Zadání úlohy Rekonstrukce sítě - črs

Cíl úlohy:

- Získat schopnost zpracovat data reálné sítě pomocí pcap záchytu.
- Naučit se načítat pcap data.
- Naučit se identifikovat role jednotlivých zařízení.
- Naučit se identifikovat komunikační situaci.

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). Vaším úkolem je identifikovat tyto mobily (IP a MAC adresu), zjistit zařízení (IP a MAC adresu), která se chovají jakou routery, a zjistit IP adresy Viber serverů, které zprostředkovávají řídí komunikaci. V rámci postupu tak vytvořte diagram sítě komunikace.
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
3. Data načtěte do paměti a zjistěte základní charakteristiky komunikační sítě:
   a) zjistěte
      - celkový počet komunikujících zařízení,tj. IP a MAC adresy,
      - celkový počet přenesených paketů,
      - celkový počet zdrojových IP adres,
      - celkový počet cílových IP adres.
4. Vytvořte vizualizaci lokálních komunikační sítí mezi MAC a IP adresami. Využijte informací protokolu ARP.
5. Vytvořte vizualizaci komunikační sítě mezi IP adresami, tj. dvě IP adresy komunikují, pokud byl mezi přenesen alespoň jeden paket v rámci TCP či UDP protokolu.
6. Identifikujte Viber servery podle protokolu DNS.

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`

7. Pro konverzi vstupního souboru PCAP na vhodný formát, např. CSV, můžete použít program Tshark. Vhodná pole vyberte podle přiložené tabulky. Zvažte, zda není výhodné použít při exportu techniku pipe, kterou transformovaná data opět kompresujete, např. Programem „gzip“.

Příkazová řádka pro MS Windows v principu může vypadat následovně

"C:\Program Files\Wireshark\tshark.exe" -T fields -e frame.number -e frame.time_relative -e frame.len -e eth.src -e eth.dst -e eth.type -e arp.opcode -e arp.src.hw_mac -e arp.src.proto_ipv4 -e arp.dst.hw_mac -e arp.dst.proto_ipv4 -e ip.src -e ip.dst -e ip.proto -e tcp.srcport -e tcp.dstport -e udp.srcport -e udp.dstport -E header=y -E separator=, -r viber.pcap > viber.csv

8. K načítání zazipovaných dat můžete v Pythonu použít konstrukci série načítacích funkcí
```
with open(fN, "rb") as inBinF:
  inGZF  = TextCodecWrapper(gzip.GzipFile(fileobj = inBinF, mode = 'r'),
                            "utf-8", errors='ignore')
  dataReader = csv.reader(inGZF, delimiter=',', quotechar='"')
```
kde „TextCodecWrapper“ je namapován v Python 2.7 na
```
  from codecs import EncodedFile as TextCodecWrapper
```
a v Python 3.* na
```
  from io import TextIOWrapper as TextCodecWrapper
```
9. Doporučujeme nejprve zrekonstruovat lokální síť pomocí MAC a IP adres, které se komunikují v rámci odpovědí v protokolu ARP. Vytvořte bipartitní graf komunikace, které dvě IP adresy se vázají na komunikující MAC adresy.
10. Komunikační síť na úrovni IP adres zrekonstrujte pomocí paketů, které se týkají IP4 paketů. Access pointy získejte pomocí MAC adres v rámci IP4 paketů. Jestliže se pro danou IP adresu zná MAC adresa, uveďte je do diagramu společně s IP adresou zařízení.
11. Určete gate-way uzly sítě.
12. Určete jména zařízení podle DNS protokolu a určete, které servery poskytují služby Viber.com.