

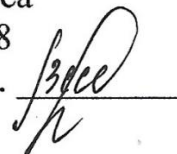
МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ

Федеральное государственное автономное
образовательное учреждение высшего образования
«ИТМО»

**ФАКУЛЬТЕТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ**

Лабораторная работа №2
Обработка и тарификация трафика *NetFlow*
Вариант 1

Работу выполнил
студент 3 курса
группы N3348
Веселова С.С.



Проверил
преподаватель:

(Таранов С. В.)

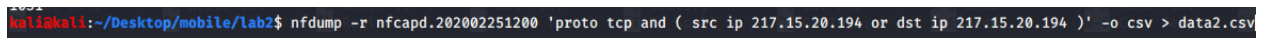
Задача:

- Протарифицировать абонента с IP-адресом 217.15.20.194 с коэффициентом k: 0,5руб/Мб.

Ход работы:

1. Описание выбранных средств реализации и их преимущества
 - 1.1. Язык программирования – Python
 - 1.1.1. Преимущества
 - низкий порог вхождения
 - кроссплатформенность
 - обширный выбор библиотек
 - широкое применение
 - 1.2. Редактор – nano
 - 1.2.1. Критерии выбора
 - простота и удобство пользования
 - возможность использования «горячих» клавиш
2. Приведение данного файла в читабельный вид и формирование собственного файла для тарификации

Рисунок 1. Результат выполнения nfdump -r



```
kali@kali:~/Desktop/mobile/lab2$ nfdump -r nfcapd.202002251200 'proto tcp and ( src ip 217.15.20.194 or dst ip 217.15.20.194 )' -o csv > data2.csv
```

