines(Sti.center(10)+coment+'\n')

    f.close()

    pass
def checkClick(self):
    soundDialog=soundwindow()
    sd.play(soundChek, samplerate=chekFs, loop=True)
    soundDialog.exec()
    sd.stop()

def runSTI(self):
    #global stiCount
    stis=[]
    #stiCount = 2
    self.ui.tableWidget.setEnabled(True)
    refAudio, refRate =readwav(filename)
    #refAudio, refRate = sf.read(filename, dtype='float32')

    for i in range(stiCount):
        degrAudio, degrRate = readwav('speech/recording' +str(i)  + '.wav')
        stis.append(stiFromAudio(refAudio, degrAudio, refRate))
    self.ui.tableWidget.setRowCount(stiCount)
    for i in range(stiCount):
```

```
                if stis[i]<0.0465 :
                    cellinfo =QTableWidgetItem(str(round(stis[i], 3))+' надежная')
                elif stis[i]<0.093:
                    cellinfo = QTableWidgetItem(str(round(stis[i], 3)) + ' нормальная')
                elif stis[i] < 0.372:
                    cellinfo = QTableWidgetItem(str(round(stis[i], 3)) + ' неудовл')
                else:
                    cellinfo = QTableWidgetItem(str(round(stis[i], 3)) + ' отсутствует')
                cellinfo.setFlags(
                    QtCore.Qt.ItemIsSelectable | QtCore.Qt.ItemIsEnabled
                )
                self.ui.tableWidget.setItem(i,0,cellinfo)
                #self.ui.tableWidget.setItem(i, 0,QTableWidgetItem(i))

            def closeEvent(self, a0: QtGui.QCloseEvent):
                if not(filesave):
                    for i in range(stiCount):
                        os.remove('speech/recording'+str(i)+'.wav')


        app = QtWidgets.QApplication([])
        application = mywindow()
        application.show()

        sys.exit(app.exec())

        def testSTI():
            # read audio
            refAudio, refRate = readwav('speech/eval1.wav')
            degrAudio, degrRate = readwav('speech/eval1_echo100.wav')

            # calculate the STI. Visually verify console output.
            stis = stiFromAudio(refAudio, degrAudio, refRate, name='eval1.wav')

            print
            'Test Result:'

            # test result
            if abs(stis - 0.63) < 0.002:
                print
                "OK"
                return 0
            else:
                print
                "FAILED"
                return 1


#!/usr/bin/python

"""
Speech Transmission Index (STI) from speech waveforms (real speech)

Copyright (C) 2011 Jon Polom <jmpolom@wayne.edu>
Licensed under the GNU General Public License
"""

from datetime import date, datetime
from matplotlib.mlab import cohere,psd
from numpy import append,array,clip,log10,nonzero,ones,power,reshape
from numpy import searchsorted,shape,sqrt,sum,vstack,zeros
from numpy.ma import masked_array
from scipy.io import wavfile
```

```
from scipy.signal import butter,firwin,decimate,lfilter
from sys import stdout
from warnings import catch_warnings,simplefilter
from math import floor



def thirdOctaves(minFreq, maxFreq):
    """
    Calculates a list of frequencies spaced 1/3 octave apart in hertz
    between minFreq and maxFreq

    Input
    -----
    * minFreq : float or int

        Must be non-zero and non-negative

    * maxFreq : float or int

        Must be non-zero and non-negative

    Output
    ------
    * freqs : ndarray
    """

    if minFreq <= 0 or maxFreq <= 0:
        raise ValueError("minFreq and maxFreq must be non-zero and non-negative")
    else:
        maxFreq = float(maxFreq)

        f = float(minFreq)
        freqs = array([f])

        while f < maxFreq:
            f = f * 10**0.1
            freqs = append(freqs, f)

        return freqs

def fftWindowSize(freqRes, hz):
    """
    Calculate power of 2 window length for FFT to achieve specified frequency
    resolution. Useful for power spectra and coherence calculations.

