ПРАВИТЕЛЬСТВО РОССИЙСКОЙ ФЕДЕРАЦИИ
НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ
«ВЫСШАЯ ШКОЛА ЭКОНОМИКИ»

Факультет компьютерных наук
Департамент программной инженерии

СОГЛАСОВАНО
Преподаватель факультета компьютерных наук, доцент базовой кафедры «Системное программирование» ИСП ран, канд. техн. наук

_____ А. И. Гетьман
«___» _____2022 г.

УТВЕРЖДАЮ
Академический руководитель образовательной программы «Программная инженерия», канд. техн .наук

_____ В. В. Шилов
«___» _____ 2022 г.

СОГЛАСОВАНО
Руководитель департамента программной инженерии, доцент факультета компьютерных наук, канд. экон. наук

_____ С. А. Лебедев
«___» _____2022 г.

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл	

ПРОГРАММНЫЙ МОДУЛЬ ДЛЯ ВЫДЕЛЕНИЯ ЛОГИЧЕСКИ СВЯЗАННЫХ ПОТОКОВ В ВЫСОКОСКОРОСТНОМ СЕТЕВОМ ТРАФИКЕ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

ЛИСТ УТВЕРЖДЕНИЯ

RU.17701729.05.03-01 01-1-ЛУ

Исполнитель:
студент группы БПИ197
_____ / Глущенко З. С./
«11» 04 2022 г.

2022

УТВЕРЖДЕНО
RU.17701729.05.03-01 81 01-1-ЛУ

ПРОГРАММНЫЙ МОДУЛЬ ДЛЯ ВЫДЕЛЕНИЯ ЛОГИЧЕСКИ СВЯЗАННЫХ ПОТОКОВ В ВЫСОКОСКОРОСТНОМ СЕТЕВОМ ТРАФИКЕ

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

RU.17701729.05.03-01 81 01-1

Листов 23

Подп. и дата	
Инв. № дубл.	
Взам. инв. №	
Подп. и дата	
Инв. № подл	

2022

СОДЕРЖАНИЕ

АННОТАЦИЯ

В данном программном документе приведена пояснительная записка к программе «Программный модуль для выделения логически связанных потоков в высокоскоростном сетевом трафике».

В разделе «Введение» указано наименование программы, краткое наименование программы и документы, на основании которых ведется разработка.

В разделе «Назначение и область применения» указано функциональное назначение программы, эксплуатационное назначение программы и краткая характеристика области применения программы.

В разделе «Технические характеристики» содержатся следующие подразделы: – постановка задачи на разработку программы; – описание алгоритма и функционирования программы с обоснованием выбора схемы алгоритма решения задачи и возможные взаимодействия программы с другими программами; – описание и обоснование выбора метода организации входных и выходных данных; – описание и обоснование выбора состава технических и программных средств.

В разделе «Ожидаемые технико-экономические показатели» указана предполагаемая потребность и экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Настоящий документ разработан в соответствии с требованиями:

1) ГОСТ 19.101-77 Виды программ и программных документов [1];
2) ГОСТ 19.102-77 Стадии разработки [2];
3) ГОСТ 19.103-77 Обозначения программ и программных документов [3];
4) ГОСТ 19.104-78 Основные надписи [4];
5) ГОСТ 19.105-78 Общие требования к программным документам [5];
6) ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом [6];
7) ГОСТ 19.404-79 Пояснительная записка. Требования к содержанию и оформлению [7].

Изменения к Пояснительной записке оформляются согласно ГОСТ 19.603-78 [8], ГОСТ 19.604-78 [9].

Перед прочтением данного документа рекомендуется ознакомиться с терминологией, приведенной в Приложении 1 настоящей пояснительной записки.

1. ВВЕДЕНИЕ

1.1. Наименование программы

Наименование программы: «Программный модуль для выделения логически связанных потоков в высокоскоростном сетевом трафике».

Наименование программы на английском языке: «Software module for logically connected streams extraction in high speed network traffic».

1.2. Документы, на основании которых ведется разработка

Программа выполнена в рамках темы курсовой работы – «Программный модуль для выделения логически связанных потоков в высокоскоростном сетевом трафике», в соответствии с учебным планом подготовки бакалавров по направлению 09.03.04 «Программная инженерия».

Основанием для разработки является учебный план подготовки бакалавров по направлению 09.03.04 «Программная инженерия» и утвержденная академическим руководителем тема курсового проекта.

2. НАЗНАЧЕНИЕ И ОБЛАСТЬ ПРИМЕНЕНИЯ

2.1. Назначение программы

2.1.1. Функциональное назначение

Программный модуль позволяет выделять логически связанные потоки в сетевом трафике. Получая на вход группу потоков и предоставляя на выходе поток, состоящий из связанных потоков, пакеты которых упорядочены по временным меткам.

