VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

IRC BOT S LOGOVANÍM SYSLOG

IRC BOT WITH SYSLOG LOGGING

SIEŤOVÉ APLIKÁCIE A SPRÁVA SIETÍ
NETWORK APPLICATIONS AND NETWORK ADMINISTRATION

AUTOR **JURAJ ONDREJ DÚBRAVA**
AUTHOR

BRNO 2017

Obsah

1 Úvod **2**
- 1.1 Protokol IRC . . . 2
- 1.2 Správy IRC . . . 2
- 1.3 Príkazy IRC . . . 3
  - 1.3.1 USER . . . 3
  - 1.3.2 NICK . . . 3
  - 1.3.3 JOIN . . . 3
  - 1.3.4 NAMES . . . 3
  - 1.3.5 PART . . . 3
  - 1.3.6 QUIT . . . 4
  - 1.3.7 KICK . . . 4
  - 1.3.8 PING, PONG . . . 4
  - 1.3.9 PRIVMSG, NOTICE . . . 4
- 1.4 Protokol SYSLOG . . . 5
  - 1.4.1 PRI . . . 5
  - 1.4.2 HEADER . . . 5
  - 1.4.3 MSG . . . 5

2 Riešenie problému **6**
- 2.1 Návrh riešenia . . . 6
- 2.2 Implementácia riešenia . . . 7
- 2.3 Argumenty . . . 7
- 2.4 Vytvorenie spojenia, komunikácia . . . 7
- 2.5 Spracovanie správ . . . 8
  - 2.5.1 NAMES . . . 8
  - 2.5.2 JOIN . . . 8
  - 2.5.3 NICK . . . 8
  - 2.5.4 PART, QUIT, KICK . . . 8
  - 2.5.5 PRIVMSG, NOTICE . . . 9
  - 2.5.6 PING . . . 9

3 Použitie programu **10**

4 Záver **11**
- 4.1 Testovanie . . . 11

Literatúra **12**

Kapitola 1

Úvod

Tento text slúži ako dokumentácia k projektu do predmetu ISA. Zadaním projektu bolo vytvoriť jednoduchého IRC bota, ktorý bude automaticky sledovať aktivitu na kanáloch IRC servera, poskytovať 2 základné funkcie a umožní logovať správy pomocou protokolu SYSLOG. Dokument v úvode predstavuje problematiku a popis protokolov IRC a SYSLOG, ďalej popisuje návrh riešenia aplikácie, implementáciu riešenia problému a použitie programu užívateľom.

1.1 Protokol IRC

IRC (Internet relay chat)[2] je protokol určený na okamžitú textovú komunikáciu. Protokol IRC bol navrhnutý na systémoch využívajúcich transportný protokol TCP. Typický základ komunikačného systému zahŕňa server, každý server tvorí centrálny bod pre ostatné servery alebo klientov, kde sa pripájajú. Server spracováva jednotlivé doručené príkazy od klientov alebo iných serverov.

Servery IRC tvoria kostru systému. Jednotlivé servery tvoria body, kde sa pripájajú užívatelia a navzájom medzi sebou komunikujú. Taktiež sa tam môžu pripojiť aj iné servery a formovať tak IRC sieť.

Klientom je každý, kto je pripojený na server a zároveň nie je serverom. Klienti sú rozlíšení unikátnym používateľským menom s maximálnou dĺžkou 9 znakov. Kvôli pôvodu protokolu IRC sa nerozlišuje veľkosť písmen, takže používateľské meno napísané veľkými a malými písmenami predstavuje toho istého používateľa.

Ďalšou dôležitou súčasťou sú IRC kanály. Kanál predstavuje skupinu jedného alebo viacerých klientov, ktorí dostávajú správy určené tomuto kanálu. Meno kanálu musí začínať znakom #, &, ! alebo +.