    Input
    -----
    * freqRes : float

        Desired frequency resolution in hertz

    * hz : int

        Sample rate, in hertz, of signal undergoing FFT

    Output
    ------
    * window : int
    """

    freqRes = float(freqRes)        # make sure frequency res is a float
    pwr = 1                         # initial power of 2 to try
```

```
    res = hz / float(2**pwr) # calculate frequency resolution

    while res > freqRes:
        pwr += 1
        res = hz / float(2**pwr)

    return 2**pwr

def downsampleBands(audio, hz, downsampleFactor):
    """
    Downsample audio by integer factor

    Input
    -----
    * audio : array-like

        Array of original audio samples

    * hz : float or int

        Original audio sample rate in hertz

    * downsampleFactor : int

        Factor to downsample audio by, if desired

    Output
    ------
    * dsAudio : ndarray

        Downsampled audio array

    * hz : int

        Downsampled audio sample rate in hertz
    """

    # calculate downsampled audio rate in hertz
    downsampleFactor = int(downsampleFactor)        # factor must be integer
    hz = int(hz / downsampleFactor)

    for band in audio:
        ds = decimate(band, downsampleFactor, ftype='fir')

        try:
            dsAudio = append(dsAudio, ds)
        except:
            dsAudio = ds

    return dsAudio, hz

def octaveBandFilter(audio, hz,
                     octaveBands=[125, 250, 500, 1000, 2000, 4000, 8000],
                     butterOrd=6, hammingTime=16.6):

    """
    Octave band filter raw audio. The audio is filtered through butterworth
    filters of order 6 (by default), squared to obtain the envelope and finally
    low-pass filtered using a 'hammingTime' length Hamming filter at 25 Hz.

    Input
    -----
    * audio : array-like
```

```
    Array of raw audio samples

* hz : float or int

    Audio sample rate in hertz

* octaveBands : array-like

    list or array of octave band center frequencies

* butterOrd : int

    butterworth filter order

* hammingTime : float or int

    Hamming window length, in milliseconds relative to audio sample rate

Output
------
* octaveBandAudio : ndarray

    Octave band filtered audio

* hz : float or int

    Filtered audio sample rate
"""

print("Butterworth filter order:", butterOrd)
print("Hamming filter length:   ", hammingTime, "milliseconds")
print("Audio sample rate:       ", hz)

# calculate the nyquist frequency
nyquist = hz * 0.5

# length of Hamming window for FIR low-pass at 25 Hz
hammingLength = (hammingTime / 1000.0) * hz

# process each octave band
for f in octaveBands:
    bands = str(octaveBands[:octaveBands.index(f) + 1]).strip('[]')
    statusStr = "Octave band filtering audio at: " + bands
    unitStr = "Hz ".rjust(80 - len(statusStr))
    stdout.write(statusStr)
    stdout.write(unitStr)
    stdout.write('\r')
    stdout.flush()

    # filter the output at the octave band f
    f1 = f / sqrt(2)
    f2 = f * sqrt(2)

    # for some odd reason the band-pass butterworth doesn't work right
    # when the filter order is high (above 3). likely a SciPy issue?
    # also, butter likes to complain about possibly useless results when
    # calculating filter coefficients for high order (above 4) low-pass
    # filters with relatively low knee frequencies (relative to nyquist F).
    # perhaps I just don't know how digital butterworth filters work and
    # their limitations but I think this is odd.
    # the issue described here will be sent to their mailing list
    if f < max(octaveBands):
```

```
            with catch_warnings():      # suppress the spurious warnings given
               simplefilter('ignore')  # under certain conditions
               b1,a1 = butter(butterOrd, f1/nyquist, btype='high')
               b2,a2 = butter(butterOrd, f2/nyquist, btype='low')

            filtOut = lfilter(b1, a1, audio)   # high-pass raw audio at f1
            filtOut = lfilter(b2, a2, filtOut) # low-pass after high-pass at f1
         else:
            with catch_warnings():
               simplefilter('ignore')
               b1,a1 = butter(butterOrd, f/nyquist, btype='high')
            filtOut = lfilter(b1, a1, audio)

         filtOut = array(filtOut)**2
         b = firwin(floor(hammingLength), 25.0, nyq=nyquist)
         filtOut = lfilter(b, 1, filtOut)
         filtOut = filtOut * -1.0