2.1.2. Эксплуатационное назначение

Данный программный модуль может быть использован в составе системы анализа трафика, разрабатываемой в ИСП РАН и применяться, когда нужно найти и объединить логически связанные потоки.

2.2. Краткая характеристика области применения

Протокол FTP появился в 1971 и является одним из старейших прикладных протоколов. Построен на архитектуре «клиент-сервер» и использует разные сетевые соединения для передачи команд и данных между клиентом и сервером. Из-за своей простоты и удобства в использовании остается популярным и по сей день.

Для обеспечения безопасности передаваемых данных были разработаны расширения стандарта FTP, обеспечивающие зашифрованность FTP-сессии.

Но далеко не все передачи по протоколу FTP используют шифрование. Поэтому анализ FTP-сессий может позволить зафиксировать противоправные действия и помочь в борьбе с преступностью и пиратством, в частности.

3. ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

3.1. Постановка задачи на разработку программы

Программа должна обеспечивать возможность выполнения следующих функций:

1) Выделять логически связанные потоки на основе сигнатур;
2) Объединять связанные логические потоки в один поток;
3) Выдавать логически связанные потоки как один поток и группу других потоков, которые не связаны логически;

3.2. Описание алгоритма и функционирования программы

Программа состоит из нескольких модулей, реализующих различные функции программы:

- ipc
- Producer
- stream_selector
- packet_filter
- ftp_extractor
- signal_converter
- stream_saver
- stream_merger

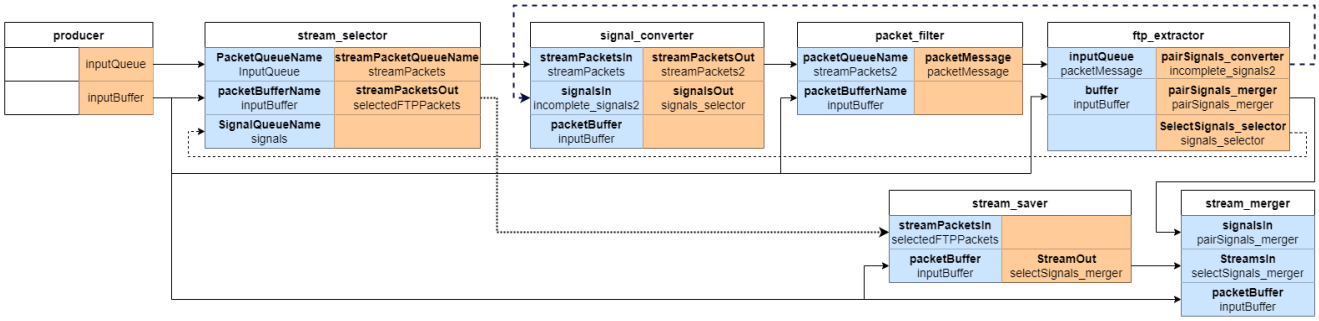

Рисунок 1. Архитектура проекта

Каждый отдельный модуль представляет собой отдельный процесс и для их взаимодействия используется модуль ipc.

Каждый модуль содержит базовые элементы статистики, такие как количество полученных и потерянных пакетов. Если необходимо, то в любом модуле можно определить дополнительные параметры для ведения статистики. Для этого нужно создать поля класса и в реализации метода declareResourcesUsage вызвать метод declareStatParam и передать ему первым параметром название параметра в виде строки, и вторым параметром передать указатель на поле класса.

