МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

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

ФАКУЛЬТЕТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ
Кафедра проектирования безопасности компьютерных систем

Управление мобильными устройствами

Лабораторная работа № 2
по теме «Обработка и тарификация трафика NetFlow»

Работу выполнил
студент группы №3351
очного отделения
Чопик Степан
Вариант 7

Проверил

_____

Федоров И. Р.

Санкт —Петербург 2020

Цель работы: разработать программный модуль для тарификации абонента по типу услуги «Интернет», используя информацию об IP - трафике, полученного с помощью протокола NetFlow.

Задачи:

1. конвертировать файл с данными в любой удобный формат,
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 в соответствии с вариантом задания.

Ход работы:

Условие моего варианта:

Протарифицировать абонента с IP-адресом 87.245.198.147 с коэффициентом k: 2 руб/Мб

Правила тарификации услуг “Интернет”:

X = Q * k, где

X – итоговая стоимость всех звонков абонента,

Q – общий объем трафика NetFlow за отчетный период,

k – множитель тарифного плана (у каждого варианта свой).

В качестве результата работы необходимо представить программный модуль для обработки, просмотра статистики (график) и тарификации трафика NetFlow.

Программный модуль реализован на языке Python 3.7. В данной работе предполагается обработка трафика NetFlow v5 из файла nfcapd.202002251200. Для удобной работы с файлом с исходными данными была применена утилита **nfdump –r nfcapd.202002251200 –o csv > data.csv.** Файл с исходными данными для программы представлены в формате csv, который

является обычным текстовым файлом с определенным и одинаковым разделителем в каждой строке, что упрощает обработку информации.

ts	te	td	sa	da	sp	dp	pr	flg	fwd	stos	ipkt	ibyt
2020-02-25 11:2	2020-02-25 11:2	533.000	192.168.250.3	23.226.231.226	80	3682	TCP	.A..S.	0	0	13	572
2020-02-25 11:2	2020-02-25 11:3	90.270	192.168.250.50	40.114.211.99	61137	443	TCP	....S.	0	0	14	2241
2020-02-25 11:2	2020-02-25 11:3	31.580	192.168.250.3	23.226.231.226	80	28857	TCP	.A..S.	0	0	7	308
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.62	192.168.250.1	58474	123	UDP	.....	0	16	2	152
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.62	123	58474	UDP	.....	0	184	2	152
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.50	192.168.250.1	62595	53	UDP	.....	0	0	2	132
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.50	53	62595	UDP	.....	0	0	2	450
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.50	173.194.73.95	62596	443	UDP	.....	0	0	11	5023
2020-02-25 11:3	2020-02-25 11:3	0.000	173.194.73.95	217.15.20.194	443	62596	UDP	.....	0	0	12	6248
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.50	192.168.250.1	60512	53	UDP	.....	0	0	2	126
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.50	192.168.250.1	56363	53	UDP	.....	0	0	2	112
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.50	53	56363	UDP	.....	0	0	2	616
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.50	53	60512	UDP	.....	0	0	2	502
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.50	108.177.14.94	56364	443	UDP	.....	0	0	6	4313
2020-02-25 11:3	2020-02-25 11:3	0.000	108.177.14.94	217.15.20.194	443	56364	UDP	.....	0	0	4	4185
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61617	53	UDP	.....	0	0	2	156
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61618	53	UDP	.....	0	0	2	156
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61617	UDP	.....	0	0	2	508
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61619	53	UDP	.....	0	0	2	158
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61620	53	UDP	.....	0	0	2	158
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61618	UDP	.....	0	0	2	700
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61621	53	UDP	.....	0	0	2	156
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61622	53	UDP	.....	0	0	2	156
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61620	UDP	.....	0	0	2	768
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61623	53	UDP	.....	0	0	2	176
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61624	53	UDP	.....	0	0	2	176
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61619	UDP	.....	0	0	2	768
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61625	53	UDP	.....	0	0	2	154
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61626	53	UDP	.....	0	0	2	154
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61622	UDP	.....	0	0	2	764
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61627	53	UDP	.....	0	0	2	154
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61628	53	UDP	.....	0	0	2	154
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61621	UDP	.....	0	0	2	764
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61624	UDP	.....	0	0	2	676
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61629	53	UDP	.....	0	0	2	152
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61630	53	UDP	.....	0	0	2	152
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61623	UDP	.....	0	0	2	676
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.1	192.168.250.27	53	61626	UDP	.....	0	0	2	888
2020-02-25 11:3	2020-02-25 11:3	0.000	192.168.250.27	192.168.250.1	61631	53	UDP	.....	0	0	2	156

