МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«Национальный исследовательский университет ИТМО»

ФАКУЛЬТЕТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ

Управление мобильными устройствами

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

«Обработка и тарификация трафика NetFlow»

Вариант №6

Выполнил студент
группы N3352
Захарова Д.И.

Проверил:
Федоров И. Р.

Санкт-Петербург

2020

Цель работы: реализация программного модуля для обработки CDR и тарификации абонента.

1. Привести данный файл в читабельный вид (проще всего это сделать с помощью утилиты nfdump nfdump -r nfcapd.202002251200
2. Сформировать собственный файл для тарификации любого формата, с которым удобно работать (в соответствии с вариантом работы)
3. Построить график зависимости объема трафика от времени (любым удобным образом)
4. Протарифицировать трафик абонента с IP-адресом 192.168.250.1 с коэффициентом k: 0,5руб/Мб первые 500Кб, после каждых последующих 500Кб k увеличивается на 0,5руб. (Мб были заменены на Кб в соответствии с примечанием к заданию)

Описание выбранных средств реализации и обоснования выбора:

Для реализации был выбран язык Python (среда разработки PyCharm).

Python - интерпретируемый объектно-ориентированный язык программирования высокого уровня с динамической типизацией, автоматическим управлением памятью и удобными высокоуровневыми структурами данных, такими как словари (хэш-таблицы), списки, кортежи. Интерпретатор Python реализован практически на всех платформах и операционных системах. Язык поддерживает классы, модули (которые могут быть объединены в пакеты), обработку исключений, а также многопоточные вычисления. Поддерживаются несколько парадигм программирования: структурное, объектно-ориентированное, функциональное и аспектно-ориентированное.

Python обладает простым и выразительным синтаксисом. В то же время стандартная библиотека включает большой объём полезных функций, в частности, модуль csv для работы с файлами CSV-формата. Этот модуль предназначен для работы с различными диалектами: разделитель-запятая, разделитель — точка с запятой, разделитель — табуляция (Excel). Также в Python есть библиотека для визуализации данных Matplotlib, удобная в построении графиков. Таким образом использование языка Python существенно упрощает процесс работы с данными.

Исходный код:

В main.py обрабатывается файл с NetFlow трафиком, проводится тарификация пользователя (с помощью tariffication.py), а также строится график зависимости объема трафика от времени.

main.py

```
import csv
from tariffication import Tariffication
import matplotlib.pyplot as plt
from matplotlib import dates
import datetime as dt
import os

command = "nfdump -r nfcapd.202002251200 'src ip 192.168.250.1 or dst ip
192.168.250.1' -o csv -q > data.csv"
os.system(command)

def lineplot(x_data, y_data, x_label="", y_label="", title=""):
    fmt = dates.DateFormatter('%H:%M:%S')
    _, ax = plt.subplots()
    time_interval = x_data
    time_interval = [dt.datetime.strptime(i, "%H:%M:%S") for i in time_interval]
    ax.plot(time_interval, y_data, lw = 2, color = '#216A9C', alpha = 1)
    ax.xaxis.set_major_formatter(fmt)
    ax.set_title(title)
    ax.set_xlabel(x_label)
    ax.set_ylabel(y_label)
    plt.show()

user = Tariffication("192.168.250.1")

with open('data.csv') as file:
    reader = csv.reader(file)
    for row in reader:

        user.addTrafficIn(row[1], int(row[12]))
        user.addTrafficOut(row[1], int(row[14]))
        user.addTimeVal(row[1], int(row[12]))
```

```
        user.addTimeVal(row[1], int(row[14]))

    user.userTariffication()

    times = list(user._time_val.keys())
    val = list(user._time_val.values())

    for i in range(1, len(val)):
        val[i] += val[i-1]

    lineplot(times, val, "time", "bytes", "Value(t)")
```

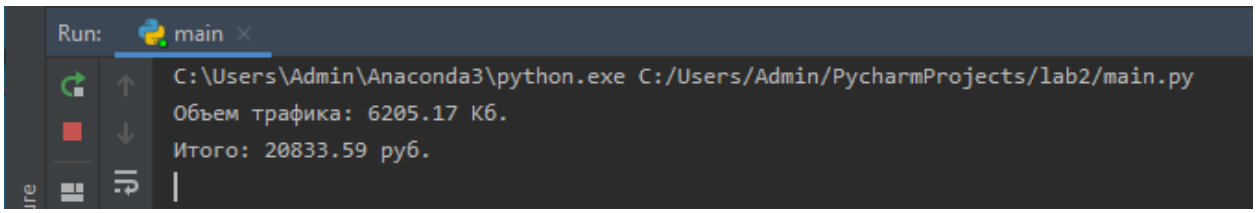
tariffication.py – модуль для тарификации пользователя