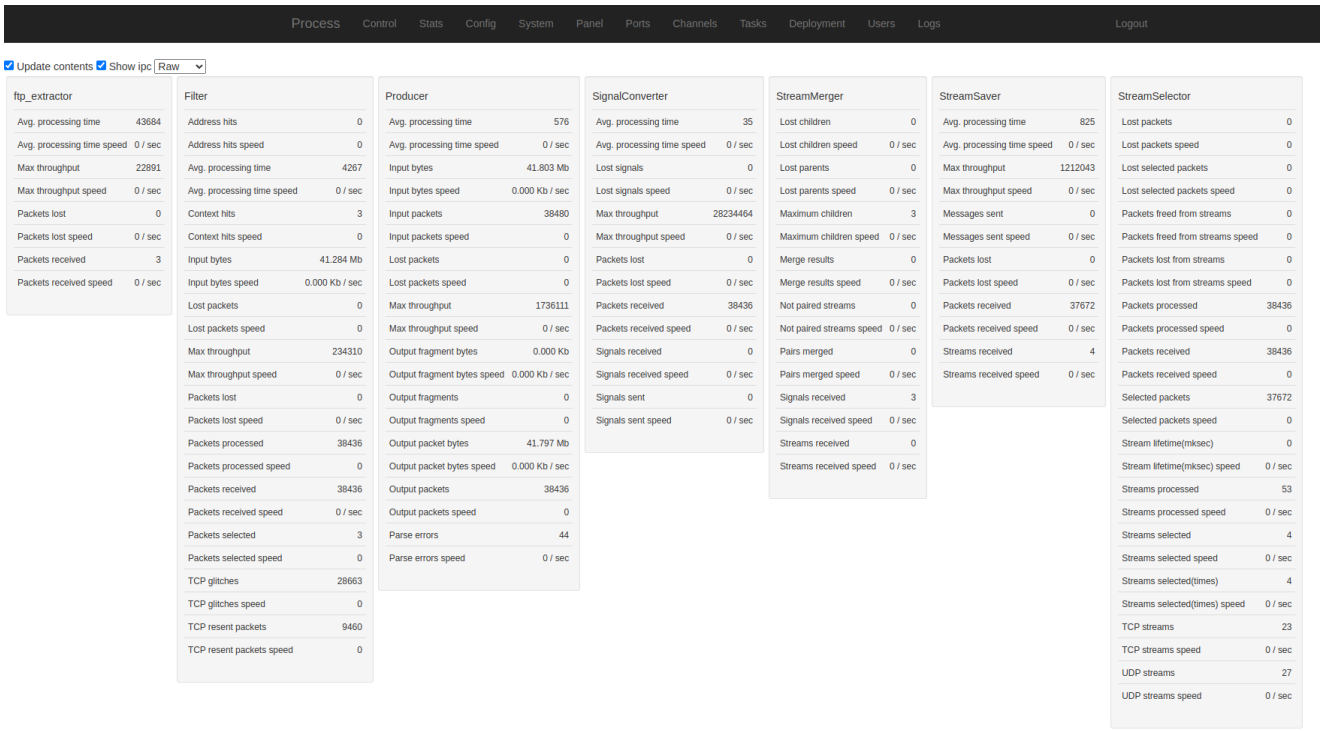

Рисунок 2. Статистика всех запущенных модулей

При запуске модулей появится сноска со статистикой модуля. При этом для модуля ipc такой сноски нет. Как показано на рис. 2 каждый модуль обладает собственным набором параметров, отражающих его текущее состояние.

3.2.1. Описание работы модуля межпроцессного взаимодействия ipc

Данный модуль используется для создания средств взаимодействия между модулями. Каждый модуль представляет из себя отдельно запущенный в системе процесс. Благодаря контролю над ресурсами модуль ipc удается отслеживать регистрацию и состояние других модулей при этом, не являясь их родительским процессом.

При запуске любого модуля проверяется доступность используемых им ресурсов, поэтому самым первым модулем должен запускаться ipc, который их и создаст. Также сразу после запуска модуль начинает ожидать поступление к нему информации через созданную очередь, и обрабатывает ее сразу в момент поступления.

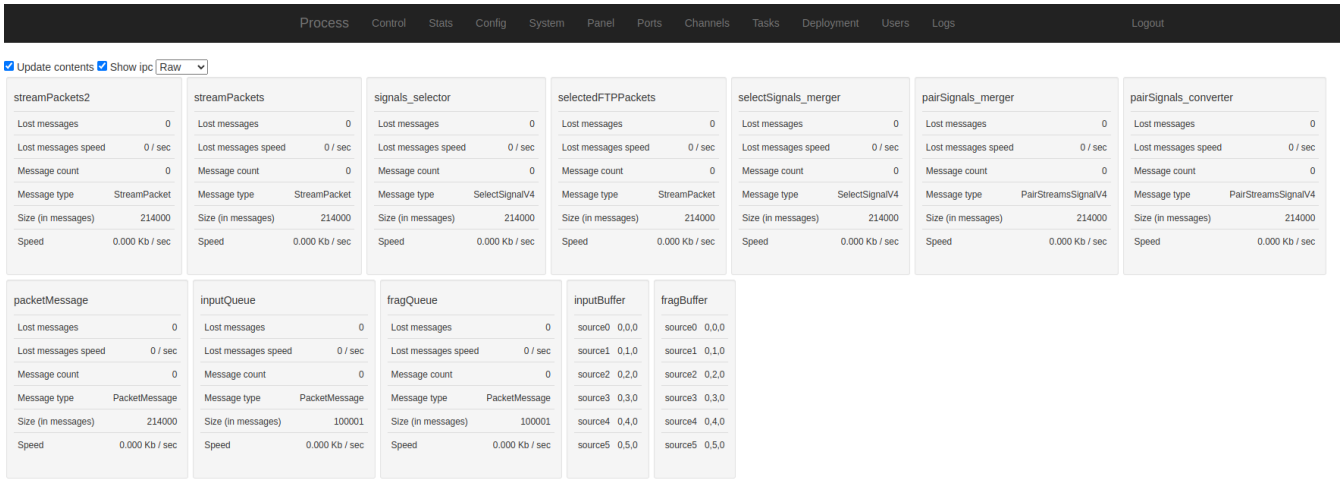

Рисунок 3. Созданные буферы и очереди модулем ipc

3.2.2. Описание работы модуля Producer

Данный модуль отвечает за считывание пакетов. Производить считывание пакетов возможно из:

- Сетевой карты
- Файлов, сохраненных в форматах ERF и PCAP

Полученные пакеты сохраняются в буфер и сообщение о новом пакете передается в выходную очередь. Сообщение содержит указатель на сохраненный пакет в буфере. Вместе с очередью сообщений к каждому модулю подключается и буфер.