1.2 Správy IRC

Servery a klienti si medzi sebou vymieňajú správy, ktoré vždy nemusia viesť k odpovedi. Ak správa od klienta obsahuje validný príkaz, klient môže od serveru očakávať odpoveď. Každá správa by sa mala skladať z 3 hlavných častí: prefixu, ktorý je voliteľný, príkazu a parametrov príkazu. Prítomnosť prefixu indikuje znak „:“ na začiatku správy. Prefix využívajú servery na indikovanie pravého pôvodu správy. Príkaz má buď textovú alebo číselnú podobu, musí sa však jednať o validný textový IRC príkaz alebo 3-miestne číslo reprezentujúce príkaz. Každá správa musí byť ukončená znakmi CRLF.

1.3 Príkazy IRC

Táto časť popisuje vybrané príkazy protokolu IRC, ktoré boli v programe implementované.

1.3.1 USER

Príkaz USER sa používa na začiatku spojenia na špecifikovanie položiek username, hostname, realname, servername nového užívateľa. Tieto položky slúžia na identifikáciu užívateľa na serveri.

Príklad: USER juraj juraj_d juraj :Juraj

1.3.2 NICK

Príkaz slúži na priradenie používateľského mena užívateľovi alebo na zmenu pôvodného. Taktiež sa používa na začiatku spojenia spoločne s príkazom USER. Ak správu NICK obdrží server, na ktorom sa už nachádza užívateľ s rovnakým menom, server odosiela odpoveď s kódom chyby.

Príklad:

NICK juraj nový nick juraj

:Peter NICK juraj užívateľ Peter mení svoj nick na juraj

1.3.3 JOIN

Príkaz je určený na pripojenie klientov k určitému IRC kanálu, odkiaľ môžu prijímať správy alebo ich na kanál odosielať. Po tom, ako sa klient úspešne prihlásil na zvolený kanál, dostáva správu s témou kanálu a zoznamom aktívnych užívateľov na kanáli vrátane jeho samotného.

Príklad: JOIN #test pripoj sa na kanál s názvom test

1.3.4 NAMES

Použítím príkazu môže užívateľ zistiť používateľské mená, ktoré môže vidieť na všetkých jemu viditeľných kanáloch.

Príklad: NAMES #test,#foo zoznam viditeľných užívateľov na kanáloch test a foo

1.3.5 PART

PART je jedným z príkazov na ukončenie spojenia. Príkaz spôsobí, že klient odošle správu o tom, aby bol vymazaný zo zoznamu aktívnych užívateľov na všetkých špecifikovaných kanáloch.

Príklad: PART #test,#foo opusti kanály test a foo

1.3.6 QUIT

Príkaz QUIT tiež slúži na ukončenie spojenia, server po jeho obdržaní ukončuje spojenie s klientom.

Príklad: QUIT :gone

1.3.7 KICK

Príkazom KICK je možné násilné odstránenie užívateľa zo zvoleného kanálu. Vyhodiť užívateľa z kanálu môže len správca kanálu.

Príklad: KICK #test Juraj vyhoď používateľa Juraj z kanálu test

1.3.8 PING, PONG

Príkaz PING slúži na testovanie prítomnosti klienta na opačnej strane spojenia. Správa PING je odosielaná serverom v pravidelných intervaloch ak nie je zistená žiadna iná činnosť počas spojenia. Každý klient musí po doručení tejto správy čo najrýchlešie odpovedať odoslaním správy PONG. Ak server neobdrží odpoveď v určitom časovom úseku, spojenie sa uzavrie.

Príklad:

PING :kornbluth.freenode.net PING správa od serveru :kornbluth.freenode.net

PONG csd.bu.edu tolsun.oulu.fi PONG správa od csd.bu.edu pre tolsun.oulu.fi

1.3.9 PRIVMSG, NOTICE

Príkazy PRIVMSG a NOTICE sú určené na odosielanie užívateľských správ. Príkaz NOTICE je podobný príkazu PRIVMSG. Rozdiel je v tom, že automatické odpovede nemôžu byť odoslané ako odpoveď na správu NOTICE. PRIVMSG je používaný na odosielanie privátnych správ medzi užívateľmi, rovnako ako aj na odosielanie správ na kanály. Správa sa skladá z mena užívateľa, ktorému je správa určená, najčastejšie je to nick alebo názov kanálu, a samotného textu správy.