         # stack-up octave band filtered audio
         try:
            octaveBandAudio = vstack((octaveBandAudio, filtOut))
         except:
            octaveBandAudio = filtOut

      print
      return octaveBandAudio

   def octaveBandSpectra(filteredAudioBands, hz, fftRes=0.06):
      """
      Calculate octave band power spectras

      Input
      -----
      * filteredAudioBands : array-like

         Octave band filtered audio

      * hz : float or int

         Audio sample rate in hertz. Must be the same for clean and dirty audio

      * fftRes : float or int

         Desired FFT frequency resolution

      Output
      ------
      * spectras : ndarray

         Power spectra values

      * fftfreqs : ndarray

         Frequencies for FFT points
      """

      # FFT window size for PSD calculation: 32768 for ~0.06 Hz res at 2 kHz
      psdWindow = fftWindowSize(fftRes, hz)

      print("Calculating octave band power spectras", end=' ')
      print("(FFT length:", psdWindow, "samples)")

      for band in filteredAudioBands:
```

```
        spectra, freqs = psd(band, NFFT=psdWindow, Fs=hz)
        spectra = reshape(spectra, len(freqs))  # change to row vector
        spectra = spectra / max(spectra)        # scale to [0,1]

        # stack-up octave band spectras
        try:
            spectras = vstack((spectras, spectra))
            fftfreqs = vstack((fftfreqs, freqs))
        except:
            spectras = spectra
            fftfreqs = freqs

    return spectras, fftfreqs

def octaveBandCoherence(degrAudioBands, refAudioBands,
                        hz, fftRes=0.122):
    """
    Calculate coherence between clean and degraded octave band audio

    Input
    -----
    * degrAudioBands : array-like

        Degraded octave band audio

    * refAudioBands : array-like

        Reference (clean) octave band audio

    * hz : float or int

        Audio sample rate. Must be common between clean and dirty audio

    * fftRes : float or int

        Desired FFT frequency resolution

    Output
    ------
    * coherences : ndarray

        Coherence values

    * fftfreqs : ndarray

        Frequencies for FFT points
    """

    # FFT window size for PSD calculation: 32768 for ~0.06 Hz res at 2 kHz
    # Beware that 'cohere' isn't as forgiving as 'psd' with FFT lengths
    # larger than half the length of the signal
    psdWindow = fftWindowSize(fftRes, hz)

    print("Calculating degraded and reference audio coherence", end=' ')
    print("(FFT length:", psdWindow, "samples)")

    for i,band in enumerate(degrAudioBands):
        with catch_warnings():      # catch and ignore spurious warnings
            simplefilter('ignore')  # due to some irrelevant divide by 0's
            coherence, freqs = cohere(band, refAudioBands[i],
                                      NFFT=psdWindow, Fs=hz)

        # stack-up octave band spectras
```

```
        try:
            coherences = vstack((coherences, coherence))
            fftfreqs = vstack((fftfreqs, freqs))
        except:
            coherences = coherence
            fftfreqs = freqs

    return coherences, fftfreqs

def thirdOctaveRootSum(spectras, fftfreqs, minFreq=0.63, maxFreq=12.5):
    """
    Calculates square root of sum of spectra over 1/3 octave bands

    Input
    -----
    * spectras : array-like

        Array or list of octave band spectras

    * fftfreqs : array-like

        Array or list of octave band FFT frequencies

    * minFreq : float

        Min frequency in 1/3 octave bands

    * maxFreq : float

        Max frequency in 1/3 octave bands

    Output
    ------
    * thirdOctaveRootSums : ndarray

        Square root of spectra sums over 1/3 octave intervals
    """

    print("Calculating 1/3 octave square-rooted sums from", end=' ')
    print(minFreq, "to", maxFreq, "Hz")

    thirdOctaveBands = thirdOctaves(minFreq, maxFreq)