Иногда полученный пакет может оказаться фрагментированным. Такие пакеты попадают в отдельные буфер и очередь для дальнейшей обработки и дефрагментации. Но в нашей работе такие не попадались, а потому их отдельная обработка не требуется.

3.2.2.1. Захват передачи сетевых пакетов

Для сбора трассы сетевых пакетов использовалась программа-анализатор трафика для компьютерных сетей Wireshark. В качестве FTP-сервера использовался vsftp. С помощью консольного клиента выполнялось взаимодействие и передача команд FTP-серверу, в то время как Wireshark сохранял сетевую активность в pcap файл. Полученные трассы применялись во время разработки.

3.2.3. Описание работы модуля stream_selector

Данный модуль выполняет несколько функций:

- Группирует полученные пакеты в потоки и присваивает идентификатор потока
- Пересылает обработанные пакеты дальше с идентификатором потока.
- Получает сигналы на отбор и отправляет пакеты соответствующего потока в указанную выходную очередь

Для того чтобы получить пакеты определенного потока нужно указать 5-tuple, время создания и число, соответствующее номеру выходной очереди, куда будут отсылаться пакеты. Время создания пакета используется для решения проблемы использования одной 5-tuple различными потоками во времени. После поступления сигнала создается правило отбора и пакеты, пришедшие до этого и удовлетворяющие правилу, сразу отсылаются, а будущие пакеты будут отосланы сразу по прибытии.

Важной особенностью является то, что *запрос* и *ответ* в рамках одного соединения 5-tuple соответствуют двум потокам, один исходящий и другой входящий соответственно. Чтобы не отправлять два сигнала с переставленными местами отправителем и получателем, у модуля есть параметр *getInversed* в конфигурационном файле.

3.2.4. Описание работы модуля signal_converter

Назначение данного модуля заключается в получение полной адресной информации, зная только частичные адресные признаки. Для этого модуль подключается после stream_selector, для просмотра всех получаемых пакетов, которые содержат идентификатор потока, и еще не были исключены из выборки packet_filter и отправляет пакеты дальше куда они должны были идти.

При нахождении пакета, который соответствует условиям, производиться отправка сигнала stream_selector’у для отбора потока и stream_merger’у для добавления записи о связности найденного потока с ранее полученным от ftp_extractor родительским потоком.

3.2.5. Описание работы модуля packet_filter

Данный модуль используется для фильтрации входящих пакетов. Для этого модуль проверяет поступающие пакеты на входную очередь. В выходную очередь попадают только пакеты, удовлетворяющие указанным требованиям. Если сам пакет и его содержимое не интересно, то можно отправлять сигнал с информацией о прохождении соответствующим пакетом условий фильтра. Выходных очередей может быть несколько, и каждой можно задать отдельное правило попадания в нее. Если пакет не попал ни в одну очередь, так как не подошел по условиям, то такой пакет пропадает. Другими словами, нету выходной очереди для не прошедших пакетов.

Фильтрация может выполняться двумя способами. Эти способы можно применять по отдельности или вместе. Для совместного использования нужно чтобы название совпадали для контекстного и адресного правила . Для указания правил используются регулярные выражения или строки. По ним, используя другой модуль, строится автоматное внутреннее представление.

3.2.5.1. Фильтрация по адресу

Данный вид фильтрации использует информацию, хранящуюся в заголовке пакета. По полученной информации можно наложить ограничение на ip адрес, номера портов, протокол транспортного уровня.

3.2.5.2. Фильтрация по содержимому

Данный вид фильтрации использует информацию, хранящуюся в полезной нагрузке пакета. Просматривая незашифрованное содержимое, можно выделить отправляемые команды между клиентом и сервером. Нас интересуют команды PORT и PASV, которые содержат адрес для нового подключения. Для получения пакетов, содержащих эти команды можно использовать следующее правило с регулярным выражением.

FTP_PASV_PORT (227 Entering Passive Mode)|(PORT ([0-9]{1,3},){5}[0-9]{1,3})

3.2.5.3. Построение автомата

Как видно в примере выше, вначале указывается название правила, а за ним через разделитель, коим может служить знак табуляции, само регулярное выражение. Если нужно указать модулю несколько правил для фильтрации, то каждое новое правило записывается в отдельной строке.

Для построения нужен json файл, задающий пути до проекта, способ фильтрации (контекстный или адресный) и список правил фильтрации. Для каждого правила можно выбрать является ли оно чувствительность к регистру символов и является ли оно строкой или регулярным выражением. Сами правила располагаются в файлах *accept_context.lst* и *accept_addr.lst* для контекстных и адресных правил соответственно. Построенный на этих данных автоматы передаются packet_filter.

