Počítačové a komunikačné siete

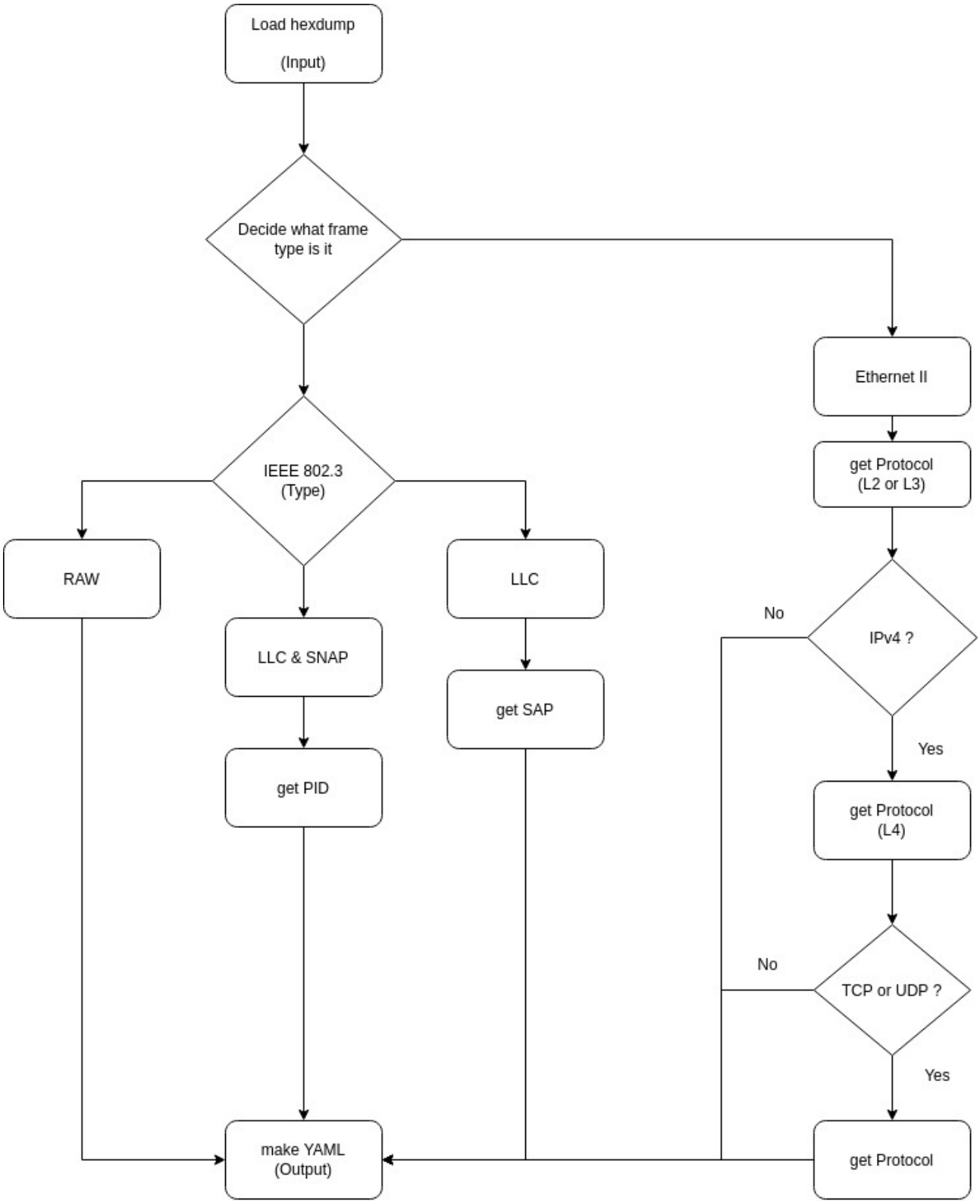
Zadanie č.1 - Analyzátor sieťovej komunikácie

Autor: Marek Čederle
AIS ID: 121193 (xcederlem)
Cvičiaci: Ing. Matej Janeba
(Utorok 16:00)

Úvod

Tento program je implementáciou analyzátora Ethernet siete a sieťovej komunikácie, ktorá je zaznamenaná v .pcap súbore. Program poskytuje základné informácie o daných komunikáciách. Jeho výstupom je súbor vo formáte YAML pre potreby validácie riešenia.

