Vysoké učení technické v Brně
Fakulta informačních technologií

Projektová dokumentace

Aplikace pro získání statistik o síťovém provozu

4. listopadu 2024 Vladimir Azarov

Obsah

1 Úvod

V této dokumentaci je popsána aplikace *isa-top*, která slouží k zobrazení aktuálních přenosových rychlostí pro jednotlivé komunikující IP adresy. Cílem projektu je vytvořit nástroj pro monitorování síťového provozu a poskytovat uživateli přehledné statistiky v reálném čase.

1.1 Motivace

Motivací pro vytvoření tohoto nástroje je potřeba efektivně monitorovat síťový provoz, identifikovat potenciální problémy a optimalizovat síťovou komunikaci.

2 Přehled nastudovaných informací z literatury

Pro implementaci aplikace byly prostudovány následující technologie, knihovny a koncepty:

- **libpcap**: Knihovna pro zachytávání síťového provozu [1].
- **ncurses**: Knihovna pro práci s terminálem a vytváření uživatelského rozhraní v konzoli [2].
- **Síťové protokoly**: TCP/IP, UDP, ICMP a jejich vlastnosti pro správné rozpoznání a zpracování paketů [3].
- **Sockety**: Práce se síťovými sockety v jazyce C/C++ [4].
- **Vícevláknové programování**: Využití vláken v C++ pro paralelní zpracování [5].
- **Synchronizace vláken**: Použití mutexů a dalších synchronizačních primitiv pro bezpečný přístup ke sdíleným datům [5].

3 Návrh aplikace

Aplikace je navržena jako konzolová aplikace, která po spuštění začne zachytávat síťový provoz na zvoleném síťovém rozhraní a vypisuje statistiky přenosových rychlostí.

3.1 Architektonický návrh

Jak je vidět na obrázku 1, architektura aplikace je rozdělena do několika hlavních komponent.

- **PacketCapture**: Tato třída zajišťuje zachytávání paketů ze zvoleného síťového rozhraní pomocí knihovny libpcap.
- **ConnectionsTable**: Tato třída slouží k uchování a správě jednotlivých spojení a jejich statistik.
- **Display**: Třída pro zobrazení aktuálních statistik v terminálu pomocí knihovny ncurses.
- **Connection**: Třída obsahuje informace o síťovém spojení, jako jsou IP adresa, počet odeslaných/přijatých bajtů a paketů, atd.
- **ConnectionID**: Identifikuje spojení pomocí atributů, jako jsou zdrojový a cílový port a IP adresy, a poskytuje metody pro získávání těchto hodnot.
- **CommandLineInterface**: Třída pro zpracování vstupních parametrů aplikace a validaci argumentů.
- **isa-top**: Hlavní soubor zajišťující inicializaci objektů a správu vláken.

3.2 Přehled datových struktur

Aplikace používá dvě hlavní hašovací tabulky:

- **connectionsTableBefore**: Tabulka uchovávající stav spojení před jednou sekundou, slouží k porovnávání pro výpočet rychlostí přenosu.
- **connectionsTable**: Tabulka uchovávající aktuální stav všech aktivních spojení s informacemi o IP adresách, portech, protokolech a statistikách přenosu.

3.3 Objektově orientovaný přístup

Aplikace je implementována v jazyce C++ s využitím objektově orientovaného programování. Každý modul je implementován jako třída s jasně definovanými metodami a atributy.

3.4 Vícevláknové zpracování

Aplikace využívá dvě vlákna:

- **Vlákno pro zachytávání paketů**: Spouští *PacketCapture*, který zachytává pakety a předává je ke zpracování.
- **Vlákno pro zobrazení statistik**: *Display* pravidelně aktualizuje zobrazení statistik v terminálu.

Pro synchronizaci přístupu ke sdíleným zdrojům je použit `mutex`.

4 Základní informace o programu a návod na použití

4.1 Spuštění programu

Nejprve je potřeba program přeložit pomocí příkazu `make`. Program musí být spuštěn s root oprávněními.