3.2.6. Описание работы модуля ftp_extractor

Модуль ftp_extractor определяет способ установления соединения данных, извлекает информацию и выполняет соответствующие действия.

Протокол FTP использует два типа соединений.

- Управляющие соединение
- Соединение данных

3.2.6.1. Управляющие соединение

Для передачи файла по FTP протоколу клиент обращается к FTP серверу по 21 порту, и создает управляющее соединение. Через управляющее соединение отправляются команды на изменение удаленного каталога на FTP сервере. Когда серверная сторона получает через управляющее соединение команду для передачи данных, то устанавливается новое соединение, называемое соединение данных. FTP-сервер передает ровно один файл через это соединение и когда передача завершается, сервер сообщает об этом через управляющее соединение и закрывает соединение данных. Если во время загрузки одного файла пользователь запрашивает передачу другого файла, то для него FTP-сервер открывает еще одно соединение данных. Таким образом, при передаче данных управляющее соединение остается открытым в течение всего пользовательского сеанса, а для передачи данных внутри этого сеанса создается новое соединение данных.

3.2.6.2. Соединение данных

Есть два способа установить соединений данных между сервером и клиентом:

- Активное
- Пассивное

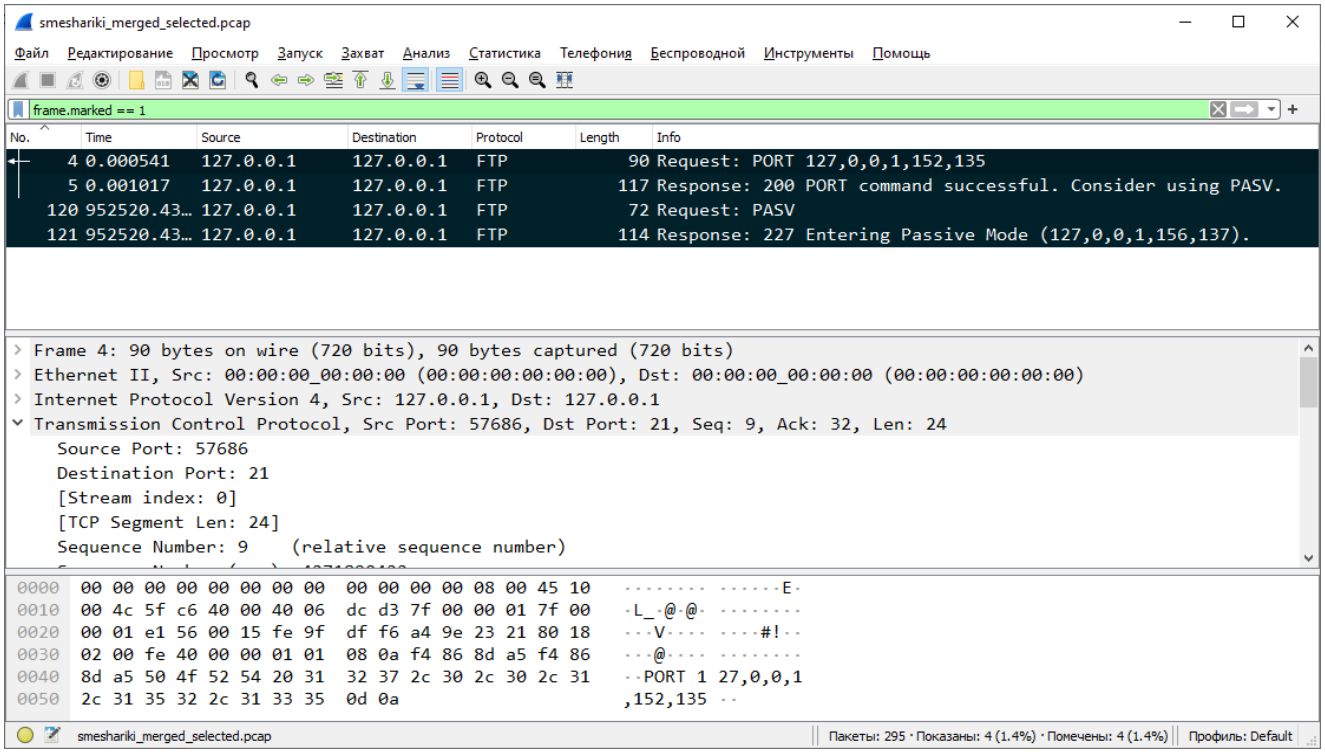

Рисунок 4. Пакеты запросов активного и пассивного режимов содержащие команды PORT и PASV и ответы на них

3.2.6.2.1. Активное подключение

1. Клиент отправляет серверу команду PORT содержащую информацию о ip-адресе и номере порта клиента.
2. Сервер инициирует соединение с клиентом по указанному в команде PORT адресу с исходящим номером порта 20.

