#### МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

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

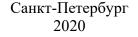
#### ФАКУЛЬТЕТ БЕЗОПАСНОСТИ ИНФОРМАЦИОННЫХ ТЕХНОЛОГИЙ

Управление мобильными устройствами

## Лабораторная работа №2 «Обработка и тарификация трафика NetFlow» Вариант № 3

Работу выполнил: Студент группы N3347 Волкова Е. А.

Проверено: Таранов С. В.





#### 1. Задание

В данной работе предполагается обработка трафика NetFlow v5 из файла nfcapd.202002251200:

- 1. Привести данный файл в читабельный вид
- 2. Сформировать собственный файл для тарификации любого формата, с которым удобно работать
- 3. Построить график зависимости объема трафика от времени
- 4. Протарифицировать абонента с IP-адресом 192.168.250.27 с коэффициентом k: 1руб/Мб

Правила тарификации услуг "Интернет": X = O \* k,

где X - итоговая стоимость, Q - общий объем трафика NetFlow за отчетный период, k - множитель тарифного плана.

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

# 2. Описание выбранных средств реализации и обоснование выбора

Для реализации программного модуля я выбрала С# так как владею им наиболее свободно из всех языков, и считаю самым удобным инструментом для реализации большинства задач.

CSV файл был сформирован с помощью утилиты nfdump

liza@liza-VirtualBox:~/Downloads\$ nfdump -r nfcapd.202002251200 -o csv > nfcapd.20200.csv

### 3. Исходный код

using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

```
using System. Windows. Forms;
using System.IO;
namespace WindowsFormsApp2
{
  public partial class Form1: Form
    public Form1()
       InitializeComponent();
     }
    private void chart2 Click(object sender, EventArgs e)
       var data = ProcessCSV("data.csv");
       string hour;
       string time;
       int count1 = 0;
       int count2 = 0;
       int count3 = 0;
       int count4 = 0;
       int count5 = 0;
       int count6 = 0;
       int count7 = 0;
       int count8 = 0;
       int count9 = 0;
       int count10 = 0;
       int count11 = 0;
       int count12 = 0;
       int count13 = 0;
       int count14 = 0;
       int count 15 = 0;
       int count16 = 0;
       int count17 = 0;
       int count 18 = 0;
       int count 19 = 0;
       int count20 = 0;
```

```
int count21 = 0;
int count22 = 0;
int count23 = 0;
int count24 = 0;
string date = textBox2.Text;
foreach (var ndata in data)
{
  time = ndata.Ts.Substring(0, 2);
  if (time == date)
    hour = ndata.Ts.Substring(11, 2);
    if (hour == "01")
     {
       count1 = count1 + Convert.ToInt32(ndata.Ibyt);
     if (hour == "02")
       count2 = count2 + Convert.ToInt32(ndata.Ibyt);
     if (hour == "03")
       count3 = count3 + Convert.ToInt32(ndata.Ibyt);
     if (hour == "04")
       count4 = count4 + Convert.ToInt32(ndata.Ibyt);
     if (hour == "05")
       count5 = count5 + Convert.ToInt32(ndata.Ibyt);
    if (hour == "06")
```

```
count6 = count6 + Convert.ToInt32(ndata.Ibyt);
if (hour == "07")
  count7 = count7 + Convert.ToInt32(ndata.Ibyt);
if (hour == "08")
  count8 = count8 + Convert.ToInt32(ndata.Ibyt);
if (hour == "09")
  count9 = count9 + Convert.ToInt32(ndata.Ibyt);
if (hour == "10")
  count10 = count10 + Convert.ToInt32(ndata.Ibyt);
if (hour == "11")
  count11 = count11 + Convert.ToInt32(ndata.Ibyt);
if (hour == "12")
  count12 = count12 + Convert.ToInt32(ndata.Ibyt);
if (hour == "13")
  count13 = count13 + Convert.ToInt32(ndata.Ibyt);
```

```
if (hour == "14")
  count14 = count14 + Convert.ToInt32(ndata.Ibyt);
if (hour == "15")
  count15 = count15 + Convert.ToInt32(ndata.Ibyt);
if (hour == "16")
  count16 = count16 + Convert.ToInt32(ndata.Ibyt);
if (hour == "17")
  count17 = count17 + Convert.ToInt32(ndata.Ibyt);
if (hour == "18")
  count18 = count18 + Convert.ToInt32(ndata.Ibyt);
}
if (hour == "19")
  count19 = count19 + Convert.ToInt32(ndata.Ibyt);
}
if (hour == "20")
  count20 = count20 + Convert.ToInt32(ndata.Ibyt);
if (hour == "21")
  count21 = count21 + Convert.ToInt32(ndata.Ibyt);
```

```
if (hour == "22")
              count22 = count22 + Convert.ToInt32(ndata.Ibyt);
            if (hour == "23")
              count23 = count23 + Convert.ToInt32(ndata.Ibyt);
            if (hour == "24")
              count24 = count24 + Convert.ToInt32(ndata.Ibyt);
       float[] a = new float[24] { count1, count2, count3, count4, count5, count6,
count7, count8, count9, count10,
       count11, count12, count13, count14, count15, count16, count17, count18,
count19, count20, count21, count22, count23, count24};
       foreach (int val in a)
         chart2.Series["K6"].Points.Add(val);
    private static List<Data> ProcessCSV(string path)
       return File.ReadAllLines(path)
         .Skip(1)
         .Where(row => row.Length > 0)
         .Select(Data.ParseRow).ToList();
     }
    private void label 1 Click(object sender, EventArgs e)
       var data = ProcessCSV("data.csv");
```

```
float costtI = 0;
  float costtO = 0;
  foreach (var ndata in data)
   {
     int result1 = String.Compare(ndata.Sa, Cost.num);
     int result2 = String.Compare(ndata.Da, Cost.num);
     if (result1 == 0 \parallel \text{result2} == 0)
       float Ibyt = Convert.ToSingle(ndata.Ibyt);
       float Obyt = Convert.ToSingle(ndata.Obyt);
       costtI = (costtI + Ibyt);
       costtO = (costtO + Obyt);
  float costt = (costtI + costtO) / (1024 * 1024) * Cost.origin;
  label1.Text = "цена за услуги \"Интернет\":" + "\n" + costt.ToString();
}
private void label2_Click(object sender, EventArgs e)
}
private void Form1 Load(object sender, EventArgs e)
}
private void label6_Click(object sender, EventArgs e)
}
private void textBox2 TextChanged(object sender, EventArgs e)
{
```

