

Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение высшего  
профессионального образования  
«Санкт-Петербургский национальный исследовательский университет информационных  
технологий,  
механики и оптики»

Факультет безопасности информационных технологий

Управление мобильными устройствами

Отчет  
по лабораторной работе №2

Обработка и тарификация Обработка и тарификация трафика NetFlow  
Вариант №6

Студент: Ниткин И.С.  
Группа: N3354

Преподаватель: Федоров И.Р.



Санкт-Петербург  
2020

## Цель работы:

Изучить и программно реализовать правило тарификации для услуг типа “Интернет” по размеру трафика

## Вариант №6:

Протарифицировать абонента с IP-адресом 192.168.250.1 с коэффициентом k: 0,5руб/Мб первые 500Мб, после каждого последующих 500Мб k увеличивается на 0,5руб

## Ход работы:

Работа включает в себя следующие этапы:

1. Преобразование файла nfcapd.202002251200 в читаемый формат
2. Парсинг полученного файла
3. Тарификация выбранных записей.
4. Создание файла данных и построение графика

Для хранения данных мною было принято решение сформировать файл data.csv. Для этого была использована утилита nfdump:

```
>nfdump -r nfcapd.202002251200 -o csv > data.csv
```

В качестве средства реализации мною было выбрано написание программного модуля на языке Java. Достоинством языка Java является динамическая компиляция.

## Исходный код программы:

```
import java.io.File;
import java.io.FileWriter;
import java.io.IOException;
import java.nio.file.Files;
import java.nio.file.Paths;
import java.util.ArrayList;
import java.util.List;

public class Tarificate {

    static String filename = "\\data.csv"; // name of NF file

    public static void main (String [] args) throws IOException {

        String IP;
        if (args.length == 0) IP = "";
        else IP = args[0];

        String path = new File("").getAbsolutePath() + filename;
```

```

List<NFRow> NF = parse (path);

calcilation(NF, new Tarif(IP));
makePlot(NF, IP.length() == 0 ? "192.168.250.1" : IP);
}

private static List<NFRow> parse (String path) throws IOException {
    List<NFRow> cdr = new ArrayList<>();
    List<String> fileLines = Files.readAllLines(Paths.get(path));
    fileLines.remove(0);
    for (int i = 0; i<3; i++) {
        fileLines.remove(fileLines.size()-1);
    }

    for (String fileLine : fileLines) {
        String[] splitedText = fileLine.split(",");

        NFRow current = new NFRow (splitedText);
        cdr.add(current);
    }

    return cdr;
}

private static void calcilation (List<NFRow> NF, Tarif tarif) {
    for (NFRow nfRow : NF) {
        if (nfRow.sa.equals(tarif.IP) || nfRow.da.equals(tarif.IP)) {
            tarif.tb += nfRow.ibyt;
        }
    }
    System.out.println(tarif.countNprint());
}

private static void makePlot (List <NFRow> NF, String IP) throws
IOException {
    List<PlotString> plot = new ArrayList<>();

    int tkb = 0;
    for (NFRow nfRow : NF) {
        if (nfRow.sa.equals(IP) || nfRow.da.equals(IP)) {
            tkb += nfRow.ibyt;
            plot.add(new PlotString(nfRow.te, tkb));
        }
    }

    for (int i = 0; i < plot.size(); i++) {
        int j = 1;

        while (i+j < plot.size() &&
plot.get(i).date.equals(plot.get(i+j).date)) j++;
        j--;
        while (--j >= 0) plot.remove(i+j);
    }

    FileWriter writer = new FileWriter(new File("plot.txt"));
    for (PlotString plotString : plot) writer.write(plotString.date + " "
+ plotString.bytes + "\n");
    writer.flush();
    writer.close();
}

}

class NFRow {