Program se spouští z příkazové řádky s následujícími argumenty:

```
sudo ./isa-top -i <interface> [-s <sort_by>] [-l]
```

- **-i** `<interface>`: Síťové rozhraní, na kterém má probíhat zachytávání paketů.
- **-s** `<sort_by>`: Kritérium pro řazení záznamů (podle počtu přenesených bajtů nebo paketů).
- **-l**: Zapnutí logování, které bude uloženo do souboru `log.csv` v aktuálním adresáři.

5 Testování

Pro zajištění správné funkčnosti aplikace byly implementovány jednotkové a integrační testy pomocí frameworku **Google Test**. Testy jsou organizovány do dvou hlavních adresářů:

- `test/unit.cpp`: Obsahuje jednotkové testy jednotlivých komponent.
- `test/int.cpp`: Obsahuje integrační testy ověřující spolupráci mezi komponentami.

5.1 Generování a spouštění testů

Testy se generují a kompilují pomocí příkazu `make`.

```
make unit_tests
make integration_tests
./unit_tests
./integration_tests
```

5.2 Systémové testy

Systémové testy byly provedeny s cílem ověřit chování aplikace v reálných podmínkách. K tomuto účelu byla implementována nadstavba pro uchovávání logů a jejich porovnání s reálným síťovým provozem.

5.2.1 Postup systémového testování

1. Spuštění nástroje `isa-top` s volitelným přepínačem pro uchování logů do souboru.
2. Generování umělého provozu s definovanými IP adresami, porty a obsahem.
3. Porovnání počtu zachycených paketů a bajtů aplikací `isa-top` se skutečným provozem.

Testování bylo provedeno pro všechny protokoly (ICMP, UDP, TCP) a s použitím různých adresních rodin (IPv4, IPv6).

Výsledky testů odpovídají těmto údajům:

```
timestamp,protocol,src_ip,src_port,dst_ip,dst_port,bytes_sent,bytes_received
    ,packets_sent,packets_received
```

5.2.2 Testování na jednom zařízení a použití loopback

```
sudo ./isa-top -i lo -l
```

- **Test pro ICMP**
  Příkaz: `ping -c 10 127.0.0.1`
  Výsledek:

  ```
  1730308612,ICMP,127.0.0.1,0,127.0.0.1,0,1960,1960,20,20
  ```

- **Test pro ICMPv6**
  Příkaz: `ping -c 10 ::1`
  Výsledek:

  ```
  1730308745,ICMPv6,::1,0,::1,0,2360,2360,20,20
  ```

- **Test pro UDP a IPv4**
  Příkaz: `for i in {1..10}; do echo "Test UDP IPv4 packet $i"| nc -u -w1 127.0.0.1 228; done`
  Výsledek:

  ```
  1730309199,UDP,127.0.0.1,42087,127.0.0.1,228,66,66,1,1
  1730309199,UDP,127.0.0.1,45373,127.0.0.1,228,65,65,1,1
  ...
  ```

- **Test pro UDP a IPv6**
  Příkaz: `for i in {1..10}; do echo "Test UDP IPv6 packet $i"| nc -u -6 -w1 ::1 228; done`

  Výsledek:

```
1730309283,UDP,::1,44791,::1,228,86,86,1,1
1730309283,UDP,::1,49001,::1,228,85,85,1,1
...
```

- **Test pro TCP a IPv4**
  Příkaz: `for i in {1..10}; do echo "Test TCP IPv4 packet $i"| nc -w1 127.0.0.1 228; done`
  Výsledek:

```
1730309351,TCP,127.0.0.1,50942,127.0.0.1,228,74,74,1,1
1730309351,TCP,127.0.0.1,50922,127.0.0.1,228,74,74,1,1
...
```

- **Test pro TCP a IPv6**
  Příkaz: `for i in {1..10}; do echo "Test TCP IPv6 packet $i"| nc -6 -w1 ::1 228; done`
  Výsledek:

```
1730309456,TCP,::1,38706,::1,228,94,94,1,1
1730309456,TCP,::1,38684,::1,228,94,94,1,1
...
```

5.2.3 Testování na dvou zařízeních a použití LAN

Setup: Testování proběhlo mezi dvěma laptopy připojenými ke stejné lokální síti. Jeden s OS Windows 11 a má IPv4 adresu `192.168.0.192` a IPv6 adresu `fe80::f3db:5049:3fb6:479b` jako odesílatel (generuje provoz) a druhý s Arch Linuxem jako příjemce a má IPv4 adresu `192.168.0.143` a IPv6 adresu `fe80::7d14:6f98:87cf:7ec5`, kde běží aplikace `isa-top`.