Команда PORT имеет вид:

PORT num,num,num,num,num,num

где num — это число от 0 до 254. Соответствующие 8 бит исходного числа. Первые 4 num означают за 32 битный ip-адрес. Оставшиеся 2 num соответствуют 16 битному номеру порта.

Таким образом, в этом режиме нам известны все данные - по какому адресу сервер создаст соединение и будет передать клиенту данные.

3.2.6.2.2. Пассивное подключение

1) Клиент отправляет команду PASV FTP-серверу на порт 21.
2) Сервер отвечает сообщением с информацией о ip-адресе и номере порта, к которому может подключиться клиент.
3) Клиент инициализирует соединение с сервером по указанному адресу

Ответ сервера на команду клиента PASV является:

227 Entering Passive Mode (num,num,num,num,num,num)

В данном случае нам известен только адрес сервера, а о клиенте не известно. Клиент может подключиться с того же ip-адреса, а вот информацию о номере port мы никак не сможем угадать. Для этого нужно будет следить за обращениями по указанному сервером ip-адресу и номеру port в надежде поймать такое соединение. Для этой цели используется signal_converter.

3.2.6.2.3. Извлечение информации

Отобранные packet_filter пакеты передаются на вход в одну очередь. Для идентификации типа пакета проверяется первые 4 символа. Ими могут быть “PORT“ или “227 “. Если полученные пакеты содержат отличные от этих символы в начале полезной нагрузки пакета, то увеличивается счетчик ошибочных пакетов, а над самим пакетом не выполняются никакие действия. Если пакет прошел проверку, то выполняется сдвиг на определенное количество байт в зависимости от типа соединения и начинается посимвольное чтение записанных чисел, разделенных запятыми. При успешном считывании 6 чисел создаются сигналы, в которые записывается полученная информация, и отправляются в соответствующие очереди.

При получении пакета с командой PORT сигналы отправляются 3 сигнала. Один stream_merger о связи потоков и два stream_selector об отборе родительского и дочернего потоков.

При получении пакета на команду PASV сигналы отправляются в две очереди. Одна signal_converter для нахождения полного адресного признака дочернего потомка и другая stream_selector для отбора родительского потока.

3.2.7. Описание работы модуля stream_saver

Функционал данного модуля заключается в сохранении потоков в файлы. Поскольку поток является некоторой абстракцией, а не единой сущностью, то в качестве потока передаются пакеты с идентификатором конкретного потока и дополнительной информацией, обозначающей является ли конкретный пакет началом или концом потока. Имя выходного файла формируется из 5-tuple и времени создания. Под временем создания понимается время получения первого пакета в потоке. Это нужно на случай, чтобы различать соединения, использующие одинаковые 5-tuple.

Эти данные используются в следующем формате: *<протокол>_<ip адрес отправителя>_<номер port отправителя>_<ip адрес получателя>_<номер port получателя>_<время создания>*

Пример имени файла: *tcp_192.168.50.102_61692_192.168.50.75_21_1648976991362416000*

Каждый поступивший пакет проверяется на несколько критериев.

- Если он является концом потока, то файл, в который он сохранялся закрывается запись и отправляется сигнал stream_merger и заканчивается обработка.

- Если он содержит информацию о том, что он является первым пакетов в новом потоке, то выполняется закрытие файла на запись и отправляется сигнал stream_merger, если таковой существовал для подобного stream_id. После, независимо от существования предыдущего файла, создается новый файл и соответствующий новому потоку.

3.2.8. Описание работы модуля stream_merger

Данный модуль используется для объединения логически связанных потоков, сохраненных в формате pcap файла. Для работы модуль использует две входные очереди различных типов и предназначений. К каждой очереди назначен соответствующий обработчик, который назначается при запуске модуля.

- Для очереди типа PairStreamSignalV4, содержащей информацию о связанности двух потоков отношением родитель-потомок, используется метод processSignal.
- Для обработки сигналов очереди типа SelectSignalV4, которые содержат информацию о готовности конкретного потока для объединения, используется метод processMessage.

3.2.8.1. Описание метода processSignal

Для обработки полученного сигнала используется хеш-таблица. Ключом для хеш-таблицы является структура данных AddrInfoV4 реализующая 5-tuple, а значением указатель на структуру, под инстанс которой выделяется место из пула. Эта структура содержит информацию о родительском потоке и связанных с ним потомках.

В информацию о потомках входит:

- число полученных сигналов, о связанности с этим родителем.
- число потомков, файлы сетевых трасс которых сохранены и готовы к объединению.
- Дек, хранящий указатели на полученных потомков.

Обрабатывая полученный сигнал выполняется поиск в хеш-таблице на наличие родительского потока, с помощью значений 5-tuple. Если такой записи еще нет, то

1. Выделяется место в пуле родительских потоков.
2. По полученному указателю записывается информация о родительском потоке.