Príklad:

:juraj!wings@irc.org PRIVMSG #NWA4 :hello správa od užívateľa juraj na kanál NWA4 s textom hello

PRIVMSG juraj :hello others odoslanie PRIVMSG správy užívateľovi juraj

NOTICE juraj :hello others odoslanie NOTICE správy užívateľovi juraj

1.4 Protokol SYSLOG

SYSLOG je protokol, ktorý umožňuje zariadeniam odosielať správy so záznamom o udalosti. Bol vytvorený pre operačné systémy typu Unix. Využíva UDP transportný protokol a port číslo 514. Plný formát SYSLOG správy má 3 časti: PRI, HEADER, MSG. Dĺžka správy musí byť najviac 1024 bajtov [1].

1.4.1 PRI

Časť PRI sa musí skladať z 3, 4 alebo 5 znakov, kde prvým znakom je „<“ a posledným „>“. Medzi zátvorkami sa nachádza číselná hodnota nazývaná Priority vyjadrujúca hodnoty Facility a Severity.

Hodnoty Facility a Severity sú číselne kódované hodnoty. Hodnota Facility vyjadruje typ programu vytvárajúci správu, rozsah hodnôt je v intervale 0-23. Každá hodnota Priority má určený odpovedajúci stupeň Severity. Hodnoty Severity sú v rozsahu 0-7. Konečná hodnota Priority sa vypočíta ako Facility * 8 + Severity.

1.4.2 HEADER

Časť HEADER sa skladá z 2 častí: TIMESTAMP a HOSTNAME.

TIMESTAMP nasleduje priamo po znaku „>“. Pole vyjadruje časové razítko vo formáte *”Mmm dd hh:mm:ss”*. *Mmm* je anglická skratka pre názov mesiaca, *dd* je deň v mesiaci, *hh:mm:ss* je lokálny čas stroja, odkiaľ bola správa odoslaná. Prípustné hodnoty *hh* sú v rozsahu 00-23, *mm* a *ss* v rozsahu 00-59.

HOSTNAME obsahuje buď meno hosťa, IPv4 alebo IPv6 adresu zdroja správy. Preferovanou hodnotou je meno hosťa. V prípade IPv4 adresy to musí byť adresa v desiatkovom zápise oddelená bodkami.

1.4.3 MSG

MSG tvorí zvyšnú časť SYSLOG správy. Zvyčajne obsahuje dodatočné informácie o procese, ktorý správu vygeneroval a text správy. Táto časť nemá žiadny ukončovací znak. MSG je tvorená z 2 častí: TAG a CONTENT.

TAG obsahuje meno programu alebo procesu, ktorý vygeneroval správu. CONTENT obsahuje detaily o správe. Znakom oddeľujúcim pole CONTENT a TAG sú zvyčajne znaky „[“, „:“ alebo medzera.

Príklad SYSLOG správy:

<134>Oct 21 16:20:00 192.168.0.1 isabot xdubra00: irc bot isa

Kapitola 2

Riešenie problému

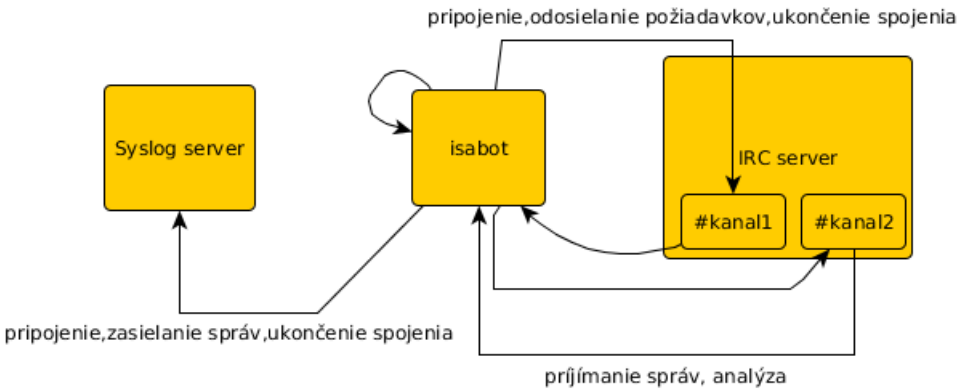
2.1 Návrh riešenia

Program isabot má slúžiť ako automatický bot na sledovanie aktivity na špecifikovaných IRC kanáloch serveru a logovanie správ na SYSLOG server. Na začiatku programu je potrebné najskôr spracovať argumenty programu, ktoré špecifikujú parametre potrebné na komunikáciu s IRC a prípadne aj SYSLOG serverom. Týmito parametrami sú názov IRC serveru spolu s číslom portu, ktorý je voliteľný, názvy jednotlivých IRC kanálov, zoznam kľúčových slov vyhľadávaných v správe a adresa SYSLOG serveru. Po overení správnosti argumentov sú ich hodnoty uložené v pomocnej štruktúre.

Po úspešnom spracovaní argumentov je nadviazané spojenie so serverom a bot sa pripojí na zadané IRC kanály. Na jednotlivých kanáloch je potrebné udržiavať zoznam aktívnych užívateľov. Na tento účel slúži asociatívny kontajner **online_users** s kľúčom, ktorý prestavuje názov kanálu. Každá položka je tvorená opäť z asociatívneho kontajnera, ktorého kľúč tvorí meno užívateľa na kanáli. U každého užívateľa sú udržiavané položky príznaku aktivity a vektoru nedoručených správ.

Príjem a spracovanie IRC správ pracuje v nekonečnom cykle, až kým program neobdrží signál SIGINT na ukončenie. Prijaté správy od servera sú spracovávané po ich doručení podľa typu príkazu. V prípade vyhľadávania kľúčových slov a logovania správ sa vytvorí spojenie so SYSLOG serverom, s ktorým sa komunikuje pomocou protokolu UDP.

IRC bot poskytuje 2 špeciálne príkazy **?today** a **?msg**, ktoré môžu byť doručené ako súčasť PRIVMSG správy. Ak IRC správa obsahuje iba príkaz *?today*, bot zistí aktuálny dátum a odošle ho na kanál odkiaľ správa prišla. V opačnom prípade nejde o validný príkaz *?today* a správa je braná ako obyčajná IRC správa. Príkaz *?msg* má formát ?msg nickname:message slúži na odoslanie textu správy späť na kanál, ak sa tam nachádza užívateľ s menom, ktoré je totožné ako nickname. Ak sa zistí prítomnosť príkazu *?msg*, je overená časť s nickom a textom správy. Po úspešnom overení je potrebné dodatočne zistiť, či užívateľ s nickom v príkaze je na danom kanáli aktívny. V prípade, že je prítomný, je správa odoslaná späť na kanál. Ak je užívateľ na kanáli neprítomný, doručené správy sú uložené v jeho zázname v mape užívateľov a odoslané sú až po opätovnom pripojení na kanál.



Obr. 2.1: Návrh riešenia aplikácie

2.2 Implementácia riešenia

Program je implementovaný v jazyku C/C++. Samotný program je tvorený z knižníc **bot_connection.h**, ktorá implementuje funkcie spojenia s IRC a SYSLOG serverom, a **bot_functions.h**, kde sú implementované jednotlivé funkcie bota spojené so spracovaním správ a sledovaním aktivity na IRC serveri.