    # loop over the spectras contained in 'spectras' and calculate 1/3 oct MTF
    for i,spectra in enumerate(spectras):
        freqs = fftfreqs[i]             # get fft frequencies for spectra

        # calculate the third octave sums
        for f13 in thirdOctaveBands:
            f131 = f13 / power(2, 1.0/6.0) # band start
            f132 = f13 * power(2, 1.0/6.0) # band end

            li = searchsorted(freqs, f131)
            ui = searchsorted(freqs, f132) + 1

            s = sum(spectra[li:ui]) # sum the spectral components in band
            s = sqrt(s)             # take square root of summed components

            try:
                sums = append(sums, s)
            except:
                sums = array([s])
```

```
        # stack-up third octave modulation transfer functions
        try:
            thirdOctaveSums = vstack((thirdOctaveSums, sums))
        except:
            thirdOctaveSums = sums

        # remove temp 'sum' and 'counts' variables for next octave band
        del(sums)

    return thirdOctaveSums

def thirdOctaveRMS(spectras, fftfreqs, minFreq=0.63, maxFreq=12.50):
    """
    Calculates RMS value of spectra over 1/3 octave bands

    Input
    -----
    * spectras : array-like

        Array or list of octave band spectras

    * fftfreqs : array-like

        Array or list of octave band FFT frequencies

    * minFreq : float

        Min frequency in 1/3 octave bands

    * maxFreq : float

        Max frequency in 1/3 octave bands

    Output
    ------
    * thirdOctaveRMSValues : ndarray

        RMS value of spectra over 1/3 octave intervals
    """

    print("Calculating 1/3 octave RMS values from", end=' ')
    print(minFreq, "to", maxFreq, "Hz")

    thirdOctaveBands = thirdOctaves(minFreq, maxFreq)

    # loop over the spectras contained in 'spectras' and calculate 1/3 oct MTF
    for i,spectra in enumerate(spectras):
        freqs = fftfreqs[i]             # get fft frequencies for spectra

        # calculate the third octave sums
        for f13 in thirdOctaveBands:
            f131 = f13 / power(2, 1.0/6.0) # band start
            f132 = f13 * power(2, 1.0/6.0) # band end

            li = searchsorted(freqs, f131)
            ui = searchsorted(freqs, f132) + 1

            s = sum(spectra[li:ui]**2)  # sum the spectral components in band
            s = s / len(spectra[li:ui]) # divide by length of sum
            s = sqrt(s)             # square root

            try:
                sums = append(sums, s)
```

```
            except:
                sums = array([s])

        # stack-up third octave modulation transfer functions
        try:
            thirdOctaveRMSValues = vstack((thirdOctaveRMSValues, sums))
        except:
            thirdOctaveRMSValues = sums

        # remove temp 'sum' and 'counts' variables for next octave band
        del(sums)

    return thirdOctaveRMSValues

def sti(modulations, coherences, minCoherence=0.8):
    """
    Calculate the speech transmission index from third octave modulation
    indices. The indices are truncated after coherence between clean and dirty
    audio falls below 'minCoherence' or 0.8, by default.

    Input
    -----
    * modulations : array-like

        Modulation indices spaced at 1/3 octaves within each octave band

    * coherences : array-like

        Coherence between clean and dirty octave band filtered audio

    * minCoherence : float

        The minimum coherence to include a mod index in the STI computation

    Output
    ------
    * index : float

        The speech transmission index (STI)
    """

