Varianta ZETA: Sniffer paketů

Dmytro Sadovskyi (xsadov06)

24.04.2022

Obsah:

2 Uvod do problematiky

Mým úkolem bylo navrhnout a implementovat síťový analyzátor v C/C++/C#, který bude schopný na určitém síťovém rozhraním zachytávat a filtrovat pakety. Také bylo nutné vytvořit k projektu dokumentaci. Program podporuje zachytávání paketů na protokolu TPC, UDP, ICMP a ARP. Déle podporuje filtrování vypisovaných paketů podle protokolu, portu a rozhraní.

Volání programu:

./ipk-sniffer [-i rozhraní | --interface rozhraní] {-p --port} {[--tcp|-t] [--udp|-u] [--arp] [--icmp] } {-n num}

1. -i rozhraní, na kterém se bude poslouchat.
2. -p bude filtrování paketů na daném rozhraní podle portu.
3. -t | --tcp zobrazovaní pouze TCP paketů
4. -u | --udp zobrazovaní pouze UDP paketů
5. –icmp zobrazovaní pouze ICMPv4 a ICMPv6 paketů
6. --arp zobrazovaní pouze ARP rámců
7. -n 10 (určuje počet paketů, které se mají zobrazit, tj. i "dobu" běhu programu; pokud není uvedeno, uvažujte zobrazení pouze jednoho paketu, tedy jakoby -n 1)

argumenty mohou být v libovolném pořadí

3 Implementace

Vybraný jazyk pro implementaci tohoto projektu je jazyk C++. Projekt
byl implementován pomocí objektově-orientovaného programování

3.1 Popis tříd.

K vyřešení problému byly vytvořeny dvě třídy. První – Arguments,

Který je vytvořen pro ukládání argumentů programu. Třída přijímá argumenty pomocí metody getargs(). Připadně vypiše seznam aktivních rozhraní. Druhá a hlavní třída – Interface. Na začátku se metodou open vybere požadované rozhraní a pomocí metody get_filter se vytvoří filtr. Metoda sniff_packets volá pcap_loop (spustí smyčku, která skončí, když najde daný počet packetů). Všechny nalezené balíčky jsou zpracovány funkcí get_packet a zobrazeny pomocí několika pomocných funkcí

3.2 Použité knihovny

<pcap.h -> knihovna sloužící k vytváření programů pro analýzu síťových dat - základ celého implementace.

<netinet> - Internet Protocol family

<getopt> - Knihovna pro zpracování argumentů

4 Testování programu

Program jsem testoval metodou porovnávání výstupu s ověřeným zdrojem. Pro generování testovacích paketů jsem využil příkaz curl případně curl -6 pro IPv6 verzi. Jako ověřený zdroj považuji program wireshark. Zde jsem si nastavil stejné filtrování jako v mém programu a následně mohl porovnat výstupy obou programů.

5 Pouzite materialy

[1] RFC3339- Transmission Control Protocol,
https://datatracker.ietf.org/doc/html/rfc3339

[2] RFC 792 - Internet Control Message Protocol,
https://datatracker.ietf.org/doc/html/rfc792

[3] RFC 4443 - ICMPv6,

https://datatracker.ietf.org/doc/html/rfc4443

[4] RFC 826 – ARP,

https://datatracker.ietf.org/doc/html/rfc826

[5] The home web site of library libpcap,

https://www.tcpdump.org/index.html

[6] Small libpcap tutorial,

http://yuba.stanford.edu/~casado/pcap/