VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

SÍŤOVÉ APLIKACE A SPRÁVA SÍTÍ
NETWORK APPLICATIONS AND NETWORK ADMINISTRATION

MĚŘENÍ ZTRÁTOVOSTI A RTT

PROJEKTOVÁ DOKUMENTACE
PROJECT DOCUMENTATION

AUTOR PRÁCE **ANETA HELEŠICOVÁ**
AUTHOR

BRNO 2017

Obsah

1 Úvod **2**
1.1 Zadání 2
1.2 Cíl práce 3

2 Teorie **4**
2.1 O co se jedná? 4
2.2 Teorie implementace 4
2.3 Výpisy 4

3 Implementace **6**
3.1 Jádro projektu 6

Kapitola 1

Úvod

Tato dokumentace byla vytvořena k procektu Programování síťové služby, varianty Měření ztrátovosti a RTT do předmětu Síťové aplikace a správa sítí školního roku 2017/2018. Cílem dokumentace je pečlivě popsat způsob řešení problému.

1.1 Zadání

Pro úplnost dokumentace je níže vloženo zadání z informačního systému.

Popis varianty: Vytvořte program, který monitoruje zadané síťové uzly. V případě ztráty paketů nebo překročení Round trip time (RTT) nad zvolenou hodnotu, vypíše informace na standardní výstup.

Příklad spuštění:

./testovac [-h] [-u] [-t <interval>] [-i <interval>] [-p <port>] [-l <port>] [-s <size>] [-r <value>]<uzel1> <uzel2> <uzel3> ...

-h – zobrazí nápovědu

-u - pro testování se použije UDP protokol

-s - velikost dat pro odeslání, výchozí hodnota 56B

-t <interval> – interval v sekundách za který je ztrátovost vyhodnocována, ve výchozím nastavení 300s

-i <interval> – interval v ms jak často zasílat testovací zprávy, ve výchozím nastavení 100 ms

-w <timeout> – doba jak dlouho se čeká na odpověď, pouze při neobdržení odpovědi, výchozí hodnota 2s jinak 2xhodnota vypočteného RTT

-p <port> – specifikace UDP portu

-l <port> – specifikace naslouchaného UDP portu

-r <value> – specifikace RTT hodnoty, pokud RTT překračuje danou hodnotu, reportuje se

-v - verbose mód, program vypisuje na stdout přijaté pakety, tj. chová se jako příkaz ping

<uzel> – IPv4/IPv6/hostname adresa uzlu

Upřesnění zadání: Pokud není specifikováno přepínačem -u a -p, použije se pro testování protokol ICMP echo request/reply. Při specifikaci protokolu UDP bude program zasílat UDP paket na daný port s 64B náhodných dat a bude očekávat odpověď se stejným obsahem. Pokud je specifikován port přepínačem -l, program naslouchá na UDP portu a na příchozí pakety odpovídá UDP paketem se stejným obsahem. Program musí zpracová-

vat jednotlivé uzly paralelně. Pokud není zadán přepínač -r, testuje se pouze ztrátovost paketů, RTT se v tomto případě počítá pouze pro souhrné statistiky. Lze doporučit inspiraci nástrojem ping, který např. vkládá timestamp do obsahu ICMP paketu pro snadnější implementaci a výpočet RTT. Nicméně není to povinnost takto řešit.

Příklad výpisu v případě detekovaných chyb:

2017-02-01 10:35:12.23 uzel3: 0.167% packet loss, 5 packet lost

2017-02-01 10:35:12.23 uzel1: 0.067% packet loss, 2 packet lost

2017-02-01 10:35:12.23 uzel2: 0.067% (2) packets exceeded RTT threshold 5ms

Každou hodinu vypíše program statistiku o stavu jednotlivých uzlů: 2017-02-01 12:00:00.00 uzel1: 0% packet loss, rtt min/avg/max/mdev 4.845/4.882/4.912/0.063 ms

2017-02-01 12:00:00.00 uzel2: 0% packet loss, rtt min/avg/max/mdev 18.283/18.311/18.343/0.024 ms

2017-02-01 12:00:00.00 uzel3: status down

Pro výpis RTT, průměrných a max. hodnot zaokrouhlujte na 3 desetinná čísla. Za status down je označen prvek, který neodpovídá na testovací data (paket loss = 100%). Pro testování můžete interval pro výpočet statistiky zkrátit.

Příklad výpisu při přepínači -v:

2017-02-01 10:35:12.23 64 bytes from uzel3 (172.16.23.238) time=4.76 ms

2017-02-01 10:35:12.33 64 bytes from uzel3 (172.16.23.238) time=4.73 ms

2017-02-01 10:35:12.43 64 bytes from uzel3 (172.16.23.238) time=4.73 ms

Rozšíření: Za rozšíření nelze získat body nad rámec projektu.

Lze implementovat rozříření, kde RTT hodnota je specifikovaná pro každý uzel zvlášť. Syntaxe je následující: <uzel;RTT> - např. 2001:db8::1;4.5 znamená monitorování prvku s IPv6 adresou 2001:db8::1. Pokud dojde při měření ke zjištění RTT > 4.5 ms, bude hodnota zaznamenána a po uplynutí interval -t vypsána na standardní výstup nebo na syslog.

