Matúš Rusňák ID: 116286

PKS – Assignment 1

In this assignment I had to create and implement parser for .pcap files. Parser also has to be able to identify communications for stated protocols and show them correctly in an output .yaml . Given frames also have to be analyzed by given criteria and shown in the output.

Main task has 4 subtask and those are:

Task 1

Analyze all frames in the order they are stored in a file. For every frame find:

A) Serial number of frame in file

B) Frame length in bytes

C) Frame type (Ethernet II, IEEE 802.3)

D) For IEEE 802.3 with LLC find SAP

E) Source and destination MAC address of frame

Task 2

Analyze Ethernet II packets:

A) Find protocol in ethernet header (ARP, IPv4, IPv6, …)

B) Source and destination IP address

C) For IPv4 also find nested protocol (TCP, UDP, …)

D) For 4th layer (TCP,UDP) find source and destination ports. If one of the ports belongs to the “well known” also find it’s application protocol

Task 3

At the end of task 2 find for IPv4 packets this statistics:

A) List of IP addresses of senders and how many packets did they sent.

B) IP address that has sent the most amount of packets (not concerning to who) and their number

Task 4

Implement a parameter “-p” (protocol) which will be followed by an argument and that is a shortcut to a known protocol.

TCP

If given argument was TCP find all complete communications that contain SYN and FIN on both sides.

Find first incomplete communication that contains either opening or ending a communication. This switch should support protocols : HTTP, HTTPS, TELNET, SSH, FTP- DATA, FTP-CONTROL. Output for packets must meet the requirements from task 1 and 2.

UDP

For protocol TFTP find all packets and show them in communications. Output must meet criteria from task 1 and 2

ICMP

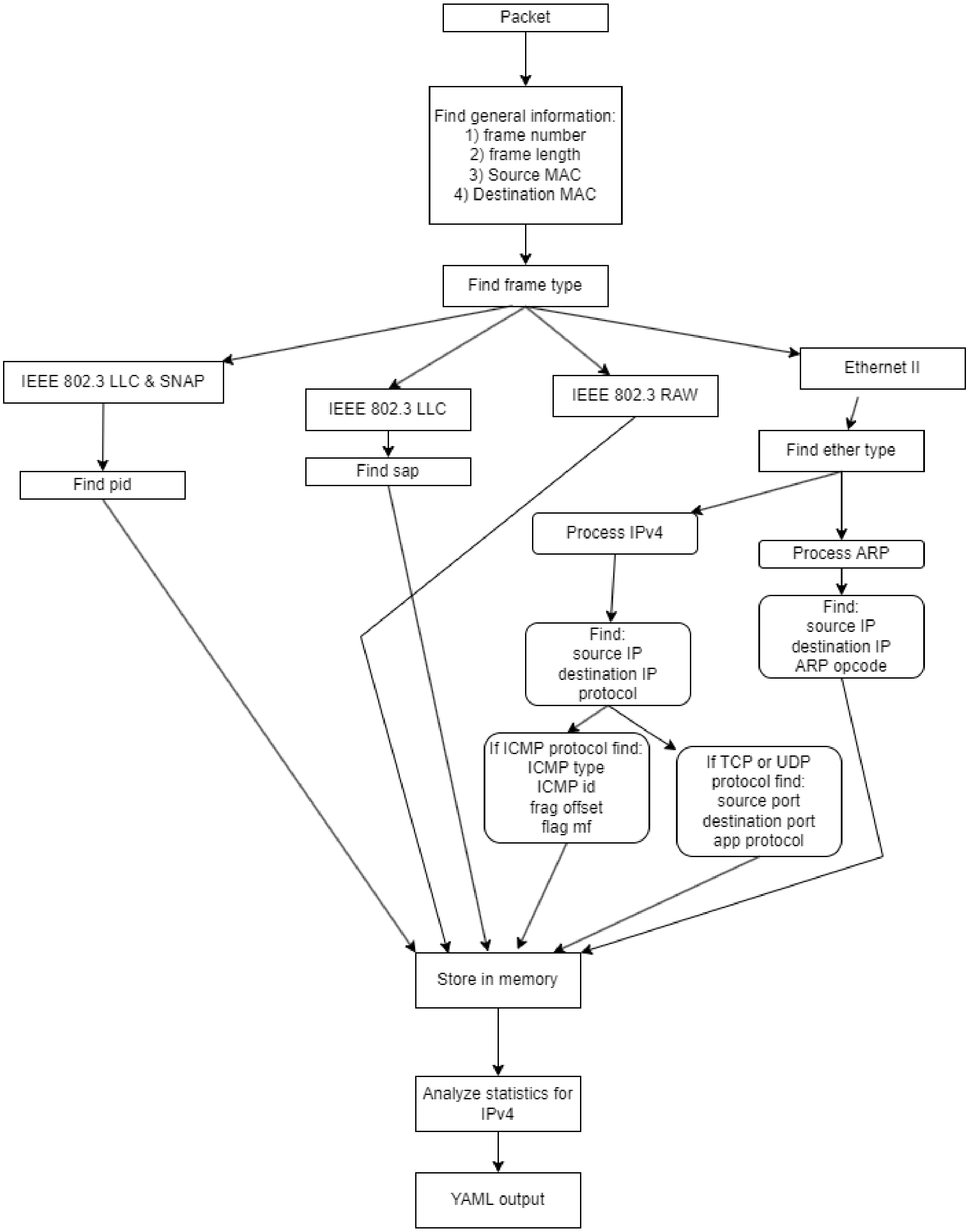
Find ICMP packet pairs of request and reply. If packet was fragmented merge them and show all of them in output. If communication was incomplete output them as incomplete

