Saish Sali
SBU ID: 111492587
CSE508: Network Security, Fall 2017

Homework 2: Passive Network Monitoring

About mydump:

- A passive network monitoring application written in C using the libpcap packet capture library.
- Captures traffic from a network interface in promiscuous mode. Promiscuous mode allows a network device to intercept and read each network packet that arrives.
- Reads packets from a pcap trace file.
- Prints record for each packet in its standard output.
- Supports following protocol types:
    - **IPv4: TCP, UDP, ICMP**
    - **ARP**
- Prints "OTHER" for all protocol types not mentioned above.
- Supports **BPF filter** for capturing a subset of the traffic.
- Supports a **<string> pattern** for capturing only packets with matching payloads.
- Supports only **Ethernet** link-layer header type.

Application flow:

- Program execution begins at main() which gets command line options specified by the user (interface, file, BPF filter, string expression).
- If both interface and pcap filename is specified as option, program reads from the pcap file.
- Finds a default device (en0 on macOS) on which to capture if both interface and pcap filename is not specified as option.
- Opens the device/file for sniffing in **promiscuous** mode.
- Compiles and applies BPF filter expression specified by the user to filter traffic.
- Starts sniffing with process_packet as a callback function.
- Closes capturing device or savefile.
- **Packet processing**:
    - Define structures (sniff_arp, sniff_ip, sniff_tcp, sniff_udp) to describe packets over Ethernet.
    - Define ethernet header.
    - Check for ether type and then determine protocols.
    - Access packet payload using character pointer.
    - Check for <string> pattern in the payload using strcasestr(). If the pattern is found in the payload, print packet record containing timestamp, source and destination mac addresses, ethertype, packet length, source and destination IP addresses and

ports, protocol type ("TCP", "UDP", "ICMP", "OTHER") and the raw content of the packet payload.

Usage:

```
mydump [-i interface] [-r file] [-s string] expression
```

-i Live capture from the network device <interface> (e.g., eth0, en0). If not specified, it automatically selects a default interface to listen on.

-r Read packets from <file> in tcpdump format.

-s Keep only packets that contain <string> in their payload (after any BPF filter is applied).

<expression> is a BPF filter that specifies which packets will be dumped. If no filter is given, all packets seen on the interface (or contained in the trace) are dumped. Otherwise, only packets matching <expression> are dumped.

Example output:

To create the executable file: **make**

To delete the executable file and all the object files: **make clean**

sudo ./mydump

```
2017-10-13 23:01:00.296528, 8c:85:90:06:c6:87 -> b8:af:67:63:a3:28, type 0x800, length 66,
130.245.168.117.56506 -> 130.245.168.117.15672 TCP

2017-10-13 23:01:00.795739, 8c:85:90:06:c6:87 -> b8:af:67:63:a3:28, type 0x800, length 65,
74.125.22.189.54458 -> 74.125.22.189.443 UDP
0c 9b c7 2f 84 cf d0 6c 26 f9 0e 2b 45 36 fe a7    .../...l&..+E6..
80 b0 d5 5e 89 38 4c                               ...^.8L
```

sudo ./mydump -r hw1.pcap

```
2013-01-12 11:39:26.560044, 44:6d:57:f6:7e:00 -> 01:00:5e:00:00:fc, type 0x800, length 63,
224.0.0.252.63923 -> 224.0.0.252.5355 UDP
90 67 00 00 00 01 00 00 00 00 00 03 54 6f 6d    .g..........Tom
00 00 ff 00 01                                  .....
```

sudo ./mydump -i en0 "src port 443"

```
2017-10-13 23:05:50.854075, b8:af:67:63:a3:28 -> 8c:85:90:06:c6:87, type 0x800, length 112,
172.24.20.99.443 -> 172.24.20.99.56531 TCP
17 03 03 00 29 00 00 00 00 00 00 7f 95 fa 27    ....).........'
c5 0d 9b d7 e2 ba f9 10 82 12 2a a6 36 d3 58 80    ..........*.6.X.
00 b3 50 56 c3 9c c1 90 16 05 72 44 0a 85          ..PV......rD..
```

sudo ./mydump -r hw1.pcap -s iphone udp

```
2013-01-14 13:14:16.730263, 3c:d0:f8:4e:4b:a1 -> ff:ff:ff:ff:ff:ff, type 0x800, length 342,
255.255.255.255.68 -> 255.255.255.255.67 UDP
01 01 06 00 c6 df ab 15 00 01 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 3c d0 f8 4e    ............<..N
4b a1 00 00 00 00 00 00 00 00 00 00 00 00 00 00    K...............
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00    ................
00 00 00 00 00 00 00 00 00 00 00 00 63 82 53 63    ............c.Sc
35 01 03 37 06 01 03 06 0f 77 fc 39 02 05 dc 3d    5..7.....w.9...=
07 01 3c d0 f8 4e 4b a1 32 04 c0 a8 00 0a 33 04    ..<..NK.2.....3.
00 76 a7 00 0c 0e 54 68 6f 6d 61 73 73 2d 69 50    .v....Thomass-iP
68 6f 6e 65 ff 00 00 00 00 00 00 00                hone........
```

References:

- http://www.tcpdump.org/pcap.html
- http://www.tcpdump.org/manpages/pcap-filter.7.html