```

```

String ts, te, sa, da;
int ibyt;

NfRow(String [] splitedText) {
    this.ts = splitedText[0];
    this.te = splitedText[1];
    this.sa = splitedText[3];
    this.da = splitedText[4];
    this.ibyt = Integer.parseInt(splitedText[12]);
}
}

class Tarif {
    String IP;

    double k = 0.5;

    int tb;
    int tkb;
    double S;

    Tarif (String IP) {
        if (IP.length() == 0) this.IP = "192.168.250.1";
        else this.IP = IP;
    }

    void count() {
        tkb = (int) Math.round((double)tb/1024);
        int tmp = tkb;

        while (tmp > 0) {
            tmp -= 500;
            int kb = tmp > 0 ? 500 : 500+tmp;
            S = S + kb*k;
            k += 0.5;
        }
    }

    String countNprint() {
        count();
        String printmessage = IP + ":\n" +
            "traffic:\t = " + tkb + " Kb\n" +
            "Cost\t = " + S + " rub.";
        return printmessage;
    }
}

class PlotString {
    String date;
    int bytes;

    PlotString (String date, int bytes) {
        this.date = date;
        this.bytes = bytes;
    }
}

```

Для запуска:

```
> java -jar L2.jar [IP]
```

По умолчанию - запуск тарификации для IP = 192.168.250.1, соответствующего варианту №6

Обработанный файл с данными NetFlow: data.csv

Файл data.csv получен с помощью утилиты nfdump, с помощью команды:

```
>nfdump -r nfcapd.202002251200 -o csv > data.csv
```

Для построения графика используется gnuplot. Данные для построения графика записываются в файл plot.txt в процессе выполнения программы.

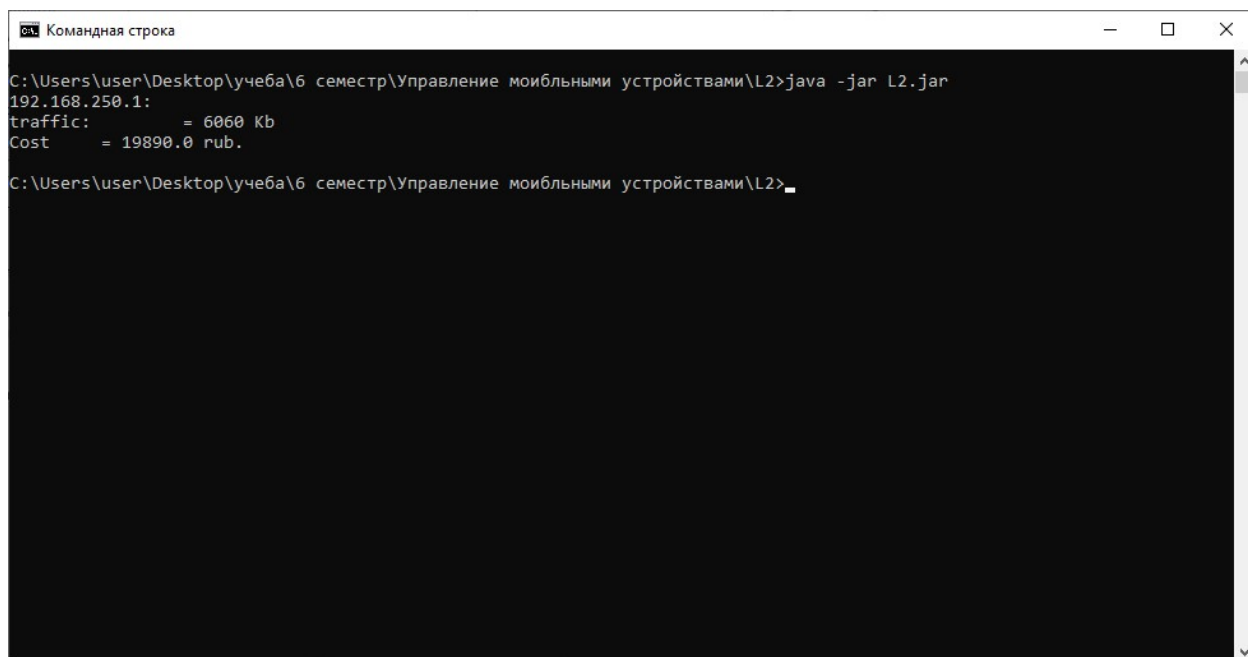
Набор команд для gnuplot для построения графика:

```
gnuplot> set xdata time
gnuplot> set timefmt "%Y-%m-%d %H:%M:%S"
gnuplot> set xtics 600
gnuplot> set ylabel "bytes"
gnuplot> plot 'plot.txt' using 1:3 with lines
```