ARP

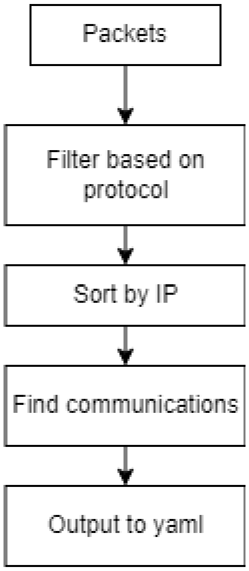
Find all ARP request and reply pair. Find IP address for which we are finding MAC address. If multiple ARP requests were send to the same IP address output them all. If request does not have a reply or reply does not have a request output them as incomplete

Diagrams

Task 1,2 and 3



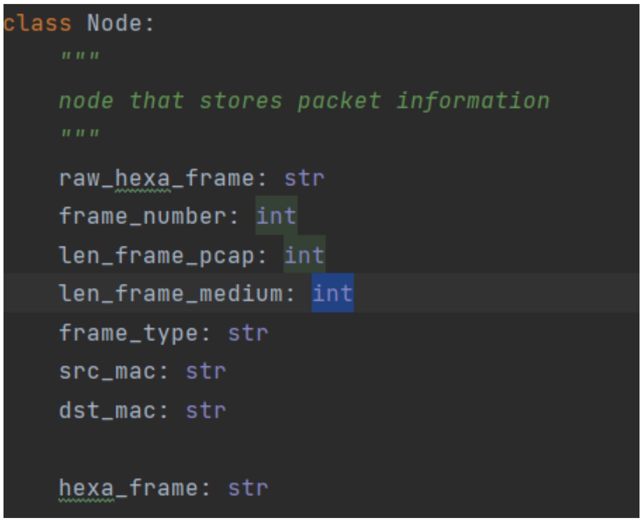
Task 4



Code

All functions, methods and classes contain docstring explaining their functionality. But there are some main code parts that I would like to explain to you

Node

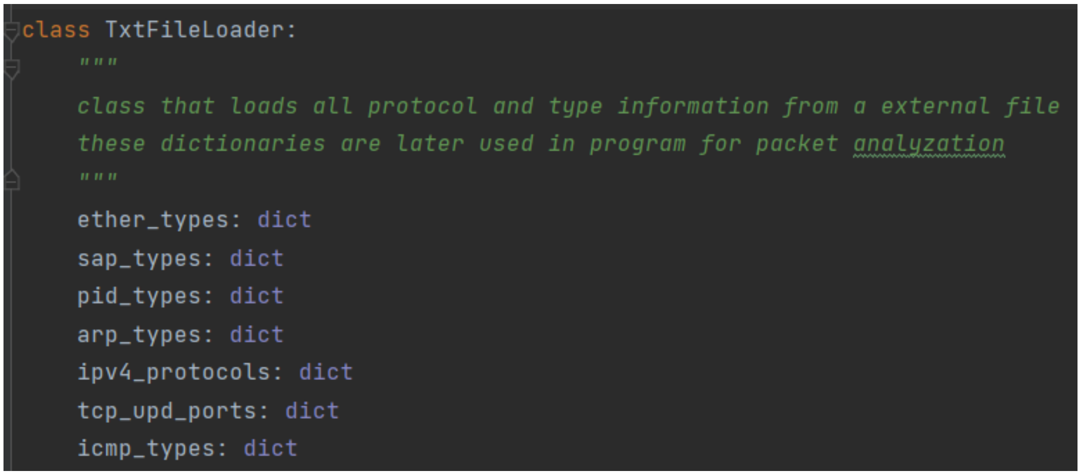


```
class Node:
    """
    node that stores packet information
    """
    raw_hexa_frame: str
    frame_number: int
    len_frame_pcap: int
    len_frame_medium: int
    frame_type: str
    src_mac: str
    dst_mac: str

    hexa_frame: str
```

This class is used to store information about packets. It has methods like return_dict which will return its stored information as a dictionary to output yaml.

File loader



```
class TxtFileLoader:
    """
    class that loads all protocol and type information from a external file
    these dictionaries are later used in program for packet analyzation
    """
    ether_types: dict
    sap_types: dict
    pid_types: dict
    arp_types: dict
    ipv4_protocols: dict
    tcp_upd_ports: dict
    icmp_types: dict
```

This class loads and stores all data from ./protocols/type_data_file.txt. This is later accessed in program in order to find identify wanted information.

Usage

If you want to use this program use the following comand:

./main.py -p {protocol} -f {pcap file}

Parameter -f is not mandatory but if it is given it will choose a pcap file from ./packets/ and analyze them based on given protocol or analyze for all protocol if parameter -p was not given. All out from this program will be located in ./out/ with the name output_{protocol}.yaml.

If parameter -p was not given, program will analyze all packets from a given .pcap file and output them into output_all.yaml.

Validation

If you want to validate generated output, use the following command:

./validator/validator.py -s ./validator/schemas/schema-all.yaml -d ./out/output_{protocol}.pcap

Evaluation

The result of this program is parsing and analyzing given pcap file. It can find communications based on given protocol as well as analyze all packets.

Dependencies

This project was developed in python 3.10. So I recommend you this version for the correct functionality of the project

Modules:

scapy 2.4.5

ruamel.yaml 0.17.21