Střední průmyslová škola, Ústí nad Labem, Resslova 5

Peer-to-Peer Chat

Dokumentace k ročníkové práci

Autor: Karel Zdešek

Třída: 3ITA

Vedoucí práce: Ing. Petr Haberzettl

2023/2024

Prohlášení

Prohlašuji, že jsem ročníkovou práci na téma „Peer-to-Peer Chat“ vypracoval samostatně a s použitím uvedené literatury a pramenů.

V (název obce, kde podepisuji) dne ……..

…….

Poděkování

Chtěl bych poděkovat Ing. Petru Haberzettlovi za vedení mé ročníkové práce, cenné rady a odborný dohled.

Anotace

Tento dokument slouží jako dokumentace k prvnímu pololetí této ročníkové práce. Dokument začíná úvodem, který nastiňuje vizi této ročníkové práce. Dále následuje rešerše, kde se pojednává o nápadu práce. Poté je sepsána řada technologií, které jsem využil. Praktická část se věnuje projektu samotnému. Nahlíží na teoretickou část práce. To, jak by hlavní princip tohoto projektu měl fungovat. V produktizaci se nahlíží do kódu a následně je zobrazena aplikace a popsána uživateli. Dokumentace je zakončená závěrem, kde je popsáno, jestli bylo naplněno cílů a jakých jsem nabyl pocitů.

Klíčová slova

Klient, server, soket, peer-to-peer, uživatel, ip adresa, port, zpráva, počítačová síť

Obsah

Úvod

Cílem této ročníkové práce je vytvoření aplikace, jejíž účelem je umožnit uživatelům si pohovořit v reálném čase. Uživatelé si budou moci pohovořit v chatovacích místnostech, jež daný uživatel bude moci vytvořit.

Hlavní zaměření této práce je, aby klienti, tj. uživatelé si mohli mezi sebou komunikovat bez potřeby serveru, který by jejich zprávy přeposílal. Avšak by musel existovat server, který by pomohl klientům vyměnit si jejich veřejné IP adresy a vnější porty.

Aplikace by měla nabídnout uživatelům jednoduché grafické rozhraní. Při spuštění aplikace by měl uživatel dvě volby: připojit se k místnosti, či ji vytvořit. Při vytvoření místnosti by měla býti dostupná možnost, kolik uživatelů může v dané místnosti být. Komunikace uživatelů by měla probíhat v reálném čase. To by mělo za důsledek, že zprávy nikam neukládají a nově připojení uživatelé by v místnosti neviděli historii zpráv.

1 Rešerše

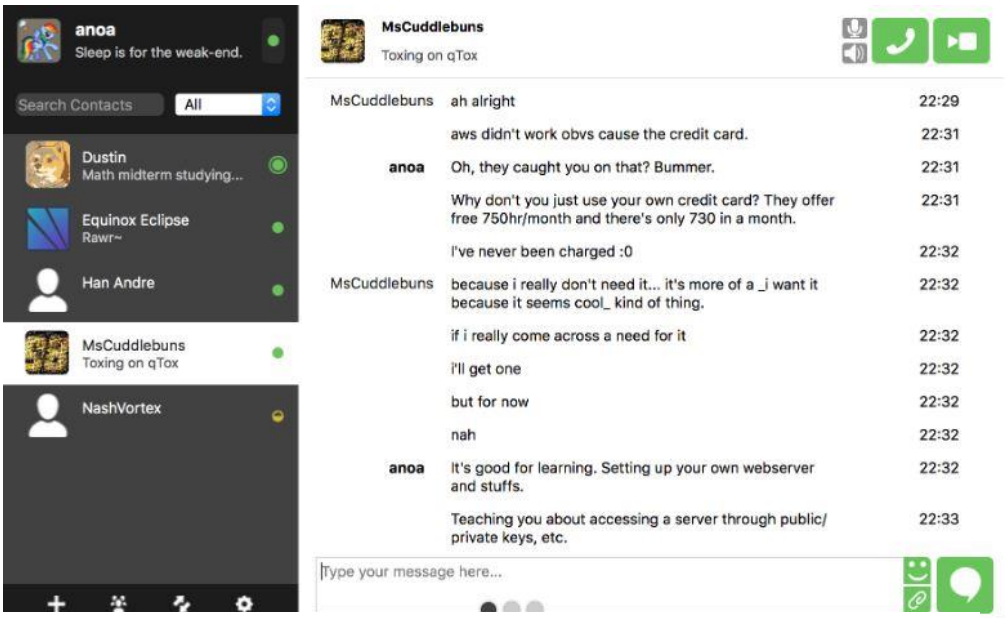
Námět tohoto projektu vznikl na základě síťové architektury Peer-to-Peer, kdy klienti v počítačové síti mohou komunikovat bez centralizovaného serveru. Je mnoho variant jak Peer-to-Peer sítě implementovat. To, jak má Peer-to-Peer síť bude fungovat jsem si navrhnul sám, avšak jsem si potřeboval udělat rešerši o teorii, jak by mohlo takové navazování spojení v Peer-to-Peer sítích vypadat. Od konkurenčních produktů jsem se nijak neinspiroval. Věděl jsem, že toto téma bude pro mě obtížné a s konkurencí jsem si nedělal hlavu.