3. Выполняется вставка полученного указателя в хеш-таблицу с 5-tuple родительского потока в качестве ключа.

Таким образом у нас есть запись о родительском потоке в хеш-таблице, который хранит информацию о самом себе.
Далее к полученной или ранее существовавшей информации о родительском потоке добавляется информация о дочернем потоке, в виде инкрементирования числа. Также добавляется запись в хеш-таблицу с 5-tuple дочернего потока в качестве ключа и указателем на информацию о родительском.
Данная запись потребуется позже для определения сигналов в processMessage

3.2.8.2. Описание метода processMessage

Данный метод обрабатывает поступивший сигнал следующим образом:

1. В начале выполняется поиск записи в хеш-таблице из информации в полученном сигнале.
2. Найденная запись проверятся на соответствие:
    a. Если получен родительский поток, то проверяется равно ли количество записей о сохраненных файлах потомков их количеству.
        - Если да, то начинается процесс объединения файлов с помощью метода startMerge.
        - Если нет, то родитель помечается как посещенный.
    b. Если получен дочерний поток, то
        1. Увеличивается число, соответствующее количеству сохраненных файлов для родительского потока.
        2. Выделяется место из пула дочерних потоков.
        3. По указателю на это место записывается информация о потомке
        4. Этот указатель добавляется в дек к родительскому потоку.
        5. Если после добавления потомка количество записей о сохраненных файлах потомков равно их количеству и родитель уже был посещен, то запускается метод для объединения файлов startMerge.
    c. Иначе если в хеш-таблице не была найдена соответствующая запись, то увеличивается счетчик не парных потоков и удаляется файл с соответствующим названием, если таковой изначально существовал.

3.2.8.3. Описание метода startMerge

Данный метод отвечает за выполнения слияния файлов и последующее их удаление, а также записей о полученных сигналах. Для этих целей он последовательно вызывает два других метода, передавая им нужные параметры.

В начале он вызывает метод mergeStreams передавая ему указатель на информацию о родителе. Вызванный метод начинает создавать команду, с помощью конкатенации

результата предыдущей итерации с файлом который нужно добавить в объединение. Таким образом команда имеет следующий вид:

mergecap -w <путь для получившихся объединений> <пути до файлов для объединения>

После всех конкатенаций полученная команда выполняется и в случае успеха, увеличиваются счетчики количества объединенных пар. В случае неудачи увеличиваются счетчики потерянных пакетов.

После вызова метода mergeStreams, происходит освобождение использованных ресурсов, тем самым удаляются из пулов родительский и дочерние потоки и вызывается метод removeStream также для удаления записей о них из хеш-таблицы и соответствующие им файлы на диске.

3.3. Описание и обоснование выбора метода организации входных и выходных данных

3.3.1. Описание метода организации входных данных

Входными данными для системы является трасса пакетов, получаемая модулем Producer, который может считать ее из сохраненного файла или сетевой карты.

3.3.2. Описание метода организации выходных данных

Выходным данным являются pcap файлы, каждый из которых состоит из пакетов входящих в группу логически связанных потоков

3.4. Описание и обоснование выбора состава технических и программных средств

3.4.1. Состав технических и программных средств

Для надежной и бесперебойной работы программы требуется следующий состав технических средств:

1) Процессор Intel Core i7 с тактовой частотой не менее 3,5 Ггц;
2) Не менее 8 ГБ оперативной памяти;
3) Не менее 20 ГБ свободного места на жестком диске;
4) Монитор с разрешением экрана 1920x1080;
5) Клавиатура и компьютерная мышь;

Для нормального функционирования программы требуется компьютер, оснащенный следующими программными средствами:

1) Система автоматизации сборки программного обеспечения из исходного кода CMake 3.5.1;
2) Установленный компилятор g++ 9.3.0;
3) Python версии 3.5 или выше;

3.4.2. Обоснование выбора технических и программных средств

3.4.2.1.Обоснование выбора состава технических средств

Минимальные требования, описанные в 3.4.1 настоящего документа, являются необходимыми для запуска и дальнейшей работы модулей, из которых состоит система анализа трафика, разрабатываемая в ИСП РАН.

3.4.2.2.Обоснование выбора состава программных средств

Выбранные программные средства являются необходимыми для сборки проекта.

4. ОЖИДАЕМЫЕ ТЕХНИКО-ЭКОНОМИЧЕСКИЕ ПОКАЗАТЕЛИ

4.1. Предполагаемая потребность

Данное решение может быть интересно сетевым администраторам или интернет провайдерам. Также оно способствует расширению возможностей существующей инфраструктуры в рамках, которой оно и разрабатывалась.

4.2. Предполагаемая потребность

Данное решение входит в исходный проект, расширяя его функционал. Включив его в сборку, поставляемую клиентам и предоставив им его вместе с другими инструментами для анализа трафика, можно повысить конечную стоимость для клиента, тем самым увеличив доход с продаж.