Рисунок 1 - содержимое файла data.csv

Для вычисления количества трафика находятся все строки, для которых значения поля в столбце sa или da совпадает со значением IP-адреса, указанным в задании варианта, и в словарь data_mas заносятся все значения времени ts как ключи, и сумма всего трафика ibyt в указанный момент времени ts, как значения для данного ключа. Кроме того к общему объему трафика прибавляются все значения ibyt для найденных записей. Найденный объём трафика переводится в мегабайты и умножается на коэффициент тарификации k, указанный в задании варианта. Полученное число является ответом.

```
data = []

with open("traffic.csv", "r") as file:
    for line in file:
        data.append(line.split(","))

# IP-адрес из условия Вра
IP = settings["IP"]
k = settings["COST"]
traffic = 0
trafic_range = []
data_mas = {}
for i in range(len(data)-3):
    line = data[i]
    ts = line[0]
    sa = line[3]
    da = line[4]
    ibyt = line[12]
    if sa == IP or da == IP:
        if not ts in data_mas:
            data_mas[ts] = int(ibyt)
        else:
            data_mas[ts] = data_mas[ts] + int(ibyt)

        traffic += int(ibyt)

statistics(data_mas)

traffic /= 1024*1024

print(round(traffic * k, 2))
```

Рисунок 2 - Исходный код основной части программы

Настройки тарификации хранятся в файле settings.py, в словаре SETTINGS, содержащем ключ “IP” - IP адрес из задания, “COST” - стоимость 1 Мб.

```
SETTINGS = {
    "IP":"87.245.198.147",
    "COST":2
}
```

Рисунок 3 - Содержимое файла settings.py

Было принято решение изобразить график в виде диаграммы и в виде графика рассеивания. Для реализации изображения был использован модуль matplotlib, применяемый для построения диаграмм к научным работам. В построенных графиках по шкале абсцисс указаны временные промежутки, а по шкале ординат указан объём трафика в зависимости от времени.

```python
def statistics(data):
    # функция изобразит график разброса зависимости траффика  от времени
    # и диаграмму зависимости траффика  от времени

    # обновляем параметры полотна
    plot.rcParams.update({"figure.titlesize": 16})
    plot.rcParams.update({"figure.figsize": (16,10)})
    plot.rcParams.update({"axes.labelsize": 16})
    plot.rcParams.update({"axes.titlesize":16})
    plot.rcParams.update({"legend.fontsize":16})
    plot.rcParams.update({"xtick.labelsize":12})
    plot.rcParams.update({"ytick.labelsize":16})

    # коэффициент 1/1024 для указания объема трафика
    coef = 0.000976563

    x = []
    y = []

    # сортируем время по возрастанию
    data_keys = list(data.keys())
    data_keys.sort()
    traffic = 0

    # формируем данные для координатных осей
    for i in data_keys:
        traffic += data[i]
        x.append(datetime.datetime.strptime(i,"%Y-%m-%d %H:%M:%S"))
        y.append(traffic*coef)

    # вызова функции для отрисовки диаграммы
    diagram(x,y,"Time","Traffic (Kb)","Traffic over time")

    dn = dts.date2num(x)

    # создаём фигуру
    plot.figure(figsize=(16,10), dpi= 80, facecolor='w', edgecolor='k')

    # задаём настройки график разброса
    plot.xticks(rotation = 14)
    axis = plot.gca()
    axis.set(xlabel = "Time", ylabel = "Traffic(Kb)")
    fmt = dts.DateFormatter('%Y-%m-%d %H:%M:%S')
    axis.xaxis.set_major_formatter(fmt)
    axis.xaxis.set_major_locator(dts.MinuteLocator(interval=10))
    plot.scatter(dn, y, s=20, c='tab:blue', label="Traffic, Kb")
    plot.title("Traffic over time", fontsize = 27)
    plot.legend(fontsize=16)
    # сохраняем график разброса
    plot.savefig("scatter.png", bbox_inches="tight")
```

Рисунок 4 - Функция отрисовки графиков

```python
def diagram(x_data, y_data, x_label="", y_label="", title=""):
    # создаем холст
    figure, axis = plot.subplots()

    axis.plot(x_data, y_data, lw = 1, color = '#01281A', alpha = 1)

    axis.set_title(title)
    axis.set_xlabel(x_label)
    axis.set_ylabel(y_label)
    # сохраняем диаграмму
    figure.savefig('diagram.png')
```

