VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Dokumentace k projektu pro předmět ISA

IRC bot s logováním SYSLOG

20. listopadu 2017

Autor: Kryštof Rykala, xrykal00

Fakulta Informačních Technologií

Vysoké Učení Technické v Brně

Obsah

Úvod

Tento dokument je dokumentací k programu isabot v rámci projektu programování síťové služby v předmětu síťové aplikace a správa sítí.

Popis programu

Program isabot, je IRC bot s logováním syslog. Jedná se o klienta, který se připojí na zvolený server a jeho kanály, na kterých poté sleduje komunikaci a na základě zpráv výkonává svou funkčnost. Bot má dvě hlavní funkce a to výpis konkretního data a zaslání zprávy uživateli, na danném kanálu. Bot rovněž umožnujě logování zpráv pomocí protokolu syslog, zprávy jsou logovány na základě klíčových slov, uvedených při spuštění.

Funkcionalita

Bot vykonává dvě funkce. Pokud je na nějaký kanál zaslána zpráva uvozená znakem ? a klíčovým slovem (?today a ?msg), provede bot definovanou operaci.

?today - zašle zprávu na daný kanál s datem ve formátu dd.mm.yyyy
?msg <nickname>:<msg> - pokud je uživatel na daném kanále, odešle mu zprávu okamžitě na tentýž kanál, pokud uživatel momentálně není na kanále, zprávu si uloží a odešle ji hned jakmile se daný uživatel připojí na daný kanál (obsah zprávy bude <nickname>:<msg>)

Další funkcí bota je sledování zpráv. Pokud zpráva obsahuje slovo ze seznamu klíčových slov je danná zpráva zalogována pomocí protokolu SYSLOG a poslána na uvedenou adresu.

Spuštění programu

Program je nutné nejdřívě přeložit příkazem make, tímto se nám vytvoří binární soubor s názvem isabot, který můžeme následně používat.

Použití programu je následující, při spuštění je nutné uvést název IRC serveru, popřípadě jeho IPv4 či IPv6 adresu. Je nutno rovněž zadat název alespoň jednoho kanálu, na který se bot připojí a na kterém bude poslouchat. Adresa syslog serveru není povinná, pokud není uvedena jako výchozí adresa bude použita IPv4 adresa 127.0.0.1. Pokud je seznám klíčových slov, pro logování zpráv, prázdný, isabot nic neloguje. V případě potřeby je možno vypsat nápovědu.

Použití

```
isabot HOST[:PORT] CHANNELS [-s SYSLOG_SERVER] [-l HIGHLIGHT] [-h|--help]
```

HOST je název serveru (např. irc.freenode.net)
PORT je číslo portu, na kterém server naslouchá (výchozí 6667)
CHANNELS obsahuje název jednoho či více kanálů, na které se klient připojí (název kanálu je zadán včetně úvodního # nebo &; v případě více kanálů jsou tyto odděleny čárkou)
-s SYSLOG_SERVER je ip adresa logovacího (SYSLOG) serveru
-l HIGHLIGHT seznam klíčových slov oddělených čárkou (např. "ip,tcp,udp,isa")

Příklady použití

```
isabot irc.freenode.net:6667 "#ISAChannel,#IRC" -s 192.168.0.1 -l "ip,isa"
isabot irc.freenode.net "#ISAChannel,#IRC" -l "ip,isa" -s 127.0.0.1
isabot irc.freenode.net #ISAChannel -l "tcp,udp"
```

Implementace

Zvolený přístup

Program je implementován pomocí programovácího jazyka C++ za pomocí objektově orientovaného návrhu, kdy každá logická část bota představuje jednu dílčí třídu, která ma za úkol plnit danou funkcionalitu.

Makefile

Soubor Makefile se nachází v hlavním adresáři a obsahuje jednotlivé cíle, které buď provádí danou akci, nebo pouze volají zanořený Makefile v adresáři /src, který ovšem obsahuje pouze cíle pro překlad programu.

Make all

Zavolá make ve složce /src a přesune vytvořený binární soubor do aktuální složky.

Make pack

Zabalí všechny relevantní soubory pro odevzdání projektu.

Make clean

Odstraní všechny nepotřebné soubory.

Make run

Přeloží a spustí program s příkladným použitím.

```
./isabot irc.freenode.net:6667 "#ISAChannel,#IRC" -s 192.168.0.1 -l "ip,isa"
```

Zpracování argumentů

O zpracování argumentů se stará třída Arguments. Tato třída rovněž obsahuje atributy pro hodnoty všech přepínačů, pokud danný přepínač má mít defaultní hodnotu, je tato hodnota implicitně takovému atributu, který reprezentuje přepínač, nastavena.

Zpracování argumentů je implementování ve veřejné metodě `parseArguments()`, která jako parametry obsahuje `int argc` a `char *argv[]`. Pole argv je poté zpracováno pomocí knihovní funkce `getopt_long()`, která je v této třídě natavena tak, aby pokrykla potřeby spouštění programu isabot. Klířové slova pro filtrování, jsou rozděleny pomocí metody `splitString()`, a jsou uloženy do vectoru řetězců. Pokud je uveden host i s číslem portu, je tento port z tohoto řetězce extrahován pomocí metody `setHost()` a je nastaven příslušnému atributu třídy Arguments.

Pro přístup k hodnotám přepínačů, které jsou uloženy jako atributy této třídy, slouží veřejné metody s prefixem get a názvem atributu, takzvané “gettery”.

Zpracování IRC zpráv

Zpracování IRC zpráv má na starosti třída s názvem IrcParser. Tato třída obsahuje metody, které berou jako parametr řetězec message, který představuje Irc zprávu. Obsahuje metody s dvěmi druhy prefixů, jedním z prefixů je is a druhým je get.

