Государственное образовательное учреждение высшего профессионального образования

«Московский Государственный Технический Университет имени Н.Э. Баумана» (МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «ИНФОРМАТИКА И СИСТЕМЫ УПРАВЛЕНИЯ»
КАФЕДРА «ИНФОРМАЦИОННАЯ БЕЗОПАСНОСТЬ»

КУРСОВОЙ ПРОЕКТ

Разработка серверной части для системы контроля сетевого трафика

Руководитель проекта: _____ /Мацак И. В.
(подпись, дата)

Разработчик проекта: _____ /Мишков А. О.
(подпись, дата)

Москва 2020

Оглавление

Цель

Создание серверной части для системы контроля сетевого трафика с возможностью выбора клиента и возможностью расширения функционала приложения

Основные определения

Алгоритмический язык программирования - формальный язык, используемый для записи, реализации и изучения алгоритмов. В отличие от большинства языков программирования, алгоритмический язык не привязан к архитектуре компьютера, не содержит деталей, связанных с устройством машины.

Объектно-ориентированное программирование (ООП) - методология программирования, основанная на представлении программы в виде совокупности объектов, каждый из которых является экземпляром определенного класса, а классы образуют иерархию наследования.

Интерфейс - общая граница между двумя функциональными объектами, требования к которой определяются стандартом. Это совокупность средств, методов и правил взаимодействия между элементами системы.

Введение

“Контроль сетевого трафика в реалиях карантина необходим”, именно так считают многие работодатели, поэтому мы и решили сделать такую программу, серверная часть это некий хаб, где собираются и сортируются данные, поэтому, я считаю что сервер должен быть быстрым и его дизайн должен быть понятным и удобным, из за этих причин я выбрал язык python3 с подключенным pyqt5. Python является очень удобными языком для написания серверов, так как обладает большим функционалом, он позволяет описать сервер в пару строк, когда в других языках на это уходит куда больше. Так же программы на Python весят сравнительно мало, а я считаю, что это хорошее качество для сервера.

Основная идея

Основной идеей является необходимость в отслеживании сетевого трафика людей, например сотрудников компании. Для этого необходима программа на компьютере сотрудника (сниффер) и программа для сбора и упорядочивания данных (сервер), сервер должен уметь и иметь: 1) визуальную составляющую (интерфейс) 2) отправлять запросы разным клиентам 3)уметь открывать присланный файлы 4)упорядочивать данных по времени и “человеку”

Разработка программы

Разработку программы стоит начать с моделирования дизайна в Qt Designer, эта программа позволяет создать макет дизайна (кнопки, диалоговые окна и поля ввода-вывода). Создав макет его надо конвертировать в код и получить класс Ui_MainWindow.

В нем содержится дизайн в виде кода именно, например:

1) self.pushButton_2 = QtWidgets.QPushButton(self.widget)- код описывающий кнопку, создается кнопка нажатие на которую можно привязать к определенному действию. pushButton_2- пременная называющая эту кнопку
2) self.horizontalLayout.addWidget(self.pushButton_2) – привязка этой кнопки к горизонтальной симметрии, помогает центрировать несколько объектов
3) self.chatTextField=QLineEdit(self)- поля для ввода-вывода текста, можно присвоить атрибут .setReadOnly(True) что бы использовать только для вывода текста

editorProgram = 'notepad' -выбор программы для редактирования и просмотра файлов, ‘notepad’ можно заменить на похожие программы, именно он был выбран из за быстродействия и его простоты

Далее идет класс ExampleApp (QtWidgets.QMainWindow, Ui_MainWindow, QDialog)- используется для расширения макета дизайна и описания функций этого макета:

1) self.pushButton.clicked.connect(self.send)- команда для привязки нажатия кнопки к определенному действию(функции) в нашем случае это функция send
2) def Open- позволяет открыть файл с выбранными нами расширениями, так как я записываю все в текстовом формате, то и выбрал соответственно .txt
3) def send- позволяет отправить команду start определенному клиенту, предварительно введя его ник в поле для ввода, так же его ник добавляется в ‘реестр’ и показывается на экране
4) def close- по аналогии с def send только отправляется команда close и ник не появляется в реестре

Следующий класс уже один из двух серверных class ServerThread(Thread)- позволяет клиентам подключаться в локальной сети и дает возможность пересылать сообщения, реализована многопоточность, порт и айпи адрес можно легко поменять, что дает возможность быстрой развертки и настройки. Сервер всегда находится в режиме ‘прослушки’ что дает возможность клиенту подключиться в любое время

lass ClientThread(Thread)- отвечает за работу с клиентом, а именно:

1. выводит айпи и порт подключившегося
2. позволяет получать и декодировать сообщение от пользователя
   2.1. при получении сообщения длинной 5 символов (это ник пользователя изначально заложенный в клиент) создает файл в формате: ник_годмесяцдень-час (при нажатии на кнопку file можно просмотреть файл и взять оттуда ник)
   2.2. при получении сообщения длинной <3 знаков файл начинает структурироваться и форматироваться для нормального ознакомления с ним
   2.3. когда длинна сообщения не соответствует вышеперечисленным значениям то эти сообщения идут в файл упомянутый в пункте 2.1

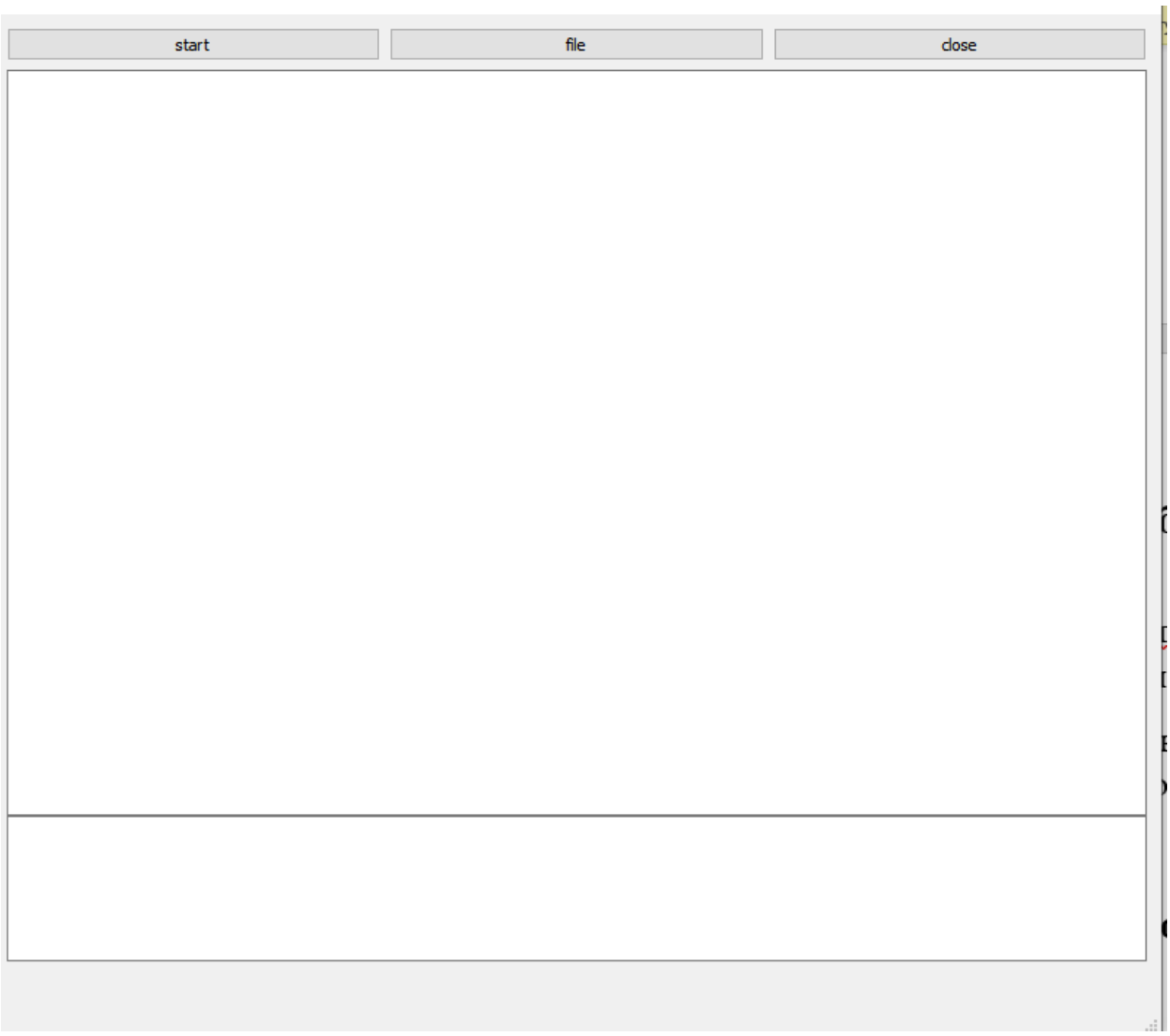
def main – запускает сам виджет и сервер

Выбор технологий

Python как язык программирования для данной задачи подходит как никто другой, я его выбрал за минималистичность кода, быстродействие, легкий синтаксис и то, что его легко расширить в будущем.