Рисунок 5 - Функция отрисовки диаграммы

Результатом работы программы является вывод в стандартный поток вывода стоимости всего трафика в рублях, а также два файла diagram.png и scatter.png.

```
su-adm@su-main:~/TrashBox/For steba$ python3 lab2.py
0.03
```

Рисунок 6 - Результат выполнения программы

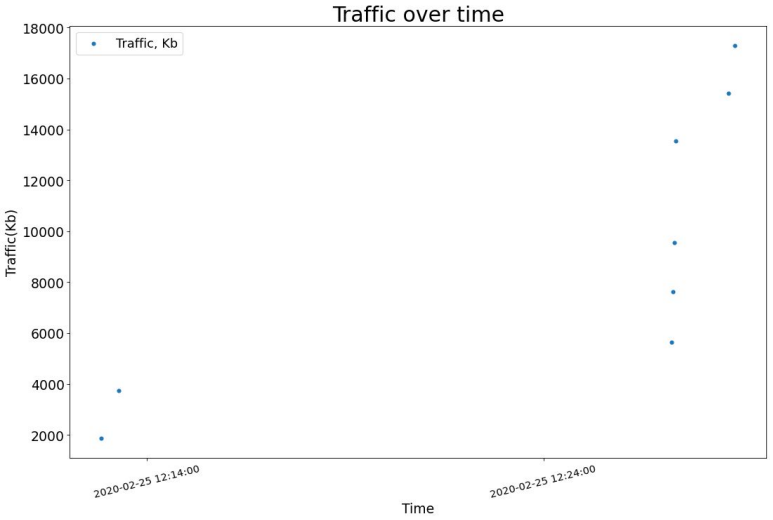

Рисунок 7 - Содержимое файла scatter.png

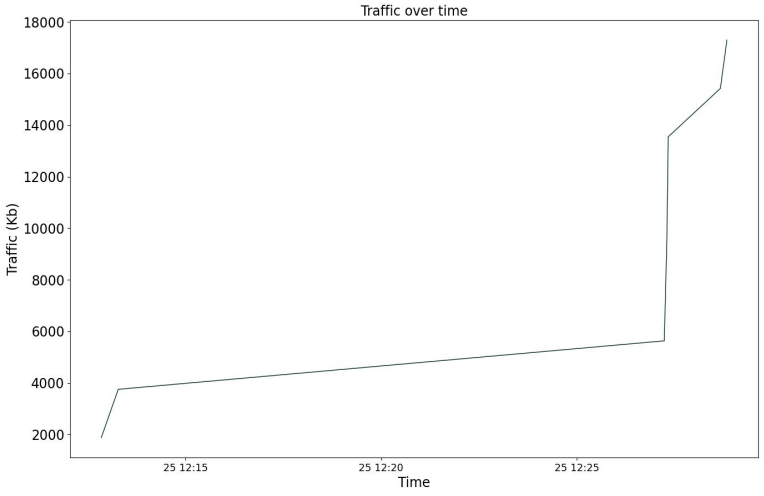

Рисунок 8 - Содержимое файла diagram.png

Вывод.

Из проделанной работы стало очевидно, то для тарификации трафика абонентов необходимо отслеживать объем трафика каждого абонента на сетевом оборудовании, например с помощью протокола NetFlow. Для анализа же полученных данных необходимо программное обеспечение, позволяющее быстро и эффективно сортировать большие объемы данных по адресам назначения и адресам источника пакетов, а так же суммировать данные по каждому пользователю. Для подобных задач, с учетом относительной неизменности стандарта NetFlow, больше подходят компилируемые языки программирования, так как необходима обработка большого количества формализованных и строго типизированных данных. Удобным так же можно назвать хранение тарифа каждого пользователя в отдельном файле, так как это позволяет менять условия использования отдельных абонентов без изменения исходного кода программы.

Механизмы, изученные в данной лабораторной работе, являются основополагающими для предоставления услуг провайдера, и как мне кажется работа поставщика услуг интернета не может обходиться без эффективной, быстродействующей и максимально автоматизированной биллинговой системы.

В ходе работы был изучен способ тарификации трафика на основе информации протокола NetFlow, который является промышленным стандартом, используемым многими разработчиками сетевого оборудования. Полученные знания и умения будут полезны для дальнейшего изучения особенностей сетевого оборудования, бизнес процессов операторов связи и систем управления.