Тарифные опции (записаны в код программы): первые 500 Кбайт -  $k = 0,5$  руб./Кбайт, далее, каждые 500 Кбайт  $k$  увеличивается на 0,5 руб.

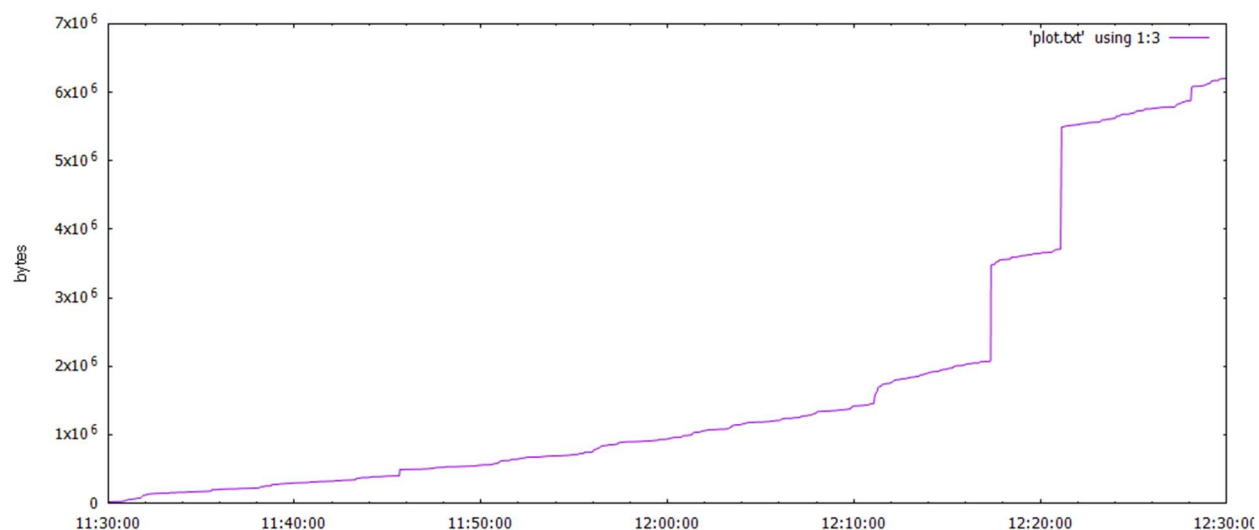
Исходное задание Варианта №6: первые 500 Мбайт -  $k = 0,5$  руб./Мбайт, далее, каждые 500 Мбайт  $k$  увеличивается на 0,5 руб., однако суммарный трафик по IP = 192.168.250.1 составил 5,918 Мбайт, и в соответствии с Примечанием №2 единица учета была изменена на килобайт.

В результате выполнения программы в консоль выводится сообщение с информацией о тарификации.



```
Командная строка
C:\Users\user\Desktop\учеба\6 семестр\Управление моибльными устройствами\L2>java -jar L2.jar
192.168.250.1:
traffic:      = 6060 Kb
Cost         = 19890.0 rub.
C:\Users\user\Desktop\учеба\6 семестр\Управление моибльными устройствами\L2>
```

График построенный с помощью утилиты gnuplot:



### Вывод:

В результате проделанной работы были изучены и программно реализованы на Java правила тарификации для услуг типа “Интернет”, а так же изучены утилиты nfdump и gnuplot