Vytvoření souboru testovac.spec umožňující vytvořit balíček .rpm

1.2 Cíl práce

Cílem projektu je tedy vytvořit program implementovaný v jazyce C nebo C++, který bude měřit ztrátovost a obousměrné zpoždění libovolného počtu na vstup zadaných uzlů. Program je vytvořený pro distibuci CentOS 7 a pro používání v příkazové řádce. Program smí používat pouze uživatelé s administrátorskými právy. O funkcionalitě si může do velké míry rozhodnout sám uživatel používáním parametrů.

Kapitola 2

Teorie

2.1 O co se jedná?

V této sekci bych se ráda pokusila vysvětlit projekt tak, aby mu rozuměl i laik. Dobrou průpravou pro tento text mi byl rozhovor s maminkou, která za mnou přišla s dotazem "A co že to vlastně deláš?". Takže, mami...

Dnes již téměř nedílnou součástí počítačů je jejich připojení do sítě a vzájemná komunikace mezi sebou. Bez internetu si už pomalu nedokážeme počítač představit. Proto se dostáváme do úzkých, když nám nějaký vzdálený počítač přestane odpovídat na naše zprávy. Pro zjištění toho, zda počítač odpovídá, běžně užíváme nástroje `ping`. Ten je ovšem vhodný pouze pro krátkodobé testování. Pokud chceme zjistit, jak nám vzdálený počítač odpovídá v průběhu hodin či dní, je vzhodnější využít právě programu takového, jaký zde prezentuji. Tento program totiž sám počítá statistiky odpovědí, dokáže nám říct na kolik procent zpráv nám druhý počítač odpověděl a kupříkladu taky jak dlouho mu to průměrně trvá.

Tyto programy jsou tedy, jak je asi zřejmé, důležitou součástí testování sítí.

2.2 Teorie implementace

Testování je implementováno pomocí zasílání ICMP packetů TCP protokolem.

Program zasílá jeden ICMP packet typu ECHO REQUEST každou časovou jednotku (zadáno uživatelem). Následně čeká na ECHO REPLY k tomuto packetu. Po obdržení je vyhodnoceno obousměrné zpoždění packetu - čas od odeslání ECHO REQUEST až po příjem ECHO REPLY. Aby si program byl jist, zda se jedná o správnou odpověď, musí také testovat jednotlivé součásti přijmutého packetu. Pokud k packetu žádná odpověď nepřijde, případně přijde až příliš pozdě, je packet vyodnocen jako ztracený. Pokud si to uživatel přeje, packety se mohou vyhodnotit i jako zpožděné, pokud překročí uživatelem stanovenou hodnotu obousměrného zpoždění.

2.3 Výpisy

Výpisy jsou zcela jistě nedílnou součástí tohoto programu, kdyby se uživatel nedozvěděl co se děje, byl by mu program úplně k ničemu.

Program má několik typů výpisů. Prvním typem výpisu je výpis ztrátovosti. Ten se provádí každých 300 sekund, pokud uživatel nestanovil jinak (pomocí parametru -t) a pokud

byl za stanovenou dobu na uzlu nějaký problém (nenulová ztrátovost nebo zpožděnost). Tento výpis je v jednoduchém fotmátu TIMESTAMP UZEL: ZTRÁTOVOST. Timestamp je klasický formát času RRRR-MM-DD HH:MM:SS.MS, udává se čas, kdy byl výpis spuštěn. Hodnota uzel je taktéž velmi prostá, jedná se o uzel, na kterém je ztrátovost vyhodnocena, ve formátu v jakém byl uživatelem zadán na vstup. Hodnota ztrátovost je základním kamenem tohoto výpisu. Rozlišujeme dvě varianty - statistika ztrátovosti a statistika zpožděných packetů. Jedná-li se o statistiku ztátovosti, je klasicky vyhodnocena jako procentuální poměr ztracených packetů a přesným číselným vyjádřením počtu ztracených packetů. Pokud je ztrátovost 100%, uzel hlásí status down. Statistika zpožděnosti je ve formátu procentuální poměr zpožděných packetů, (počet zpožděných packetů) překročil uživatelem stanovenou maximální hodnotu obousměrného zpoždění *uživatelem stanovená hodnota* ms.

2017-11-20 12:52:35.20 seznam.cz: 0.165% packet loss, 2 packet lost

2017-11-20 12:52:35.20 seznam.cz: 0.330% (4) packets exceeded RTT threshold 35 ms

2017-11-20 12:52:35.20 seznam.cz: 0.330% (4) packets exceeded RTT threshold 35 ms 0.165% packet loss, 2 packet lost

Druhým typem výpisu jsou hodinové statistiky. Tento čas nemůže uživatel ovlivnit, každou hodinu se vypíše statistika ztrátovosti ve formě TIMESTAMP UZEL: ZTRÁTOVOST RTT-STATISTIKA. Hodnoty timestamp a uzel jsou stejně jako při předešlém výstupu. Hodnota ztrátovost je taktéž podobná, vyhodnocuje se jako procentuální ztrátovost packetů. Rtt statistika je novou hodnotou, která obsahuje několik typů údajů o obosměrném zpoždění, které by mohly uživatele zajímat. Přesněji se jedná a minimální hodnotu, průměrnou hodnotu, maximální hodnotu a mean odchylku obousměrného zpoždění všech packetů, které byly za poslední hodinu vyhodnoceny.