Metody označené prefixem is mají návratovou hodnotu typu boolean. Takovéto metody jsou 2 a tím je `isCommand()` a `isChatCommand()`. Obě metody ověřují zda se jedná o konkrétní příkaz ve zprávě IRC, ovšem druhá s tou vyjímkou, že tento příkaz je implementovanná funkcionalita bota, pro příkazy ?today a ?msg <uživatel>: <zpráva>, a tyto dva příkazy se nachází až v textu zprávy IRC a nikoliv jako položka command IRC protokolu (toto ovšem platí pro `isCommand`).

Metody s prefixem get, zpracovávají IRC zprávu a vrací jako řetězec hodnotu, kterou očekáváme, podle jména metody, například metoda `getMessageText()`, vrací jako hodnotu text, který obsahuje IRC zpráva. Tyto metody v případě výskytu chyby, či neobsahuje-li danná zpráva hodnotu, kterou metoda má extrahovat, vrací prázdný řetězec. při použití těchto metod musíme na tuto skutečnost brát zřetel.

TCP klient

O připojení pomocí TCP se stará třída s názvem TcpClient. Tato třída obsahuje tři veřejné metody, které se využívá pro práci s daným spojením. Metoda connect, má za úkol vytvořit socket a připojit se na něj, třída poté uchovává socket v atributu s názvem socket, pro další práci se spojením. Pro navázání spojení je možno využít hostname,IPv4 nebo IPv6. V případě výskytu jakkékoliv chyby je vytvořená run time vyjímka, kterou je nutno obsloužit.

Pro práci se spojením poté slouží metody `send()`, která zapíše na příslušný socket data, které této metodě předáme. A pro čtení ze socketu, slouží metoda `receive()`, která přečtě data ze socket a naplní jimi buffer, který předáme metodě jako parameter.

Třída obsahuje jedinou privátní metodu a tou je `getIpFromSocket()`. Tato metoda vrací jako návratovou hodnotu řetězec s IPv4 adresou klienta.

UDP klient

Pro připojení pomocí UDP se stará třída s názvem UdpClient. Tato třída obsahuje pouze dvě metody a těmí jsou `connect()` a `send()`. Metoda `connect()` se stará o vytvoření socketu. Data na daný socket poté zapisujeme pomocí metody `send()`.

Bot

Hlavní logika bota je implementována ve třídě s názvem IrcBot. V konstruktoru třídy se vytvoří všechny potřebné instance tříd TcpClient, UdpClient, Arguments, IrcParser a je vyvolána metoda `parseArguments()` třídy IrcParser. Start bota se provádí pomocí metody `start()`, ve které se naváže spojení se syslog serverem (pokud je zadáno alespoň jedno klíčové slovo pro filtrování zpráv) a Irc serverem, na který je následně odeslána zpráva, která nastaví nickname, username, hostname, servername a realname pro komuniaci se serverem a v další zprávě se připojí na dané kanály. Následně začné bot filtrovat jednotlivé IRC zprávy.

Filtrování zpráv

Filtrování zpráv je implementováno ve funkci filterMessages(). Tato metoda filtruje takové zprávy, které jsou pro důležité pro správnou funkcionalitu bota. Pokud se jedná o zprávu s příkazem PING, bot odpovídá příslušnou zprávou PONG, tak aby nedošlo k odpojení od serveru. Pokud se jedná o zprávu s PRIVMSG, dále testujeme zda-li se jedná o příkaz ?today nebo ?msg. V případě, že přijde zpráva s chybovým kódem 4xx nebo 5xx, je bot ukončen a na STDERR je vypsána chybová hláška, jelikož bot není schopen vykonávat svou funkcionalitu bez omezení.

Příkaz ?msg

Jedná-li se o příkaz *?msg* je nutno takovou zprávu vypsat jenom pouze, pokud je danný příjemce na kanálu přítomen, v opačném případě je tuto zprávu nutno uložit, do té doby, než se danný uživatel na kanál připojí. Pokud tedy bot zaregistruje příkaz *?msg*, je danná zpráva uložena do vektoru vektorů, který obsahuje 3 položky a to název kanálu, příjemce a text zprávy a zvýšší počítadlo aktuálních privátních zpráv, jelikož může nastat situace, že nám příjdou dvě zprávy za sebou, bez toho aniž bychom otestovali zda-li je uživatel první zprávy online a zprávu poslali. Bot poté odešle na server zprávu *NAMES* a v případě, že je danný uživatel online mu je zpráva odeslána a počítadlo zpráv je sníženo. V opačném případě, klient filtruje zprávy tak, že kontroluje kdo se na kanál připojí (zpráva *JOIN*), a jakmile takovouto zprávu obdrží zkontroluje, zda-li nově připojený uživatel není příjemcem nějáké zprávy. Pokud je, jsou mu odesláný příslušné zpráv.

Logování zpráv

V případě, že zpráva je typu *PRIVMSG* nebo *NOTICE* a text zprávy obsahuje alespoň jedno z klíčových slov, je zpráva zalogována v metodě `logMessage()`, která zprávu zaloguje v příslušném formátu protokolu *SYSLOG*, za pomocí UDP klienta.

Spuštění bota a ošetření vyjímek

Instance bota je vytvářena v souboru main.cpp, v bloku `try, catch`. Následně je bot spuštěn pomocí metody `start()`. Vznikne-li během běhu programu jakákoliv vyjímka, je odchycena a obsah její zprávy je vypsán na *STDERR* a program je náležitě ukončen.

Závěr

Veškerá požadovaná funkcionalita bota byla implementována a program by tedy měl fungovat bez jakýchkoliv omezení.