- **Test pro ICMP**
  Příkaz: `ping -n 10 192.168.0.143`
  Výsledek:

```
1730325068,ICMP,192.168.0.143,0,192.168.0.192,0,740,0,10,0
1730325068,ICMP,192.168.0.192,0,192.168.0.143,0,0,740,0,10
```

- **Test pro ICMPv6**
  Příkaz: `ping -n 10 -6 fe80::7d14:6f98:87cf:7ec5%30`
  Výsledek:

```
1730325360,ICMPv6,fe80::f3db:5049:3fb6:479b,0,fe80::7d14:6f98:87cf:7
   ec5,0,0,1112,0,12
```

- **Test pro UDP a IPv4**
  Příkaz:

```
for ($i=1; $i -le 10; $i++)  echo "Test UDP IPv4 packet $i"
| ncat -u 192.168.0.143 228
```

  Výsledek:

```
1730326848,UDP,192.168.0.192,50319,192.168.0.143,228,0,66,0,1
1730326848,UDP,192.168.0.192,50318,192.168.0.143,228,0,66,0,1
...
```

- **Test pro TCP a IPv4**
  Příkaz:
  `for ($i=1; $i -le 10; $i++)  echo "Test TCP IPv4 packet $i"| ncat 192.168.0.143 228`
  Výsledek:

```
1730327281,TCP,192.168.0.192,52898,192.168.0.143,228,0,306,0,5
...
```

- **Test pro TCP a IPv6**
  Příkaz:
  `for ($i=1; $i -le 10; $i++)  echo "Test TCP IPv6 packet $i"| ncat -6 [fe80::7d14:6f98:87cf:7ec5%30] 228`
  Výsledek:

```
1730327353,TCP,fe80::f3db:5049:3fb6:479b,52930,fe80::7d14:6f98:87cf
   :7ec5,228,0,407,0,5
...
```

5.2.4 Testování s vysokou zátěží

Test s vysokou zátěží byl proveden pomocí následujících příkazů:

```
iperf -c 127.0.0.1 -u -b 1000M -t 60 -i 1
iperf -c 127.0.0.1 -t 60 -i 1 -P 10
iperf -s -B 127.0.0.1
sudo ping -f 127.0.0.1
```

Aplikace `isa-top` dokázala zpracovat a správně zobrazit veškerý provoz i při vysoké zátěži.

6 Závěr

Aplikace isa-top byla úspěšně navržena a implementována jako efektivní nástroj pro monitorování síťového provozu v reálném čase. Splňuje všechny stanovené cíle a prokázala svou schopnost správně zachytávat a zpracovávat pakety různých síťových protokolů a adresních rodin.