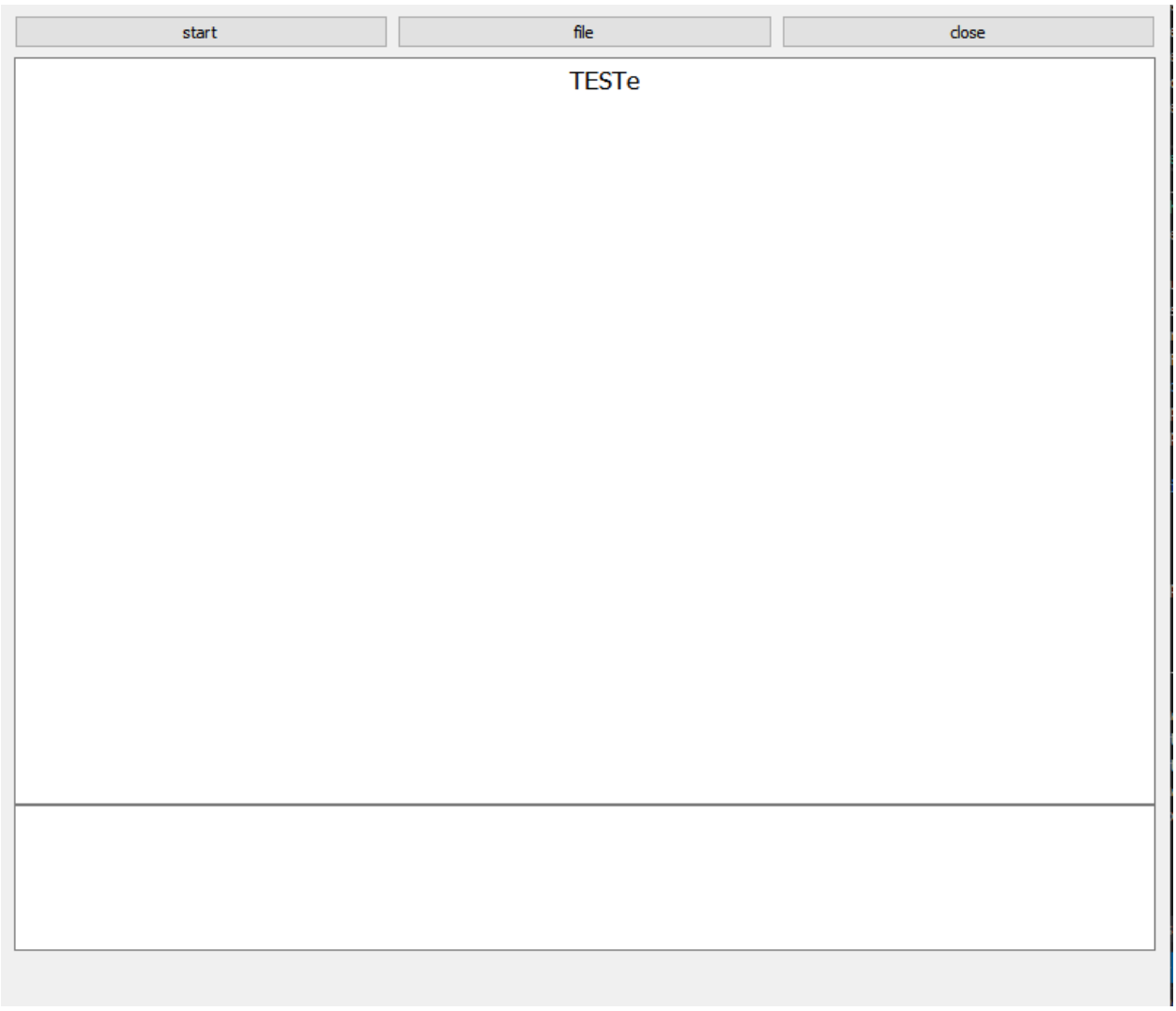
Pyqt5 выбран из за того что это практически единственная библиотека с возможностью простого и быстрого дизайна