3. Исходный код и результат работы программы

Рисунок 2. Исходный код

```
GNU nano 4.5 /home/.../lab2$ python3 file1.py
import csv
import matplotlib.pyplot as plt

with open('data2.csv') as data:
    readData = csv.reader(data, delimiter=',')

# row[12] - In Byte
# row[3] - Src IP Addr
# row[4] - Dst IP Addr
# row[0] - Date First Seen

    countByte = 0
    k = 0.5
    arrayTime = []
    arrayByte = []

# tariffing
    for row in readData:
        if len(row) == 48:
            if row[3] == '217.15.20.194':
                countByte += int(row[12])
                arrayByte.append(int(row[12]))
                arrayTime.append(row[0])
            if row[4] == '217.15.20.194':
                arrayByte.append(int(row[12]))
                arrayTime.append(row[0])
                countByte += int(row[12])

# output the result
    MB = countByte / 1000000
    resultSum = MB * k
    print('Sum =', '%.2f' % resultSum)

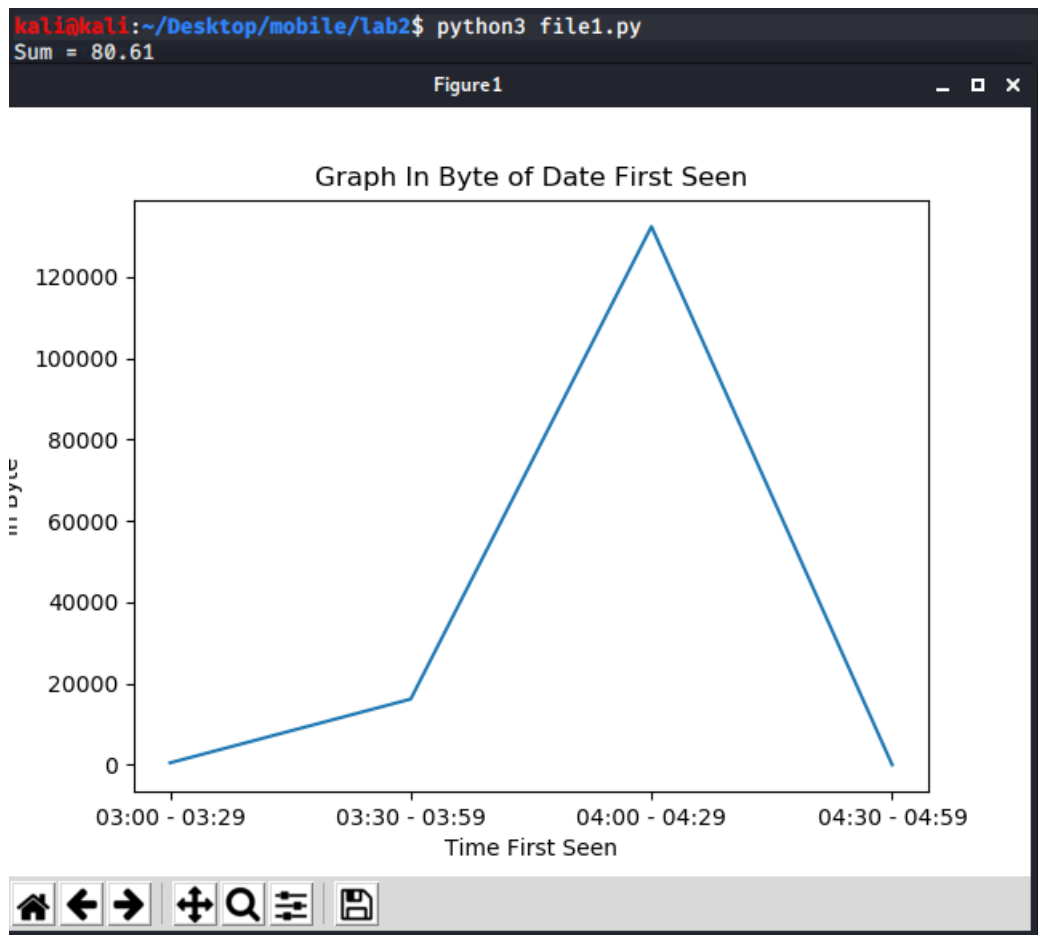
    arrayTime.pop(0)
    arrayByte.pop(0)

# plotting averages graph
    b3_0 = 0
    b3_3 = 0
    b4_0 = 0
    b4_3 = 0
    for i in range(len(arrayTime)):
        if '25 03:0' in arrayTime[i]:
            b3_0 += int(arrayByte[i])
        if '25 03:1' in arrayTime[i]:
            b3_0 += int(arrayByte[i])
        if '25 03:2' in arrayTime[i]:
            b3_0 += int(arrayByte[i])
        if '25 03:3' in arrayTime[i]:
            b3_3 += int(arrayByte[i])
        if '25 03:4' in arrayTime[i]:
            b3_3 += int(arrayByte[i])
        if '25 03:5' in arrayTime[i]:
            b3_3 += int(arrayByte[i])
        if '25 04:0' in arrayTime[i]:
            b4_0 += int(arrayByte[i])
        if '25 04:1' in arrayTime[i]:
            b4_0 += int(arrayByte[i])
        if '25 04:2' in arrayTime[i]:
            b4_0 += int(arrayByte[i])
        if '25 04:3' in arrayTime[i]:
            b4_3 += int(arrayByte[i])
        if '25 04:4' in arrayTime[i]:
            b4_3 += int(arrayByte[i])
        if '25 04:5' in arrayTime[i]:
            b4_3 += int(arrayByte[i])

# average value of In Byte and Date First Seen
    arrByte = [int(b3_0/1079), int(b3_3/1079), int(b4_0/1079), int(b4_3/1079)]
    arrTime = ['03:00 - 03:29', '03:30 - 03:59', '04:00 - 04:29', '04:30 - 04:59']

    plt.plot(arrTime, arrByte)
    plt.title('Graph In Byte of Date First Seen')
    plt.xlabel('Time First Seen')
    plt.ylabel('In Byte')
    plt.show()
```

Рисунок 3. Результат работы программы



Вывод:

- в ходе работы было принято решение построить график на основе средних значений для упрощения визуального понимания графически представленной информации;
- программа реализует заявленные требования и выполняет расчеты согласно заданному абонентскому тарифу и номеру.