Odkazy

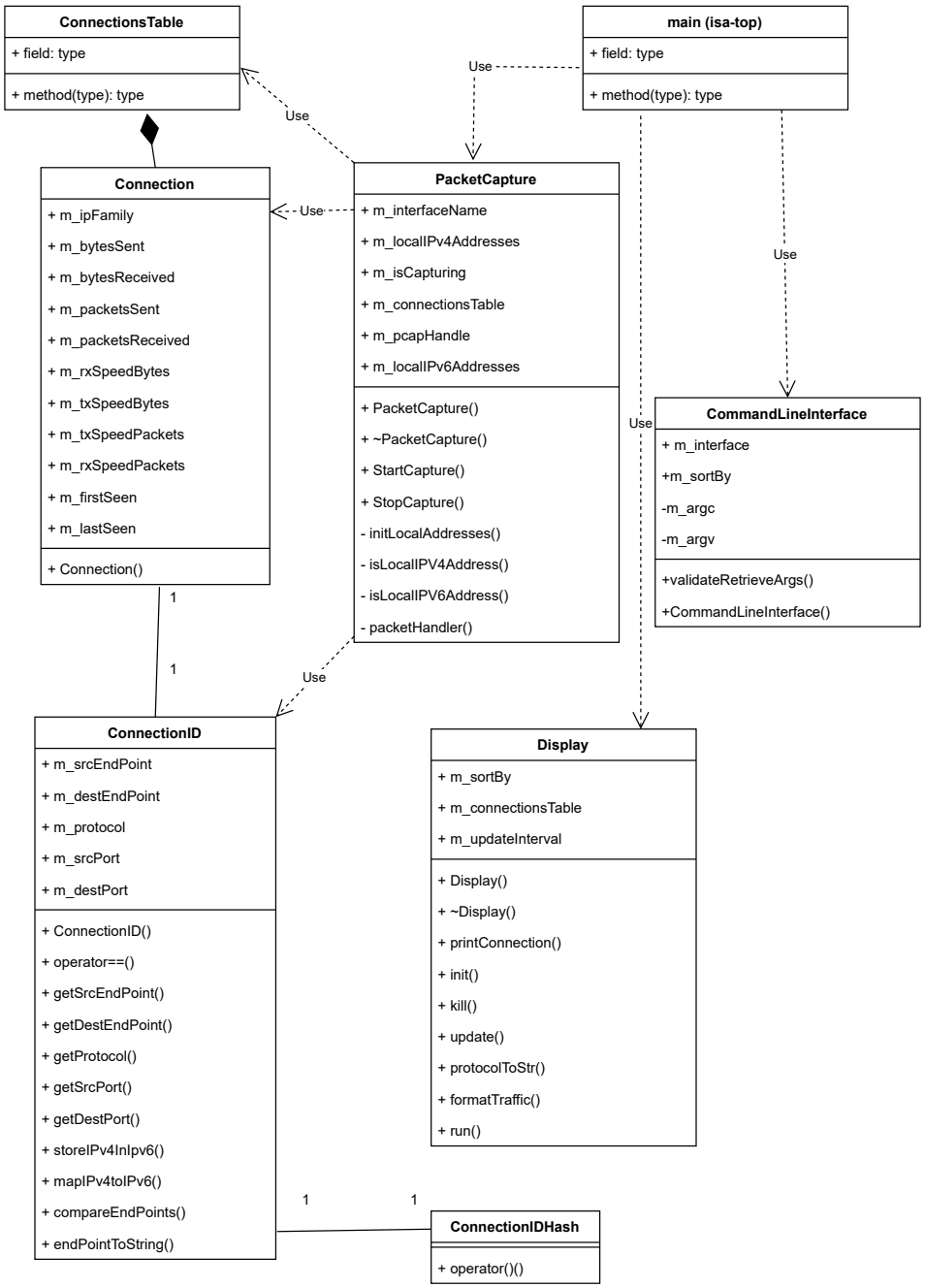
[1] Van Jacobson, Craig Leres a Steven McCanne. *libpcap Library Documentation*. K dispozici na: `https://www.tcpdump.org/manpages/pcap.3pcap.html`. Network Research Group, Lawrence Berkeley Laboratory, 2023.

[2] Juergen Strang. *NCurses Programming Guide*. Network Theory Ltd, 2010. ISBN: 978-0-9541617-4-4.

[3] Douglas E. Comer. *Internetworking with TCP/IP Volume One*. 6th. Pearson Education, 2013. ISBN: 978-0136085300.

[4] Brian “Beej” Hall. *Beej's Guide to Network Programming*. Dostupné na: `http://beej.us/guide/bgnet/`. 2023.

[5] Anthony Williams. *C++ Concurrency in Action*. 2nd. Manning Publications, 2019. ISBN: 978-1617294693.



Obrázek 1: Architektura isa-top