Работа программы



Стартовый экран при запуске

Созданный автоматически файл от пользователя TESTe в 23 часа



‘реестр’ с никами, ник надо вводить в поле ниже

b'Device: en0
b'Number of packets: 50
b'Filter expression: ip
b'
b'Date and time of capture: 25.12.2020 22:0
b'Packet number 1:
b'From: 192.168.1.102
b'To: 192.168.1.68
b' Protocol: TCP
b' Src port: 548
b' Dst port: 51364
b' Payload (16 bytes):
b'00000 00 05 4f 02 00 00 00 00 00 00 00 00 00 00 00 00 ..O..
b'
b'Date and time of capture: 25.12.2020 22:0
b'Packet number 2:
b'From: 192.168.1.68
b'To: 192.168.1.102
b' Protocol: TCP
b' Src port: 51364
b' Dst port: 548
b'
b'Date and time of capture: 25.12.2020 22:0
b'Packet number 3:
b'From: 192.168.1.68
b'To: 192.168.1.1
b' Protocol: UDP
b'
b'Date and time of capture: 25.12.2020 22:0
b'Packet number 4:
b'From: 192.168.1.1
b'To: 192.168.1.68
b' Protocol: UDP
b'
b'Date and time of capture: 25.12.2020 22:0
b'Packet number 5:
b'From: 192.168.1.68
b'To: 192.168.1.1
b' Protocol: UDP
b'
b'Date and time of capture: 25.12.2020 22:0
b'Packet number 6:
b'From: 192.168.1.1
b'To: 192.168.1.68
b' Protocol: UDP

```
b'Date and time of capture: 25.12.2020 22:0
b'Packet number 15:
b'From: 192.168.1.68
b'To: srv186-129-240-87.vk.com
b'   Protocol: TCP
b'   Src port: 51811
b'   Dst port: 443
b'   Payload (55 bytes):
b'00000   17 03 03 00 32 62 fb a4  82 cf d3 1b b9 bf 3e 52    ....2b.....>R
b'00016   d3 f4 4d 93 ca 6f 81 8d  43 ae 56 d9 ad af 0f ba    ..M..o..C.V.....
b'00032   84 8b 49 1f f0 fd eb 3e  c8 c9 0c 6c de bf 89 8d    ..I....>...l....
b'00048   54 20 bb 86 a4 b6 e9                                T .....
b'
b'Date and time of capture: 25.12.2020 22:0
b'Packet number 16:
b'From: 192.168.1.68
b'To: srv186-129-240-87.vk.com
b'   Protocol: TCP
b'   Src port: 51811
b'   Dst port: 443
b'   Payload (143 bytes):
b'00000   17 03 03 00 8a b4 f4 c1  f0 c3 c5 35 07 d7 ad 1c    ...........5....
b'00016   ae 2a bc bb 76 52 7e ef  6d 71 9c 1e 7e 44 d2 cb    .*..vR~.mq..~D..
b'00032   cd e6 46 13 f7 2c 72 3f  a1 34 92 fc d5 98 e9 2e    ..F..,r?.4......
b'00048   84 ff f1 0c 8c 77 85 1e  18 12 9b 7b 55 29 be eb    .....w.....{U)..
b'00064   52 d4 bc ca 4a 30 dc 97  d4 1b 30 25 88 24 9f 23    R...J0....0%.$.#
b'00080   c7 75 85 70 77 a9 e4 89  2a 34 8a 53 3c ba bb 92    .u.pw...*4.S<...
b'00096   50 2c e1 86 9c 03 6c 48  ae 0e b8 89 6e 2b fa ff    P,....lH....n+..
b'00112   45 5c 67 17 12 25 88 4e  e8 c6 90 22 ba f3 8b 3c    E\\g..%.N..."...<
b'00128   1c fa 15 d1 89 35 c9 5e  d4 b0 9a 92 3d 67 bc       .....5.^....=g.
b'
```

Пример присланного и обработанного файла

Заключение

В процессе выполнения курсового проекта бы частично изучен pyqt5 и получены навыки реализации серверных приложений. Был реализован локальный сервер с системой файлов и их форматирования

Список используемых источников

1. Документация: документация по Python. URL: https://docs.python.org/3/
2. Документация:Pyqt5. URL: https://doc.qt.io/qtforpython/