1.1 Konkurence

Vývoj Peer-to-Peer chatovacích aplikací je nejrozšířenější mezi open-source komunitou. Nejlepší místo, kde hledat takovéto aplikace je verzovací systém GitHub.

1.1.1 Tox

Aplikace pro přenos zpráv, hlasu, videa a souborů. Používá svůj vlastní šifrovaný Peer-to-Peer protokol.



1. Tox

1.1.2 Briar

Mobilní aplikace primárně zaměřená na přenos textových zpráv. Používá Wi-Fi a Bluetooth pro spojení v lokálních sítích. A Tor síť na širokém internetu.

2 Technologie

2.1 C

Nízkoúrovňový programovací jazyk. Byl navrhnut pro účely operačního systému UNIX. **(1)**

Jazyk jsem si vybral, protože nabízí jednoduše použitelnou knihovnu pro interakci se sokety.

2.2 C Networking API

Řada knihoven v jazyku C, slouží jako programové rozhraní (API) pro práci se sokety.

2.3 Rasberry Pi

Rasberry Pi je název řady jednodeskových počítačů vytvořených Rasberry Pi Foundation, britskou charitativní organizací, jejíž cílem je vzdělání v oblasti informatiky a usnadnění přístupu k němu. **(2)**

Rasberry Pi používám jako server. Většinově testuji program na loopback rozhraní, ale při nutnosti testování reálného spojení používám Rasberry jako server.

2.4 Wireshark

Wireshark je počítačový software určený k analýze síťového provozu (tzv. packet analyzer). Používá se k odhalování a řešení problémů při práci s počítačovými sítěmi. **(3)**

Wireshark používám k tomu, abych hlavně poznal, jestli zprávy proudí síťovým provozem.

2.5 GCC compiler for C

Nástroj pro převedení zdrojového kódu do strojového.

2.6 Visual Studio Code

Vývojové prostředí a editor zdrojových kódů od firmy Microsoft. **(4)**

Využívám už dlouhodobě, dobré uživatelské rozhraní pro interakci se soubory i možnost vytvoření více oken příkazového řádku.

2.7 Make

Nástroj pro automatizaci vytvoření spustitelných souborů ze zdrojového kódu.

3 Praktická část

3.1 Teorie

3.1.1 Peer-to-Peer

Práce je založena na síťové architektuře Peer-to-Peer. Základní princip Peer-to-Peer sítí je komunikace počítačů bez centralizovaného serveru, který by zprostředkovával jejich komunikaci.

3.1.2 UDP hole-punching

Metoda, pomocí níž klienti z privátních počítačových sítích navazují obousměrná spojení a překonávají tak překlad privátní síťové adresy na veřejnou (NAT), pomocí veřejně dostupného serveru. Klienti tak učiní, že veřejně dostupný server jim pomůže ve výměně informací o veřejné IP adrese a vnějším portu routeru. Komunikace zprostředkována pomocí protokolu UDP, protože UDP záznam v NAT tabulce by měl vydržet nanejvýše dvě minuty. Klienti právě využívají tohoto záznamu a doufají že paket v cílovým portem, který klient od serveru dostal, NAT tabulka skrz záznam propustí.

Tato metoda je hlavní princip Peer-to-Peer komunikace této aplikace.

3.1.3 Session Traversal Utilites for NAT (STUN)

U některých sítí se děje, že při cestě na široký internet může poskytovatel Internetového připojení (ISP), měnit veřejnou IP adresu. STUN je protokol, který pomáhá klientům zjistit jejich veřejné IP adresy a vnější port i přes tyto problémy.