2.3 Argumenty

Spracovanie argumentov prebieha vo funkcií *parse_arguments()*. Nebola použitá žiadna špeciálna funkcia typu getopt, argumenty som sa rozhodol spracovať klasicky. Vo funkcí najprv prebehne kontrola na prítomnosť možnosti nápovedy, ak je prítomná, vypíše sa nápoveda a program je ukončený. Ak sú argumenty vyhovujúce, ich hodnoty sú uložené do štruktúry Arguments na ďalšie použitie.

2.4 Vytvorenie spojenia, komunikácia

Funkcie realizujúce spojenie sú implementované v knižnici **bot_connection.h**. Spojenie s IRC aj SYSLOG serverom sa vytvára vo funkcí *create_connection()*. Na základe čísla portu sa nastaví typ spojenia na TCP alebo UDP. V prípade SYSLOG spojenia sa do pomocnej štruktúry **sys_info** uložia hodnoty potrebné pre funkciu *sendto()* a funkcia vráti vytvorený soket. Na vytvorenie TCP spojenia sa využíva funkcia *getaddrinfo()*, bot podporuje len IPv4 spojenie. Funkcia vracia zoznam adries, kde sa bot skúša pripojiť. Ak sa nepodarí pripojiť ani k jednej z adries v zozname tak nastáva chyba, inak funkcia vráti vytvorený soket.

Ďalšími funkciami v tomto module sú funkcie na začatie komunikácie s IRC serverom. Vo funkcií *irc_login()* sa vytvára a odosiela správa s požiadavkom na zaregistrovanie užívateľa na IRC serveri a vytvorenie užívateľského mena na komunikáciu. Funkcia *irc_join_channel()* slúži na pripojenie k IRC kanálom, *send_pong()* na odoslanie PONG

správy na server, aby bot potvrdil prítomnosť na serveri. Pri neúspechu komunikácie s IRC serverom nastáva chyba.

2.5 Spracovanie správ

Príjem a spracovanie IRC správ prebieha vo funkcií *parse_message()*. Funkcia prijíma a spracováva správy v nekonečnom cykle v hlavnej funkcí programu, kde môže byť prerušená signálom SIGINT. Príjem dát od IRC servera poskytuje funkcia *receive_data()*, ktorá využíva štandardnej funkcie *recv()*. Po doručení sa správa uloží do textového reťazca, s ktorým sa ďalej môže pracovať.

Ďalšou fázou je analyzovanie správy. Na začiatku prebehne overenie, či správa obsahuje prefix, zistí sa teda prítomnosť znaku „:“ na začiatku správy. Ak správa prefix obsahuje, pokračuje sa rozdelením obsahu na 3 časti: užívateľské meno, IRC príkaz a zvyšná časť správy, s ktorou sa pracuje podľa typu príkazu. Ak prišiel príkaz oznamujúci chybu, program je ukončený, inak sa pokračuje v spracovaní príkazu.

Nasledujúca časť popisuje spracovanie jednotlivých príkazov podporovaných botom.

2.5.1 NAMES

Pri úspešnom prihlásení na kanál server pomocou príkazu NAMES zistí aktívnych užívateľov na kanáli a v odpovedi odosiela ich zoznam. Odpoveď má číselný kód príkazu 353 a po jeho doručení sa do mapy *online_users* pridá kanál s položkami mien užívateľov. Taktiež sa užívateľovi nastaví príznak aktivity.

2.5.2 JOIN

Ak bot dostal správu s príkazom JOIN, znamená to, že sa na kanál pripojil nový užívateľ alebo sa užívateľ pripája opätovne. Uživateľské meno sa najprv prevedie na malé písmená a následne je pridané do mapy. Ďalej sa overí, či položka užívateľa neobsahuje nedoručené správy, ktoré mu mohli byť doručené v neprítomnosti pomocou príkazu ?msg. Ak má užívateľ nejaké nedoručené správy, okamžite sú odoslané na aktuálny kanál pomocou funkcie *send_data()* a vymazané z jeho záznamu.