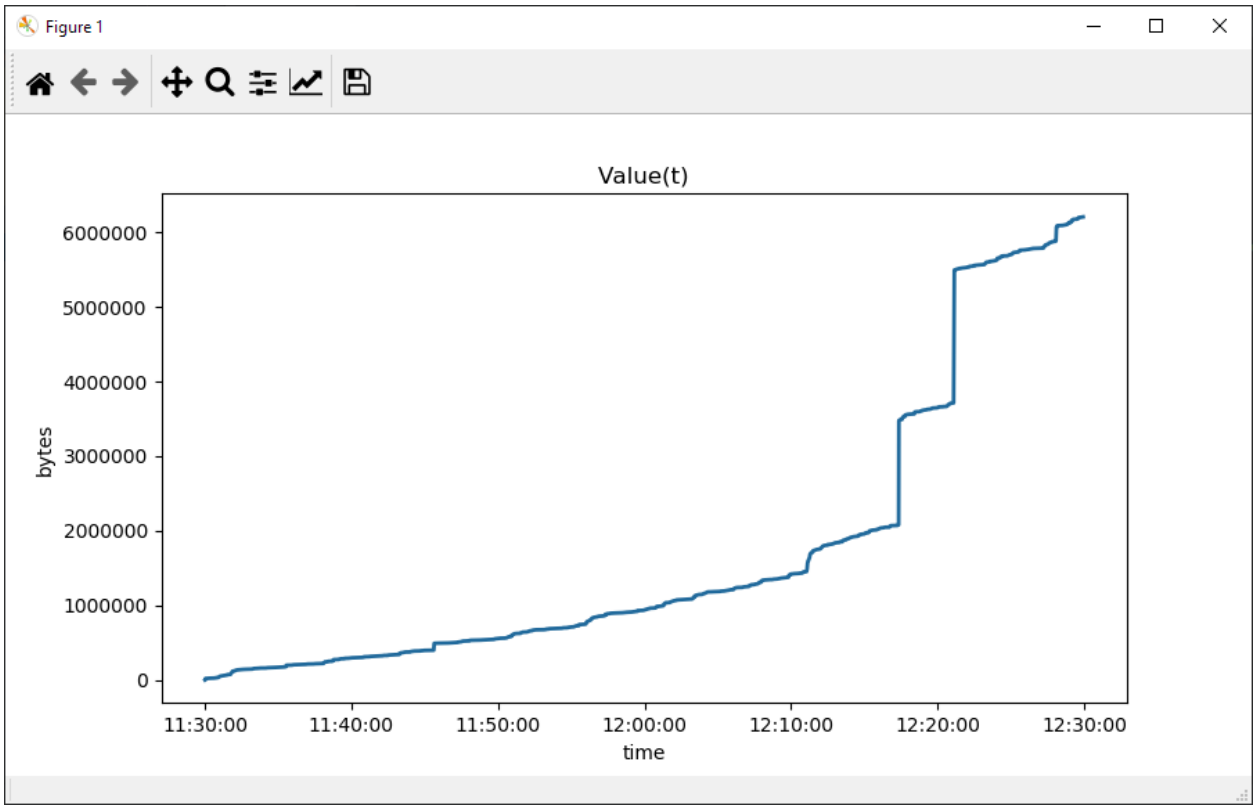
tariffication.py

```
class Tariffication:
    tarif_rate = 0.5

    def __init__(self, ip):
        self._ip = ip
        self._traffic_in = {}
        self._traffic_out = {}
        self._time_val = {}

    def getId(self):
        return self._ip

    def addTrafficOut(self, time, bytes):
        if time in self._traffic_out.keys():
            self._traffic_out[time].append(bytes)
        else:
            self._traffic_out[time] = [bytes]

    def addTrafficIn(self, time, bytes):
        if time in self._traffic_in.keys():
            self._traffic_in[time].append(bytes)
        else:
            self._traffic_in[time] = [bytes]

    def addTimeVal(self, time, bytes):

        time = time.split(" ")[1]
        if time in self._time_val.keys():
            self._time_val[time] = self._time_val[time] + bytes
        else:
            self._time_val[time] = bytes

    def userTariffication(self):

        k = self.tarif_rate
        total_traffic_out = 0
        for i in self._traffic_out.keys():
            for j in self._traffic_out[i]:
                total_traffic_out += j

        total_traffic_in = 0
        for i in self._traffic_in.keys():
            for j in self._traffic_in[i]:
                total_traffic_in += j
```

```
        total_traffic = total_traffic_out + total_traffic_in
        total_traffic_bill = 0

        k = self.tarif_rate
        while total_traffic >= 500*1000:
            total_traffic -= 500*1000
            total_traffic_bill += 500*k
            k += 0.5

        total_traffic_bill += (total_traffic/1000)*k

        print("Объем трафика:", round((total_traffic_out + total_traffic_in)/1000,
2), "Кб.")
        print("Итого:", round(total_traffic_bill, 2), "руб.")
```

Демонстрация работы программы:





Вывод: в процессе выполнения работы были изучены основные принципы работы с протоколом NetFlow, а также реализован программный модуль для обработки, просмотра статистики (график) и тарификации трафика NetFlow.