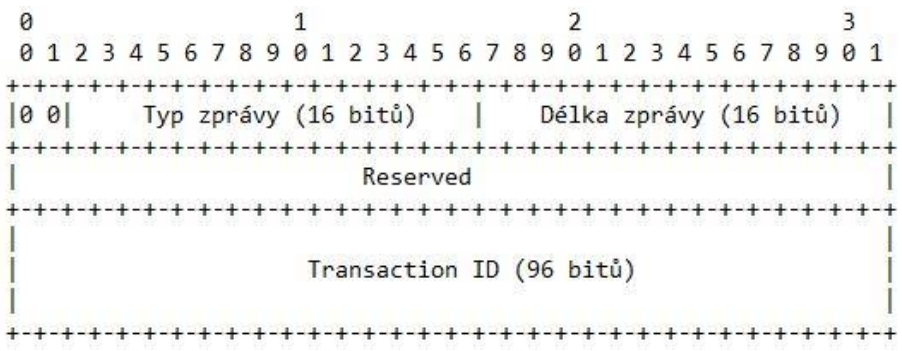
Formát zpráv je inspirován na základě tohoto protokolu.

3.2 Přehled operace

Mezi serverem a klientem existují dva typy transakce: create a finding request/response. Klient vytváří create request (požadavek), v tu chvíli, kdy chce vytvořit místnost a finding request, když se chce k místnosti připojit. Při vytváření místnosti serverem odpoví (response) klientovi s tím, že místnost byla vytvořena, či že místnost už existuje. Při připojování k místnosti, pokud se najde požadovaná místnost, tak server nejprve odešle IP adresu a port o klientovi, který místnost vytvořil a následně zbývajících klientech, kteří jsou připojeni k místnosti. Pokud server žádnou místnost nenajde, tak odpoví klientovi, že žádnou místnost nenašel.

3.3 Formát zpráv

3.3.1 Hlavička zprávy



2. Hlavička P2P chat zprávy

Jak již dříve řečeno, formát zpráv je inspirován z protokolu STUN a hlavička není moc odlišná. První dva bity jsou nulové defaultně, typ zprávy stanovuje, o jakou zprávu se jedná (požadavek, odpověď, či typy již zmíněné dříve). Délka zprávy určuje délku zprávy bez hlavičky zprávy. Transcation ID – náhodně vybrané číslo dlouhé 96 bitů, které pomáhá při asociaci požadavku a odpovědi při komunikace mezi klientem a serverem.

3.3.2 Data zprávy

Data jsou posílána na základě typu zprávy. To znamená, že každý typ zprávy dodržuje určitý řád, co ta zpráva může obsahovat. Např. obsah zprávy u vytvoření místnosti se liší mezi obsahem zprávy při připojení k místnosti.

3.4 Produktizace

3.4.1 Vytvoření požadavku o připojení k místnosti

```
8   struct cm
9   {
10      int16_t message_type;
11      int16_t message_length;
12      uint8_t trasaction_id[12];
13      uint8_t attributes[MINLENGTH];
14  };
15  typedef struct cm CLIENT_MSG;
```

3. Nadefinovaná struktura pro zprávu v header file klienta

Výše zobrazený obrázek zobrazuje nadefinovanou strukturu v header souboru klienta pro zprávu. Pro typ zprávy a délku zprávy je používán datový typ, který dává danou délku proměnné 16 bitů. Dále jsou ve struktuře nadefinována dvě celo-číselná pole s velikostí jednoho indexu 8 bitů. První jest transaction id s délkou 12 (12 * 8 jest 96 bitů). A pole attributes, které slouží jako payload zprávy.

```c
void make_find_req(CLIENT_MSG find_req, int clientfd,char roomname[],char username[], struct sockaddr_in address, int address_len)
{
    bzero(&find_req, sizeof(find_req));
    time_t t;
    srand((unsigned)time(&t));

    find_req.message_type = 0x01;
    for (int i = 0; i < sizeof(find_req.trasaction_id) / sizeof(find_req.trasaction_id[0]); i++)
    {
        find_req.trasaction_id[i] = rand() % 256;
    }
    find_req.attributes[1] = 0x01;

    find_req.attributes[13] = 0x01;
    find_req.attributes[14] = strlen(roomname);

    char let;
    int a;

    for(int i = 15; i < 14 + strlen(roomname); i++)
    {
        let = roomname[i - 15];
        a = (int)let;
        find_req.attributes[i] = a;
    }

    find_req.attributes[16 + strlen(roomname)] = strlen(username);
    for (int i = 17 + strlen(roomname); i < 17 + strlen(roomname) + strlen(username); i++)
    {
        let = username[i - (17 + strlen(roomname))];
        a = (int)let;
        find_req.attributes[i] = a;
    }

    u_int16_t Value_size = 0;

    find_req.message_length = htons(sizeof(find_req.attributes));
    find_req.attributes[3] = ( htons(Value_size) >> 8 ) & 0xFF;

   int hh = sendto(clientfd,&find_req,sizeof(find_req),MSG_WAITALL,(struct sockaddr*)&address,address_len);
   if( hh < 0)
   {
     printf("\n Last error was: %s",strerror(errno));
     fflush(stdout);
   }

}
```