Diagram fungovania programu



Opis fungovania programu

Použité knižnice

Knižnicu scapy som použil na načítanie .pcap súboru (iba hexdump).
Knižnicu ruamel.yaml som použil na generovanie .yaml súboru ako výstupu programu a jeho formátovanie.

Poznámka:
Kedže používam operačný systém Linux a mám staršiu verziu jadra s podporou staršej verzie Pythonu, tak YAML output je usporiadaný podľa abecedy a argument, ktorý by tomu zabraňoval je dostupný iba v novších verziách knižnice, ktoré sú dostupné iba pre novšiu verziu Pythonu. (podľa toho čo som zistil na internete)

Vlastné funkcie

getDataFromFile(option, number)

Funkcia je zodpovedná za vyhľadávanie údajov z externého súboru. Vráti nám string hodnotu podľa zadaného čísla.

get_frame_info(packet, packet_count)

Funkcia získava základné informácie z rámca/paketu, ako je jeho číslo, dĺžka, typ, zdrojová a cieľová adresa MAC a ďalšie. Identifikuje tiež typ rámca a podľa toho pridá daľšie údaje.

getDecimalFrom2BytesAfterOther(data, index1)

Funkcia vráti číslo v desiatkovej sústave, ktoré v hexdump je reprezentované dvomi bytami.

getEtherType(data)

Funkcia zistí decimálnú hodnotu ether typu a následne vráti string hodnotu pre toto číslo.

getPID(data)

Funkcia zistí decimálnu hodnotu PID a vráti zodpovedajúcu string hodnotu.

getDecimalIf2BytesSame(data, index1)

Funkcia zistí či sú dva po sebe idúce bajty rovnaké, ak áno vráti hodnotu tohto bajtu.

getSAP(data)

Funkcia zistí číselnú hodnotu SAP z binárnych údajov a vráti zodpovedajúci názov protokolu.

getFrameType(data)

Funkcia určí typ rámca paketu na základe jeho binárnych údajov a vráti jeho string hodnotu.

getDstMAC(data) && getSrcMAC(data)

Funkcie extrahujú a formátujú cieľové a zdrojové adresy MAC z binárnych údajov paketu.

getIPv4(data, s_or_d)

Funkcia zistí IPv4 adresy z hexdump rámca typu Ethernet II, a vráti ich na základe zadaného parametra s - source, d – destination.

getIP_for_ARP(data, s_or_d)

Funkcia zistí IPv4 adresy z hexdump rámca pre ARP, a vráti ich ako string hodnoty na základe zadaného parametra s - source, d – destination.

getIP_for_ARP_number(data, s_or_d)

Podobne ako predošlá funkcia, vráti nám IPv4 adresy pre ARP ale v číselnej podobe ako jeden integer.

getIPv4Protocol(data)

Funkcia vráti názov protokolu ktorý beží nad IP.

getARP_OPCode(data)

Funkcia určuje ARP opcode (Request alebo Reply) z hexdump ARP packetu.

getTCPorUDP_port(data, s_or_d)

Funkcia vráti čísla portov (zdrojových alebo cieľových) z hexdump paketu TCP alebo UDP na základe zadaného parametra s - source, d – destination.

helloHelp()

Funckia, ktorá vypíše ako pracovať s programom.

main()

Všetko ostatné sa deje práve v main funkcii, ktorá bude opísaná nižšie.

Opis fungovania programu

Overenie platnosti argumentov:
Najskôr program skontroluje argumenty príkazového riadka a zabezpečí, aby spĺňali očakávaný formát. Ak argumenty chýbajú alebo sú nesprávne, program vypíše chybu.

Čítanie PCAP súboru:
Program sa pokúsi prečítať zadaný PCAP súbor využitím knižnice scapy a iteruje cez všetky pakety.

