Zadání úlohy Struktura a dynamika sítě

Cíl úlohy:

- Získat schopnost zpracovat data reálné sítě pomocí pcap záchytu.
- Naučit se načítat pcap data.
- Naučit se identifikovat komunity zařízení.
- Naučit se identifikovat časový průběh komunikace.

Zadání:

1. Vstupní data ve formátu PCAP obsahují vedle řady komunikací i pokusné relace mezi dvěma mobilními zařízeními. Jinými slovy, dva zákazníci se se svými mobily pokoušejí navázat několikrát spojení v rámci neznámého protokolu Viber. Jejich komunikace je zachycena na jejich lokálních Wifi routerech (jeden sedí v Praze a druhý ve Vancouveru).
   a) Použijte PCAP soubor 20141021merged.zip
2. Připravte si vstupní data:
   a) poskytnutý pcap záchyt dekompresujte do Vámi vybraného adresáře
   b) převeďte si vstupní data do formátu, které umí zpracovávat Vámi vybrané knihovny (např. scapy, dpkt)
3. Data načtěte do paměti a zjistěte základní charakteristiky komunikační sítě:
   - zjistěte páry komunikujících IP adres,
   - určete komunity IP adres (pomocí Louvain algoritmu). Vytvořte vizualizaci sítě IP adres s uzly obarvenými podle komunit)
   - Identifikujte IP adresy provádějící komunikaci s uzly z jiných komunit. Vytvořte vizualizaci takto zjednodušené sítě.
   - celkový počet cílových IP adres.
4. Zobrazte provoz na síti z hlediska počtu paketů, počtu bytů, počet zahájených relací (SYN pakety) po minutách
   a) Vypište 10 nejčetnějších cílových portů z hlediska počtu paketů, bytů a zahájení relací
   b) Pro významné nejčetnější porty (<32000) zobrazte provoz (kromě SYN) jako průběh hustoty paketů, bytů, zahájených TCP relací.

Doporučení:

1. Doporučujeme použít jazyk Python a knihovny NetworkX a Matplotlib a nástroj Tshark (použitelného v rámci distribuce WireShark pro Linux i MS Windows)
2. K vytvoření a nastavení cesty pro generované diagramy doporučujeme použít funkce

```
def SetOutPN(subFolder):
  #====================
  # Open a necessary output infrastructure
  outPN = subFolder
  if not os.path.exists(outPN):
    os.makedirs(outPN)
  return outPN
```

3. K zobrazeni histogramu doporučujeme vyjít z funkce

```
def Histogram(sPlotIndex, x = None, xLabel = '', yLabel = '', num_bins = 100, log = False,
              normed = 0, colorLabel = None, alpha = None, **kwargs):

#=============================================================================================
  fig = plt.figure()
  fig.set_size_inches(4, 3) # width and height in inches
  ax = plt.subplot(sPlotIndex)
  n, bins, patches = plt.hist(x, num_bins, normed=normed, log = log, facecolor='green',
alpha=0.5)
  if log:
    plt.xscale('log')
    plt.ylim(ymin=0.5)
  if xLabel: plt.xlabel(xLabel)
  if yLabel: plt.ylabel(yLabel)
```

4. V případě, že Vám diagramy z různých vizualizací pomocí knihovny matplotlib neočekávaným způsoben interagují, je možné po uložení diagramu doporučujeme resetovat knihovnu matplotlib pomocí funkcí použitých v následující funkci (Upozornění: výběr funkcí je však potřeba přizpůsobit Vámi používané verzi knihovny).

```
def PltClear():
  #============
  plt.clf()
  fig = plt.gcf()
  plt.close(fig)
  plt.close('all')
```

5. Před zapsáním diagram do souboru doporučujeme použít funkci `plt.tight_layout()`
6. Pro zápis diagram do souboru doporučujeme použít funkci `plt.savefig`