    # create masking array of zeroes
    snrMask = zeros(modulations.shape, dtype=int)

    # sort through coherence array and mask corresponding SNRs where coherence
    # values fall below 'minCoherence' (0.8 in most cases and by default)
    for i,band in enumerate(coherences):
        lessThanMin = nonzero(band < minCoherence)[0]
        if len(lessThanMin) >= 1:
            discardAfter = min(lessThanMin)
            snrMask[i][discardAfter:] = ones((len(snrMask[i][discardAfter:])))

    modulations = clip(modulations, 0, 0.99)      # clip to [0, 0.99] (max: ~1)
    snr = 10*log10(modulations/(1 - modulations)) # estimate SNR
    snr = clip(snr, -15, 15)                      # clip to [-15,15]
    snr = masked_array(snr, mask=snrMask)         # exclude values from sum
    snrCounts = (snr / snr).sum(axis=1)           # count SNRs
    snrCounts = snrCounts.data                    # remove masking
    octaveBandSNR = snr.sum(axis=1) / snrCounts   # calc average SNR
    alpha = 7 * (snrCounts / snrCounts.sum())     # calc alpha weight

    # octave band weighting factors, Steeneken and Houtgast (1985)
    w = [0.129, 0.143, 0.114, 0.114, 0.186, 0.171, 0.143]
```

```
    # calculate the STI measure
    snrp = alpha * w * octaveBandSNR
    snrp = snrp.sum()
    index = (snrp + 15) / 30.0

    print("Speech Transmission Index (STI):", index)
    return index

def stiFromAudio(reference, degraded, hz, calcref=False, downsample=None,
                 name="unnamed"):
    """
    Calculate the speech transmission index (STI) from clean and dirty
    (ie: distorted) audio samples. The clean and dirty audio samples must have
    a common sample rate for successful use of this function.

    Input
    -----
    * reference : array-like

        Clean reference audio sample as an array of floating-point values

    * degraded : array-like

        Degraded audio sample as an array, or array of arrays for multiple
        samples, of floating-point values

    * hz : int

        Audio sample rate in hertz

    * calcref : boolean

        Calculate STI for reference signal alone

    * downsample : int or None

        Downsampling integer factor

    * name : string

        Name of sample set, for output tracking in larger runs

    Output
    ------
    * sti : array-like or float

        The calculated speech transmission index (STI) value(s)
    """

    # put single sample degraded array into another array so the loop works
    if type(degraded) is not type([]):
        degraded = [degraded]

    print("-" * 80)
    print("Speech Transmission Index (STI) from speech waveforms".center(80))
    print("-" * 80)
    print
    print("Sample set:             ", name)
    print("Number of samples:      ",len(degraded))
    print( "Date/time:              ",datetime.now().isoformat())
    print ("Calculate reference STI:",)
    if calcref:
```

```
    print("yes")
else:
    print("no")
print
print (" Reference Speech ".center(80,'*'))

refOctaveBands = octaveBandFilter(reference, hz)
refRate = hz

# downsampling, if desired
if type(downsample) is type(1):
    refOctaveBands, refRate = downsampleBands(refOctaveBands, refRate,
                                      downsample)

# calculate STI for reference sample, if boolean set
if calcref:
    # STI calc procedure
    spectras, sfreqs = octaveBandSpectra(refOctaveBands, refRate)
    coherences, cfreqs = octaveBandCoherence(refOctaveBands, refOctaveBands,
                                     refRate)
    thirdOctaveMTF = thirdOctaveRootSum(spectras, sfreqs)
    thirdOctaveCoherences = thirdOctaveRMS(coherences, cfreqs)

    # add to interim array for MTFs and coherences
    try:
        thirdOctaveTemps.append([thirdOctaveMTF, thirdOctaveCoherences])
    except:
        thirdOctaveTemps = [[thirdOctaveMTF, thirdOctaveCoherences]]

print

# loop over degraded audio samples and calculate STIs
for j,sample in enumerate(degraded):
    print( " Degraded Speech: Sample {0} ".format(j + 1).center(80,'*'))
    degrOctaveBands = octaveBandFilter(sample, hz)
    degrRate = hz

    # downsampling, if desired
    if type(downsample) is type(1):
        degrOctaveBands, degrRate = downsampleBands(degrOctaveBands,
                                        degrRate, downsample)

