Develop tcpdump Packet Sniffer with:
Scapy: https://thepacketgeek.com/scapy-sniffing-with-custom-actions-part-1/
JNet: https://javatutorial.net/capture-network-packages-java
Libpcap: http://homes.di.unimi.it/~gfp/SiRe/2002-03/progetti/libpcap-tutorial.html
Berkeley Packet Filter (BPF): https://biot.com/capstats/bpf.html

In this assignment, you will develop a passive network monitoring application written in Java/Python/C using the JNet/scapy/libpcap packet capture library. Your program, called 'mydump', will capture the traffic from a network interface in promiscuous mode and print a record for each packet in its standard output, much like a simplified version of tcpdump. The user should be able to specify a BPF filter for capturing a subset of the traffic, and/or a string pattern for capturing only packets with matching payloads.

Your program should conform to the following specification:

mydump [-i interface] [-s string]

-i Listen on network device <interface> (e.g., eth0). If not specified, mydump should select a default interface to listen on.

-s Keep only packets that contain <string> in their payload. You are not required to implement wildcard or regular expression matching. A simple string matching operation will suffice.

For each packet, mydump outputs a record containing the timestamp, source and destination MAC address, EtherType, packet length, source and destination IP address and port, protocol (TCP, UDP, ICMP, OTHER), and the raw content of the application-layer packet payload. You are free, but not required, to enrich the output with other useful information from the packet headers (e.g., TCP flags, IP/TCP options, ICMP message types, etc.).

What to submit:

A zip with all required source code files, an appropriate README, and a short report (txt file is fine) with a brief description of your implementation and an example output from your program.