2.5.3 NICK

Príkazom NICK sa mení pôvodné používateľské meno. Po doručení správy s týmto príkazom sa v mape pre všetky kanály vytvorí nová položka pre nové užívateľské meno s pôvodnými údajmi používateľa, starý záznam sa odstráni.

2.5.4 PART, QUIT, KICK

Všetky tri príkazy slúžia na ukončenie spojenia s IRC serverom. Pri spracovaní príkazu PART sa u užívateľa nastaví príznak aktivity na neprítomný funkciou *set_status()*. PART sa týka len jedného kanálu. Pri príkaze QUIT je potrebné nastaviť aktivitu na všetkých pripojených kanáloch.

Na násilné ukončenie spojenia slúži príkaz KICK. Najskôr sa overí, či vyhodený užívateľ nie je bot, v prípade zhody sa program ukončí. V opačnom prípade sa nastaví status pre vyhodeného užívateľa.

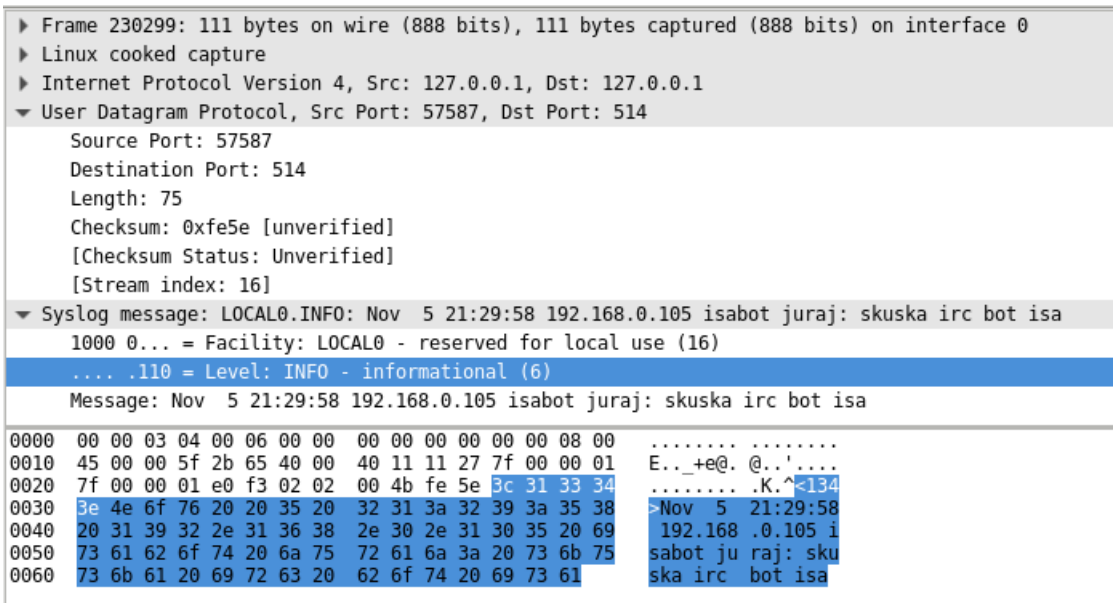
2.5.5 PRIVMSG, NOTICE

Správa obsahujúca príkaz PRIVMSG alebo NOTICE slúži na doručovanie správ medzi užívateľmi. Tento typ správy môže obsahovať špeciálne príkazy bota, preto sa overí ich prítomnosť v správe. Ak obsahuje príkaz *?today*, na aktuálny kanál odošle správu PRIVMSG s aktuálnym dátumom. Pri výskyte príkazu *?msg* sa najprv overí jeho správnosť. Ak je príkaz správny, pokračuje sa v kotrole statusu užívateľa pomocou funkcie *get_status()*, ktorému je správa určená. Neprítomnosť spôsobí, že správa sa uloží do vektoru nedoručených správ. Rovnako prebieha aj kontrola na parameter vyhľadávania kľúčových slov v správach. Ak je parameter zadaný a našlo sa kľúčové slovo, vytvorí sa SYSLOG správa.