2017-11-20 12:52:35.20 seznam.cz: 0.165% packet loss, rtt min/avg/max/mdev

4.523/12.682/20.402/2.503 ms

Posledním typem výpisu je výpis, který si může uživatel zapnout argumentem -v. Jedná se o takzvaný verbose výstup, s jehož používáním je program velmi podobný nástroji `ping`. Výpis probíhá při každém přijatém packetu a je ve formátu TIMESTAMP VELIKOST UZEL (IP) RTT. Timestamp a uzel jsou upět známé hodnoty. Hodnota velikost udává velikost packetu, který přišel na vstup. (Pozor, nerovná se případnému zadanému -s, neboť -s udává jen náhodná data, která se za packet přidávají, zde se jedná o velikost celého packetu.) . Hodnota IP je důležitá zejména, je-li uzel zadán pomocí hostname. Nabývá totiž hodnoty IPv4/IPv6 adresy, na kterou packet zasíláme. Poslední hodnota Rtt nám ve formátu time=*obousměrné zpoždění konkrétního packetu* ms.

2017-11-20 12:52:35.20 75 bytes from seznam.cz (77.75.79.39) time=35.245 ms

Kapitola 3

Implementace

V této kapitole se budu zabývat řešením implementace.

3.1 Jádro projektu

Projekt jsem implementovala v jazyce C++, standart C++11. Vyvíjen a testován je pro operační systém CentOS 7, k testování jsem používala virtuální stroj, jehož předpis jsme dostali k dospozici.

Běh programu začíná zpracováním argumentů, které se uloží do struktury. Pokud byl zadán argument -h, je vypsán Help a program se ukončí. Jinak se pro každý zadaný uzel vytvoří dvě `std::thread` vlákna, jedno pro zasílání packetů, druhé pro jejich přijímání. Dvě vlákna jsou důležitá, aby program mohl dodržovat rozestupy pro zasílání zpráv bez toho, aby musel čekat na odpověď, což by v případě, kdy by musel čekat déle, než dobu, za kterou má poslat další packet, způsobovalo nevyžádané chování.

Ve vlákně k zasílání packetů pak proběhne získání dat o adrese pomocí `gethostbyname` v případě IPv4 a `getaddrinfo` v případě IPv6 adresy. Dále dojde k vytvoření RAW socketu pro ICMP pomocí funkce `socket`. Tento socket se pak nastaví funkcí `setsockopt`. Následuje vytvoření ICMP hlavičky a datové části packetu. Nastavení ICMP hlavičky probíhá standartním způsobem. Velikost datové části se určí pomocí argumentu -s, případně je defaultně nastaveno 56B. Na začátek datové části se uloží informace o aktuálním čase, za tato data se připojí náhodná data o správné velikosti. Datová část s hlavičkou se poté spojí, vypočítá se kontrolní součet, který se uloží do hlavičky a packet je zaslán funkcí `sendto`. Poté se thread na počet sekund stanovený parametrem -i, případně defaulně na 100 ms uspí funkcí `usleep`. Tento thread má ještě jednu funkci a to zajišťování výpisů ztrátovosti a hodinových výpisů. Zda je správný čas zjišťuje pomocí srovnávání dvou `timeval` struktur nastavovaných funkcí `gettiomeofday`. V případě, že je čas některý výpis vypsat, přejde do funkce buď `tOutput` nebo `hourOutput`. Výpisy jsou popsány v předešlé kapitole.

Druhé vlákno slouží k přijímání packetů. Začne vytvořením socketu a jeho nastavením stejně jako thread pro zasílání. Dále v nekonečné smyčce naslouchá pomocí funkce `recvfrom`. Přijde-li nějaký packet, zkontroluje především jeho typ (`ICMP ECHOREPLY`), id a neporušenost dat. Pokud packet vše správně splňuje, označí jej za přijatý. Pokud byl zadán parametr -r a packet svým rtt překročí stanovenou hodnotu, je označen za zpožděný. Pokud na packet vlákno čeká déle než hodnotu -w nebo dvojnásobek předešlého rtt, je označen za ztracený. Vlákno tedy ještě počítá rtt každého packetu, tuto hodnotu pak ukládá pro

pozdější zpracování. Dále má na starosti výpis verbose, pokud byl uživatelem vyžádaný pomocí parametru -v (popsán taktéž v předešlé kapitole).

Program aktuálně funguje pouze pro IPv4 adresy a hostname, která jsou do IPv4 přeložitelná. Při zadání parametru -u je vypsána chybová hláška a program dále pokračuje stejně, jako by parametr zadán nebyl - pomocí ICMP.