Vysoké učení technické v Brně

Fakulta informačních technologií

2. Projekt do předmětu počítačové komunikace sítě

Varianta 1: Bandwidth Measurement (Ryšavý)

Autor: František Horázný

Třída: 2BIT

Školní rok: 2017/2018

Login: xhoraz02

Úvod	3
1 Struktura kódu	4
1.1 Argumenty	4
1.2 Navázání kontatku mezi metrem a reflektorem	4
2 Problémy při řešení	5
3 Příklad spuštění	5
4 Zdroje	

Úvod

V prvním projektu máme měřit rychlost (pásmo) linky pomocí UDP packetů s nespolehlivým přenosem dat. Nejprve jsem přečetl několik článků a manuálů jak pracovat s UDP a posléze jsem začal programovat a zjišťovat úskalí daného problému.

1 Struktura kódu

Nejdříve jsem si rozdělil celý kód podle počtu argumentů, neboť je pevně daný, bylo to velmi jednoduché řešení.

Druhý krok byl, že jsem zpracoval argumenty pomocí *getopt()*.

Poté jsem navázal spojení reflektoru s metrem.

A zakončil jsem přidáním pár drobností, které jsem vymýšlel dlouhé noci a dny a diskutoval na různých diskuzích.

1.1 argumenty

Díky velmi pěknému zadání nebylo třeba vymýšlet žádné tragické algoritmy a jednoduše jsem použil funkci *getopt()*. Do hloubky jsem špatné zadání parametrů neřešil, neboť jsem přepokládal při spouštění obeznámeného uživatele, ale i při špatně zadaných argumentech by program neměl mít problém vše zajistit a upozornit na chyby.

1.2 Navázání kontaktu mezi metrem a reflektorem

Při využívání UDP protokolu je navázání o něco jednodušší a zřetelnější. Na straně reflektoru (server) je funkce sock() a bind() na straně metru je pouze sock().

Poté už lze pomocí funkcí sendto() a recvfrom() nebo recv() posílat zprávy, které potřebujeme.

2 Problémy při řešení

Způsobů měření je několik. Já jsem vybral přehlcení linky a snažení se poslat co nejvíce packetů (kolik mi procesor dovolí). Tento způsob zapříčiní nižší rychlost u malých packetů. Problém však nastává jak měřit rychlost. Pokud by šlo pouze o průměrnou bez RTT, tak by stačilo na reflektoru spočítat kolik mu přišlo paketů a poslat zpět při detekci, že už nechodí další (například timeoutem, kterým to dělám). Jenže pokud máme počítat i rychlosti během měření a RTT musel jsem modifikovat způsob.

Nejlepší způsob by byl v druhém vláknu přijímat všechny packety a na reflektoru opravdu jen přeposílat zpět. Takový způsob by znamenal orpavdu jednoduchou implementaci reflektoru a vše implementovat v metru. Ale neboť neumím s vlákny, rozhodl jsem se to udělat jinak. Rozdělím si měření na několik kratších a při každém tomto měření pošlu nejdříve paket, u kterého změřím dobu, jakou cestuje (tu si zapíšu) a pak teprve posílám packety ve while cyklu.

Z důvodu využívání timeoutu by mohl nastat problém, při komunikaci s hodně vzdáleným místem. Další z úskalí programu je přetékání proměnných. Do proměnné

3 Příklad spuštění

Jako příklad řešení uvádím merlin lokální měření:

```
xhoraz02@merlin: ~/ipk/prjk2$ ./ipk-mtrip reflect -p 9997
ort=9997
socket vytvoren
socket bindnut
cekam na ping
dosla zprava o rtt
ekam na ping
```

Obrázek 1: výpis konzole na straně relfectoru

```
xhoraz02@merlin: ~/ipk/prjk2$ make
gcc -std=gnu99 -Wall -W -Wextra -pedantic -g -Werror
                                                         ipk-mtrip.c -o ipk-mtrip
xhoraz02@merlin: ~/ipk/prjk2$ ./ipk-mtrip meter -p 9997 -h localhost -s 600 -t 9
host=localhost
port=9997
velikost sondy=600
cas=9
nastavuji socket
vstupuji do cyklu!
cekam na odpoved
RTT = 0.000101
rychlost=93013800 B/s
cekam na odpoved
RTT = 0.000141
rychlost=89921400 B/s
cekam na odpoved
RTT = 0.000086
rychlost=86086800 B/s
cekam na odpoved
RTT = 0.000040
rychlost=97099800 B/s
cekam na odpoved
RTT = 0.000109
rychlost=88493400 B/s
cekam na odpoved
RTT = 0.000120
rychlost=97340400 B/s
cekam na odpoved
RTT = 0.000162
rychlost=93223800 B/s
cekam na odpoved
RTT = 0.000166
rychlost=90806400 B/s
cekam na odpoved
RTT = 0.000184
rychlost=90546000 B/s
prumerne RTT = 0.000123
minimalni rychlost = 82.098770 MiB/s
maximalni rychlost = 92.831039 MiB/s
prumerna rychlost = 87.000000 MiB/s
xhoraz02@merlin: ~/ipk/prjk2$
```

Obrázek 2: výpis překladu a následně výpis metru

4 Zdroje

https://www.sallyx.org/sally/c/linux/internet

https://www.sallyx.org/sally/c/linux/udp

https://www.sallyx.org/sally/c/c20.php

https://www.wikihow.cz/Jak-vypo%C4%8D%C3%ADtat-sm%C4%9Brodatnou-odchylku

https://stackoverflow.com/questions/3437404/min-and-max-in-c

https://en.wikipedia.org/wiki/Sizeof

http://man7.org/linux/man-pages/man3/gethostbyname.3.html

https://linux.die.net/man/7/socket

http://pubs.opengroup.org/onlinepubs/7908799/xsh/systime.h.html

https://www.gta.ufrj.br/ensino/eel878/sockets/sockaddr_inman.html

https://www.sallyx.org/sally/c/signal.php

https://en.wikipedia.org/wiki/C_data_types