4.3. Экономические преимущества разработки по сравнению с отечественными и зарубежными образцами или аналогами

Главным преимуществом разработки является ее интегрированность в существующую инфраструктуру, с возможностью дальнейшего использования

5. ИСТОЧНИКИ, ИСПОЛЬЗОВАННЫЕ ПРИ РАЗРАБОТКЕ

1. ГОСТ 19.101-77 Виды программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
2. ГОСТ 19.102-77 Стадии разработки. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
3. ГОСТ 19.103-77 Обозначения программ и программных документов. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
4. ГОСТ 19.104-78 Основные надписи. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
5. ГОСТ 19.105-78 Общие требования к программным документам. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
6. ГОСТ 19.106-78 Требования к программным документам, выполненным печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
7. ГОСТ 19.201-78 Техническое задание. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
8. ГОСТ 19.603-78 Общие правила внесения изменений. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
9. ГОСТ 19.604-78 Правила внесения изменений в программные документы, выполненные печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
10. ГОСТ 15150-69 Машины, приборы и другие технические изделия. Исполнения для различных климатических районов. Категории, условия эксплуатации, хранения и транспортирования в части воздействия климатических факторов внешней среды. – М.: Изд-во стандартов, 1997.
11. ГОСТ 19.301-79 Программа и методика испытаний. Требования к содержанию и оформлению. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
12. ГОСТ 19.602-78 Правила дублирования, учета и хранения программных документов, выполненных печатным способом. //Единая система программной документации. – М.: ИПК Издательство стандартов, 2001.
13. Джеймс Куроуз. Компьютерные сети. Нисходящий подход. // Джеймс Куроуз, Кит Росс – 6-е изд., Москва, 2016

ПРИЛОЖЕНИЕ 1

ТЕРМИНОЛОГИЯ

Термин	**Определение**
Тракт	Именованный контейнер, который содержит набор модулей с фиксированными конфигурациями, которые требуется запустить. Позволяет организовывать модули в иерархические структуры.
Сетевой пакет	Передаваемый по сети блок данных, сформированный определенным образом. Состоит из двух частей данных: управляющей информации и полезной нагрузки. Управляющая информация содержит данные, необходимые для доставки данных пользователя.
Полезная нагрузка	Часть содержимого пакета данных без служебной информации. То есть данные без учета заголовка, битов синхронизации и т. д.
IP адрес	Уникальный числовой идентификатор устройства в компьютерной сети, работающей по протоколу IP
Номер port	Целое неотрицательное 16 битное число, записываемое в заголовках протоколов транспортного уровня
Протокол передачи данных	Набор определённых правил или соглашений интерфейса, который определяет обмен данными между различными источниками. Источниками в зависимости от уровня протокола, могут выступать отдельные узлы сети или программы. Протокол вышестоящего уровня опирается на функционал протокола нижестоящего уровня и расширяет его возможности. Использование протоколов позволяют стандартизировать способ передачи сообщений и обработки ошибок.
5-tuple	Кортеж, состоящий из 5 элементов: ● IP адрес отправителя ● Номер port отправителя ● IP адрес получателя ● Номер port получателя Протокол (TCP | UDP)
Фрагментация пакета	Генерация двух сетевых пакетов из одного. Происходит при превышении длины кадра MTU интерфейса, через который он в данный момент проходит.
Указатель	Переменная, которая хранит *(указывает на)* адрес ячейки памяти. Указатель может хранить адрес переменных, констант, указателей или функции.
Хеш-таблица	Структура данных, реализующая интерфейс ассоциативного массива. Позволяет хранить пары (ключ, значение) и выполнять

	три операции: операцию добавления новой пары, операцию поиска и операцию удаления пары по ключу.
Дек	Структура данных, представляющая из себя список элементов, в которой добавление новых элементов и удаление существующих возможно выполнить с обоих концов.
Регулярные выражения	Используемый в компьютерных программах, работающих с текстом, формальный язык поиска и осуществления манипуляций с подстроками в тексте, основанный на использовании метасимволов. Для поиска используется строка-образец (паттерн), состоящая из символов и метасимволов и задающая правило поиска.
Конфигурационный файл	Файл, содержащий необходимые настройки для работы программы, которые можно редактировать для изменения поведения исходной программы

ЛИСТ РЕГИСТРАЦИИ ИЗМЕНЕНИЙ

Лист регистрации изменений									
Номера листов (страниц)					Всего листов (страниц в докум.)	№ документа	Входящий № сопроводительного докум. и дата	Подп.	Дата
Изм.	Измененных	Замененных	Новых	Аннулированх					

Изм.	Лист	№ докум.	Подп.	Дата
RU.17701729.05.03-01 81				
Инв. № подл.	Подп. и дата	Взам. инв. №	Инв. № дубл.	Подп. и дата