Analýza paketov:
Pre každý paket program zistí podrobnosti vrátane čísla rámca, dĺžky, typu rámca, adries MAC a špecifických informácií pre daný packet na základe jeho typu.

Filtrovanie:
Ak je pomocou prepínača -p zadaný protokol, program vyfiltruje packety iba pre daný protokol. Zmení sa aj trocha formát výstupu v YAML.

Generovanie výstupov:
Program štruktúruje získané údaje do dictionary a vypisuje ich vo formáte YAML. Výstupný súbor má názov output.yaml.

Spracovanie výnimiek:
Ak program nenájde zadaný .pcap súbor tak vyhodí chybu.

Štruktúra externých súborov

Súbor pre ether types, pid, sap, …:

```
ether_types:
  267: "PVSTP+"
  512: "Xerox PUP"
  513: "PUP Addr Trans"
  2048: "IPv4"
…
pid:
  267: "PVSTP+"
  512: "Xerox PUP"
  513: "PUP Addr Trans"
  2048: "IPv4"
…
saps:
  0: "Null SAP"
  2: "LLC Sublayer Management / Individual"
  3: "LLC Sublayer Management / Group"
  6: "IP (DoD Internet Protocol)"
…
tcp_protocols:
  7: "echo"
  19: "chargen"
  20: "ftp-data"
  21: "ftp-control"
…
udp_protocols:
  37: "time"
  53: "dns"
  67: "dhcp"
  68: "dhcp"
…
ip_protocols:
  1: "icmp"
  2: "igmp"
  6: "tcp"
  9: "igrp"
…
icmp_codes:
  0: "Echo Reply"
  3: "Destination Unreachable"
  4: "Source Quench"
  5: "Redirect"
…
```

Súbor protokolov pre úlohu č.4:

http
https
telnet
ssh
ftp-control
ftp-data
tftp
icmp
arp

Používateľské rozhranie

Výstup programu je v súbore output.yaml
Konzolový výstup je dostupný iba pri prepínači -h (help):

```
Usage: ./main.py <pcap_file> <-switch> <protocol>
Example: ./main.py eth-1.pcap -p arp
Supported switches: -p
Supported protocols: http, https, telnet, ssh, ftp-control, ftp-data,
tftp, icmp, arp
```

Voľba implementačného prostredia

Zvolil som si IDE (integrované vývojové prostredie) PyCharm z dôvodu oblúbenosti vývojových prosterdí od firmy JetBrains. Má pokročilé funkcie najmä upozornenia na syntaktické ale aj iné chyby v kóde. Za programovací jazyk som si zvolil Python z dôvodu zjednodušeného syntaxu oproti C/C++ a kvôli tomu že som sa chcel naučiť Python, ktorý sa mi bude hodiť aj na iné predmety a zadania.

Zhodnotenie

Tento program predstavuje implementáciu analyzátora sieťovej komunikácie, schopného spracovať PCAP súbory so zachytenou sieťovou komunikáciou. Jeho úlohou je poskytnúť základné informácie o tejto komunikácii, a jeho výstupom je súbor vo formáte YAML, ktorý môže byť následne použitý pre validáciu riešení.

Program je navrhnutý na základe knižníc Scapy pre čítanie PCAP súborov a ruamel.yaml pre generovanie výstupu vo formáte YAML.

Jednotlivé funkcie programu sú zodpovedné za extrakciu a analýzu informácií z paketov v rámci PCAP súboru. Program tiež umožňuje filtrovať pakety na základe špecifikovaného protokolu, čo zjednodušuje analýzu v prípade záujmu o konkrétny protokol.

Program by sa dal rozšíriť o nedokončené požiadavky, prípadne by sa dalo pomocou viacej prepínačov určiť aký presný output požadujeme od programu.

Zdroje

Prednášky z predmetu PKS
https://en.wikipedia.org/wiki/Address_Resolution_Protocol
https://en.wikipedia.org/wiki/Subnetwork_Access_Protocol#Use
https://github.com/fiit-ba/pks-course/tree/main/202324
Linky z github repozitára predmetu na stránky, ktoré boli spomenuté v zadaní