    # STI calc procedure
    spectras, sfreqs = octaveBandSpectra(degrOctaveBands, degrRate)
    coherences, cfreqs = octaveBandCoherence(refOctaveBands,degrOctaveBands, refRate)
    thirdOctaveMTF = thirdOctaveRootSum(spectras, sfreqs)
    thirdOctaveCoherences = thirdOctaveRMS(coherences, cfreqs)

    # add to interim array for MTFs and coherences
    try:
        thirdOctaveTemps.append([thirdOctaveMTF, thirdOctaveCoherences])
    except:
        thirdOctaveTemps = [[thirdOctaveMTF, thirdOctaveCoherences]]

    print

# calculate the STI values
print( " Speech Transmission Index ".center(80,'*'))
for i in range(0,len(thirdOctaveTemps)):
    sampleSTI = sti(thirdOctaveTemps[i][0], thirdOctaveTemps[i][1])

    # add to STI output array
    try:
```

```
            stiValues.append(sampleSTI)
        except:
            stiValues = [sampleSTI]

    # unpack single value
    if len(stiValues) == 1:
        stiValues = stiValues[0]

    print
    return stiValues

def readwav(path):
    """
    Reads Microsoft WAV format audio files, scales integer sample values and
    to [0,1]. Returns a tuple consisting of scaled WAV samples and sample rate
    in hertz.

    Input
    -----
    * path : string

        Valid system path to file

    Output
    ------
    * audio : array-like

        Array of scaled sampled

    * rate : int

        Audio sample rate in hertz
    """
    wav = wavfile.read(path)

    rate = wav[0]
    audio = array(wav[1])

    scale = float(max(audio))
    audio = audio / scale

    return audio, rate
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'main.ui'
#
# Created by: PyQt5 UI code generator 5.15.7
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName("MainWindow")
        MainWindow.resize(467, 271)
        self.centralwidget = QtWidgets.QWidget(MainWindow)
        self.centralwidget.setObjectName("centralwidget")
        self.horizontalLayout_2 = QtWidgets.QHBoxLayout(self.centralwidget)
        self.horizontalLayout_2.setObjectName("horizontalLayout_2")
```

```
self.verticalLayout_3 = QtWidgets.QVBoxLayout()
self.verticalLayout_3.setContentsMargins(-1, -1, -1, 0)
self.verticalLayout_3.setObjectName("verticalLayout_3")
self.pushButton_2 = QtWidgets.QPushButton(self.centralwidget)
self.pushButton_2.setObjectName("pushButton_2")
self.verticalLayout_3.addWidget(self.pushButton_2)
self.verticalLayout_2 = QtWidgets.QVBoxLayout()
self.verticalLayout_2.setObjectName("verticalLayout_2")
self.verticalLayout = QtWidgets.QVBoxLayout()
self.verticalLayout.setObjectName("verticalLayout")
self.label = QtWidgets.QLabel(self.centralwidget)
self.label.setObjectName("label")
self.verticalLayout.addWidget(self.label)
self.horizontalLayout = QtWidgets.QHBoxLayout()
self.horizontalLayout.setObjectName("horizontalLayout")
self.spinBox = QtWidgets.QSpinBox(self.centralwidget)
self.spinBox.setMinimum(1)
self.spinBox.setObjectName("spinBox")
self.horizontalLayout.addWidget(self.spinBox)
self.pushButton = QtWidgets.QPushButton(self.centralwidget)
self.pushButton.setObjectName("pushButton")
self.horizontalLayout.addWidget(self.pushButton)
self.verticalLayout.addLayout(self.horizontalLayout)
self.verticalLayout_2.addLayout(self.verticalLayout)
self.pushButton_3 = QtWidgets.QPushButton(self.centralwidget)
self.pushButton_3.setObjectName("pushButton_3")
self.verticalLayout_2.addWidget(self.pushButton_3)
self.verticalLayout_3.addLayout(self.verticalLayout_2)
self.horizontalLayout_2.addLayout(self.verticalLayout_3)
self.tableWidget = QtWidgets.QTableWidget(self.centralwidget)
self.tableWidget.setEnabled(False)
self.tableWidget.setColumnCount(2)
self.tableWidget.setObjectName("tableWidget")
self.tableWidget.setRowCount(0)
self.tableWidget.horizontalHeader().setDefaultSectionSize(130)
self.horizontalLayout_2.addWidget(self.tableWidget)
MainWindow.setCentralWidget(self.centralwidget)
self.menubar = QtWidgets.QMenuBar(MainWindow)
self.menubar.setGeometry(QtCore.QRect(0, 0, 467, 21))
self.menubar.setObjectName("menubar")
self.menu = QtWidgets.QMenu(self.menubar)
self.menu.setObjectName("menu")
self.menu_2 = QtWidgets.QMenu(self.menubar)
self.menu_2.setObjectName("menu_2")
MainWindow.setMenuBar(self.menubar)
self.action = QtWidgets.QAction(MainWindow)
self.action.setObjectName("action")
self.action_2 = QtWidgets.QAction(MainWindow)
self.action_2.setObjectName("action_2")
self.action_3 = QtWidgets.QAction(MainWindow)
self.action_3.setObjectName("action_3")
self.action_4 = QtWidgets.QAction(MainWindow)
self.action_4.setObjectName("action_4")
self.menu.addAction(self.action)
self.menu.addAction(self.action_2)
self.menu.addAction(self.action_3)
self.menu_2.addAction(self.action_4)
self.menubar.addAction(self.menu.menuAction())
self.menubar.addAction(self.menu_2.menuAction())