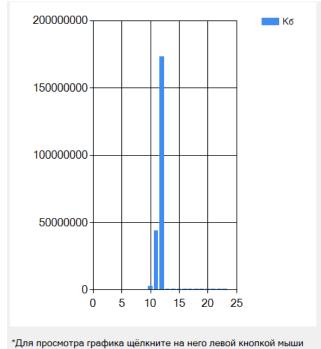
}

```
public class Data
     public string Ts { get; set; }
     public string Te { get; set; }
     public string Td { get; set; }
     public string Sa { get; set; }
     public string Da { get; set; }
     public string Sp { get; set; }
     public string Dp { get; set; }
     public string Pr { get; set; }
     public string Ra flg { get; set; }
     public string Fwd { get; set; }
     public string Stos { get; set; }
     public string Ipkt { get; set; }
     public string Ibyt { get; set; }
     public string Opkt { get; set; }
     public string Obyt { get; set; }
     public string In { get; set; }
     public string Out_ { get; set; }
     public string Sas { get; set; }
     public string Das { get; set; }
     public string Smk { get; set; }
     public string Dmk { get; set; }
     public string Dtos { get; set; }
     public string Dir { get; set; }
     public string Nh { get; set; }
     public string Nhb { get; set; }
     public string Svln { get; set; }
     public string Dvln { get; set; }
     public string Ismc { get; set; }
     public string Odmc { get; set; }
     public string Idmc { get; set; }
     public string Osmc { get; set; }
     public string Mpls1 { get; set; }
     public string Mpls2 { get; set; }
     public string Mpls3 { get; set; }
     public string Mpls4 { get; set; }
     public string Mpls5 { get; set; }
```

```
public string Mpls6 { get; set; }
public string Mpls7 { get; set; }
public string Mpls8 { get; set; }
public string Mpls9 { get; set; }
public string Mpls10 { get; set; }
public string Cl { get; set; }
public string S1 { get; set; }
public string Al { get; set; }
public string Ra { get; set; }
public string Eng { get; set; }
public string Exid { get; set; }
public string Tr { get; set; }
internal static Data ParseRow(string row)
{
  var columns = row.Split(';');
  return new Data()
     Ts = columns[0],
     Te = columns[1],
     Td = columns[2],
     Sa = columns[3],
     Da = columns[4],
     Sp = columns[5],
     Dp = columns[6],
     Pr = columns[7],
     Ra flg = columns[8],
     Fwd = columns[9],
     Stos = columns[10],
     Ipkt = columns[11],
     Ibyt = columns[12],
     Opkt = columns[13],
     Obyt = columns[14],
     In = \text{columns}[15],
     Out_= columns[16],
     Sas = columns[17],
     Das = columns[18],
     Smk = columns[19],
```

```
Dmk = columns[20],
           Dtos = columns[21],
           Dir = columns[22],
           Nh = columns[24],
           Nhb = columns[24],
           Svln = columns[25],
           Dvln = columns[26],
           Ismc = columns[27],
           Odmc = columns[28],
           Idmc = columns[29],
           Osmc = columns[30],
           Mpls1 = columns[31],
           Mpls2 = columns[32],
           Mpls3 = columns[33],
           Mpls4 = columns[34],
           Mpls5 = columns[35],
           Mpls6 = columns[36],
           Mpls7 = columns[37],
           Mpls8 = columns[38],
           Mpls9 = columns[39],
           Mpls10 = columns[40],
           Cl = columns[41],
           S1 = columns[42],
           Al = columns[43],
           Ra = columns[44],
           Eng = columns[45],
           Exid = columns[46],
           Tr = columns[47]
         };
      }
 public class Cost
 {
      internal const float origin = 1;
      internal const string num = "192.168.250.27";
 }
}
```

## 4. Результат работы



Введите число, за которое хотите посмотреть зависимость объёма трафика от времени байт(час)

25

цена за услуги "Интернет": 3,621222

\*Для расчёта тарификации щёлкните на поле