Pri správe s príkazom NOTICE nemôže byť podľa RFC odoslaná odpoveď. Bot kontroluje iba prítomnosť argumentu vyhľadávania kľúčových slov a prípadne generuje a odosiela SYSLOG správu. Funkcie ?msg ani ?today nie sú podporované, pretože vedú k odpovedi.

SYSLOG správa sa vytvára pomocou funkcie *create_syslog_message()*. Správa obsahuje všetky polia SYSLOG správy. Hodnota Facility je 16, čo značí local use 0, a Severity 6 vyjadruje informačné správy. Položka HOSTNAME obsahuje IP adresu zariadenia, ktorá je získaná funkciou *get_my_ip()*. Tá využíva funkciu *getifaddrs()* zo štandardnej knižnice. Funkcia nájde všetky rozhrania zariadenia spolu s IP adresami. V SYSLOG správe som použil prvú adresu rôznu od adresy loopbacku. Pole CONTENT obsahuje meno užívateľa spolu s textom správy. Na odoslanie SYSLOG správy sa využíva funkcia *send_syslog()*.

Pri doručení obyčajnej správy bez prítomnosti špeciálneho príkazu prebieha iba overenie na parameter vyhľadávania kľúčových slov a prípadné vytvorenie SYSLOG správy.



Obr. 2.2: Syslog správa zachytená programom Wireshark

2.5.6 PING

Príkaz PING je doručený v správe bez prítomnosti prefixu. Po detekovaní príkazu sa na server odošle odpoveď s príkazom PONG, využije sa funkcia *send_pong()*, kde sa vytvára reťazec s odpoveďou a zároveň sa aj odošle.

Kapitola 3

Použitie programu

Program isabot má 4 voliteľné a 2 povinné argumenty. Voliteľné parametre sú uvedené v hranatých zátvorkách. Príklad spustenia programu je nasledovný:

./isabot HOST[:PORT] CHANNELS [-s SYSLOG_SERVER] [-l HIGHLIGHT] [-h|–help]

HOST[:PORT] doménové meno alebo IP adresa IRC servera s voliteľnou špecifikáciou čísla portu

CHANNELS názov IRC kanála, prípadne kanálov oddelných čiarkov

[**-s SYSLOG_SERVER**] parameter určujúci doménové meno alebo IP adresu SYSLOG serveru

[**-l HIGHLIGHT**] voliteľný parameter špecifikujúci zoznam kľúčových slov odddelených čiarkov

[**-h**|–**help**] výpis užívateľskej nápovedy

Kapitola 4

Záver

4.1 Testovanie

Na testovanie navrhnutého riešenia som si zvolil programy Hexchat a Wireshark. Program Hexchat bol použitý na testovanie komunikácie na IRC kanáloch. Pomocou neho som vytvoril testovacích klientov, ktorí si môžu medzi sebou odosielať správy. Ako testovací IRC server bol používaný IRC server **freenode**. Program Wireshark slúžil na overenie odoslania SYSLOG správy na server. Ak bola správa správne vytvorená a odoslaná, objaví sa v záznamoch protokolu SYSLOG. Taktiež som vyskúšal spustenie lokálneho UDP servera naslúchajúceho na porte 514 a pozoroval príchod SYSLOG správ.

Literatúra

[1] Lonvick, C.: The BSD syslog Protocol. RFC 3164, RFC Editor, Aug 2001.
URL http://www.rfc-editor.org/rfc/rfc3164.txt

[2] Oikarinen, J.; Reed, D.: Internet Relay Chat Protocol. RFC 1459, RFC Editor, May 1993.
URL http://www.rfc-editor.org/rfc/rfc1459.txt