4. Funkce vytvářející požadavek pro připojení k místnosti

Výše zobrazený obrázek vyobrazuje funkci, která vytváří požadavek pro připojení k místnosti. Funkce přijímá následující parametry: strukturu zprávy, souborový deskriptor soketu, název místnosti, jméno uživatele, strukturu držící informace o serveru, velikost předtím zmíněné struktury. Zpráva je vytvářena před vstupem do funkce, deskriptor soketu je vytvořen při spuštění programu, jméno a místnost je získána pomocí uživatelského vstupu, struktura držící informace o serveru je nadefinována při inicializaci soketu.

Nejdříve se vynuluje zpráva, poté se začne tvořit. Začne se stanovením typu zprávy a vytvořením transaction id. Následně se začne plnit payload zprávy jménem místnosti a jménem klienta. Děje se tak, ve dvou for cyklech, kdy se zjistí písmenko z daného řetězce a přetypuje se na celé číslo a uloží se do zprávy. Před tím, než řetězce jsou do zprávy zapsány, tak před ně jsou zapsány jejich délky. Následně do hlavičky zprávy uloží délka nákladu a odešle se serveru skrz funkci sendto, která je dostupná skrze knihovnu pro provádění operací se soketem. Poté jen následuje podmínka, jestli odeslání zprávy bylo úspěšné.

3.5 Popis pro uživatele

5. Aktuální uživatelský zážitek

V této chvíli aplikace postrádá grafické rozhraní, a spouští se v příkazovém řádku. Poté co uživatel spustí spustitelný soubor klienta, tak má na výběr dvě možnosti: první je připojit se k místnosti, či ji vytvořit (možnost 1,2). Po spuštění uživatel zadává potřebné údaje. Po úspěšném výměně informací mezi serverem klient přechází do stavu, kdy může začít chatovat.

Závěr

Za první pololetí, se podařilo splnit hlavní náplň cílů na každý měsíc krom ledna. Plnění těchto hlavních cílů však mělo dopad na to, že vedlejší prvky, méně tak důležité, postrádají. Např. transaction id, které funguje jako označení pro konverzaci mezi klientem a serverem, je ve zprávách zakomponováno, ale není funkci autentifikace. To, jak vypadá kód si také vybralo svou daň.

Největší obtíží bylo programování v jazyku C. Před začátkem tohoto projektu, jsem sotva měl znalost o programování v tomto jazyce. Pouze jsem si dělal malinké programy se sokety, které by mě připravily na takovýto projekt. Ačkoliv se mám ještě mnoho co učit, tak si myslím, že od začátku po teď, jsem se mnohé naučil.

Plánuji do budoucna plnit cíle, které jsem si zadal tak jako doteď.

Použitá literatura

1. **Co je to C? *ITSlovník.* [Online] [Citace: 12. 1 2024.] https://it-slovnik.cz/pojem/c.**

2. K čemu je Raspberry Pi a co se s ním dá dělat? *Chip.* [Online] [Citace: 12. 1 2024.] https://www.chip.cz/k-cemu-je-raspberry-pi-a-co-se-s-nim-da-delat.

3. Co je to Wireshark? *ITSlovník.* [Online] [Citace: 12. 1 2024.] https://it-slovnik.cz/pojem/wireshark.

4. ITSlovník. *Co je to Microsoft Visual Studio Code?* [Online] [Citace: 13. 1 2024.] https://it-slovnik.cz/pojem/microsoft-visual-studio-code.

Seznam obrázků