self.retranslateUi(MainWindow)
QtCore.QMetaObject.connectSlotsByName(MainWindow)
```

```
    def retranslateUi(self, MainWindow):
        _translate = QtCore.QCoreApplication.translate
        MainWindow.setWindowTitle(_translate("MainWindow", "Оценка защищенности"))
        self.pushButton_2.setText(_translate("MainWindow", "тест громкости"))
        self.label.setText(_translate("MainWindow", "Введите количество точек замера"))
        self.pushButton.setText(_translate("MainWindow", "Замер"))
        self.pushButton_3.setText(_translate("MainWindow", "рассчет"))
        self.menu.setTitle(_translate("MainWindow", "файл"))
        self.menu_2.setTitle(_translate("MainWindow", "Справка"))
        self.action.setText(_translate("MainWindow", "Открыть"))
        self.action_2.setText(_translate("MainWindow", "Сохранить"))
        self.action_3.setText(_translate("MainWindow", "Экспорт отчет"))
        self.action_4.setText(_translate("MainWindow", "О программе"))
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'sc.ui'
#
# Created by: PyQt5 UI code generator 5.15.7
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_Dialog(object):
    def setupUi(self, Dialog):
        Dialog.setObjectName("Dialog")
        Dialog.resize(364, 152)
        self.buttonBox = QtWidgets.QDialogButtonBox(Dialog)
        self.buttonBox.setGeometry(QtCore.QRect(10, 100, 341, 32))
        self.buttonBox.setOrientation(QtCore.Qt.Horizontal)
        self.buttonBox.setStandardButtons(QtWidgets.QDialogButtonBox.Cancel|QtWidgets.QDialogButtonBox.Ok)
        self.buttonBox.setObjectName("buttonBox")
        self.label = QtWidgets.QLabel(Dialog)
        self.label.setGeometry(QtCore.QRect(10, 10, 351, 81))
        self.label.setObjectName("label")

        self.retranslateUi(Dialog)
        self.buttonBox.accepted.connect(Dialog.accept) # type: ignore
        self.buttonBox.rejected.connect(Dialog.reject) # type: ignore
        QtCore.QMetaObject.connectSlotsByName(Dialog)

    def retranslateUi(self, Dialog):
        _translate = QtCore.QCoreApplication.translate
        Dialog.setWindowTitle(_translate("Dialog", "Sound check"))
        self.label.setText(_translate("Dialog", "<html><head/><body><p><span style=\" font-size:11pt;\">Установите громкость тестового </span></p><p><span style=\" font-size:11pt;\">сигнала чтобы звуковое давление</span></p><p><span style=\" font-size:11pt;\">на расстоянии в 1м равнялось 65дБ</span></p></body></html>"))
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'warning.ui'
#
# Created by: PyQt5 UI code generator 5.15.7
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets
```

```
class Ui_Dialog_warning(object):
    def setupUi(self, Dialog_warning):
        Dialog_warning.setObjectName("Dialog_warning")
        Dialog_warning.resize(174, 140)
        Dialog_warning.setModal(True)
        self.verticalLayout = QtWidgets.QVBoxLayout(Dialog_warning)
        self.verticalLayout.setObjectName("verticalLayout")
        self.label = QtWidgets.QLabel(Dialog_warning)
        self.label.setMinimumSize(QtCore.QSize(156, 93))
        self.label.setMaximumSize(QtCore.QSize(156, 93))
        self.label.setObjectName("label")
        self.verticalLayout.addWidget(self.label)
        self.pushButton = QtWidgets.QPushButton(Dialog_warning)
        self.pushButton.setObjectName("pushButton")
        self.verticalLayout.addWidget(self.pushButton)

        self.retranslateUi(Dialog_warning)
        self.pushButton.clicked.connect(Dialog_warning.accept) # type: ignore
        QtCore.QMetaObject.connectSlotsByName(Dialog_warning)

    def retranslateUi(self, Dialog_warning):
        _translate = QtCore.QCoreApplication.translate
        Dialog_warning.setWindowTitle(_translate("Dialog_warning", "Warning"))
        self.label.setText(_translate("Dialog_warning", "<html><head/><body><p><span style=\" font-size:14pt;\">Внимание!</span></p><p><span style=\" font-size:14pt;\">громкость замера</span></p><p><span style=\" font-size:14pt;\">не установлена</span></p></body></html>"))
        self.pushButton.setText(_translate("Dialog_warning", "OK"))
# -*- coding: utf-8 -*-

# Form implementation generated from reading ui file 'microphone.ui'
#
# Created by: PyQt5 UI code generator 5.15.7
#
# WARNING: Any manual changes made to this file will be lost when pyuic5 is
# run again.  Do not edit this file unless you know what you are doing.


from PyQt5 import QtCore, QtGui, QtWidgets


class Ui_Dialog_microphone(object):
    def setupUi(self, Dialog_microphone):
        Dialog_microphone.setObjectName("Dialog_microphone")
        Dialog_microphone.resize(177, 105)
        Dialog_microphone.setMinimumSize(QtCore.QSize(177, 105))
        Dialog_microphone.setMaximumSize(QtCore.QSize(177, 105))
        Dialog_microphone.setModal(True)
        self.verticalLayout = QtWidgets.QVBoxLayout(Dialog_microphone)
        self.verticalLayout.setObjectName("verticalLayout")
        self.label = QtWidgets.QLabel(Dialog_microphone)
        self.label.setObjectName("label")
        self.verticalLayout.addWidget(self.label)
        self.pushButton = QtWidgets.QPushButton(Dialog_microphone)
        self.pushButton.setObjectName("pushButton")
        self.verticalLayout.addWidget(self.pushButton)

        self.retranslateUi(Dialog_microphone)
        self.pushButton.clicked.connect(Dialog_microphone.accept) # type: ignore
        QtCore.QMetaObject.connectSlotsByName(Dialog_microphone)

    def retranslateUi(self, Dialog_microphone):
        _translate = QtCore.QCoreApplication.translate
```

```
        Dialog_microphone.setWindowTitle(_translate("Dialog_microphone", "next point"))
        self.label.setText(_translate("Dialog_microphone", "<html><head/><body><p><span style=\" font-size:11pt;\">Установите микрофон </span></p><p><span style=\" font-size:11pt;\">в новое место замера</span></p></body></html>"))
        self.pushButton.setText(_translate("Dialog_microphone", "продолжить"))
```