VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Dokumentácia k projektu do predmetu ISA
Klient SIP

15.11.2015

Autor: Marek Marušic, xmarus05@stud.fit.vutbr.cz
Fakulta Informačních Technologií
Vysoké Učení Technické v Brně

Obsah

1 Úvod

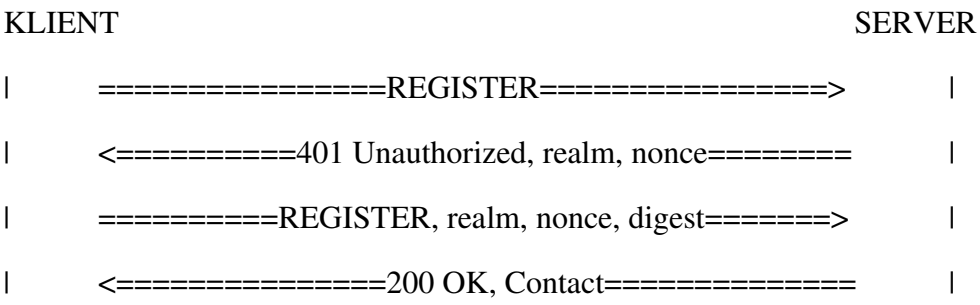
Tento dokument vznikol ako dokumentácia k projektu do predmetu ISA (Sieťové aplikácie a správa sietí) s názvom Klient SIP. Popisuje ako funguje SIP protokol a následne sa zaoberá implementáciou a používaním vytvorenej aplikácie.

2 Protokol SIP

Každý paket obsahuje request line alebo response line. Request line obsahuje metódu adresu kam paket má byť poslaný a verziu protokolu. Response line obashuje verziu protokolu, kód odpovede a popis odpovede. Za týmto riadkom nasleduje telo paketu.

2.1 Registrácia pomocou SIP

Priebeh registrácie na server môžeme vidieť na obrázku č.1.

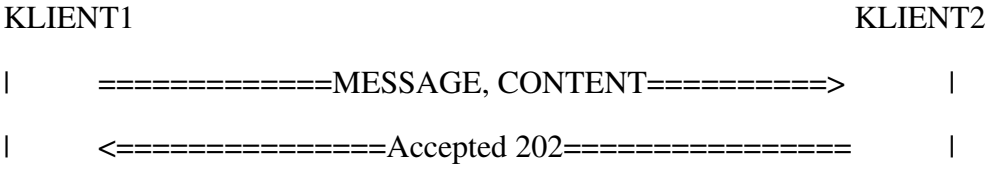


Obrázok č.1

2.2 Odosielanie a príjmanie správ

Klienti komunikujú priamo bez nutnosti komunikácie cez server, pomocou IP adries. Taktiež môžu komunikovať cez server, pokial klienti nepoznajú svoje IP adresy.

Priebeh odosielania a príjmania správ môžeme vidieť na obrázku č.2.



Obrázok č.2

3 Návrh programu

Klient SIP sa musí po spustení pokúsit o prihlásenie na server. Pokial server neodpovedá alebo hlási chybu serveru, klient sa pokúša o prihlásenie znova každých 5 sekúnd. Po 10 opakovaní tohoto scenáru klient prestane s registráciou a dalej pošle správy špecifikované v súbore messages.txt. Po odoslaní správ klient čaká na správy od iných klientov, a po prijatí správy odosiela správnu odpoved o tom že prijal správu. Klient sa ukončí pomocou SIGINT, SIGTERM alebo SIGQUIT. Pri ukončovaní klienta sa zistuje či je zaregistrovaný na server. V prípade, že je klient zaregistrovaný na server, sa klient odregistruje a následne sa správne ukončí.

4 Implementácia

Aplikácia je písana v jazyku C++. Klient SIP podporuje iba transportný protokol UDP. Pre sieťovú komunikáciu používa BSD sockety. Implementácia výpočtu MD5 digest hesla pre autentizáciu klienta je realizovaná pomocou knižnice OpenSSL. Všetky príjmané pakety sú spracované pomocou funkcie *processPacket()*. Pre odosielanie a príjmanie paketov je nastavený timeout 5 sekúnd.

4.1 Vstup

Klient spracuje zadané parametre a ich hodnoty. Parameter -p specifikuje meno súboru, v ktorom s nachádzajú údaje pre registráciu na server. Údaje potrebné pre registráciu sú nasledovné. Server je doména alebo IP serveru s volitelným portom. Username a password obsahuje užívateľské meno a heslo používané pre prihlásenie. Expires značí dobu, ako dlho má registrácia platiť. Kedže je nutné po registrácii poslať správy z klienta, mohlo by sa stať, že počas čakania na odpoveď vyprší registrácia, preto pokial je zadaná hodnota expires menšia ako minimálna hodnota[1] nastaví sa na túto minimálnu hodnotu.

Tento súbor sa spracuje a získané informácie sa uložia do štruktúry *s_options*. Parameter -m špecifikuje meno súboru, v ktorom sa nachádza zoznam príjemcov a správ. Súbor má dva stĺpce oddelené medzerou, v prvom stĺpci sa nachádza adresa príjemcu a v druhom správa pre prijemcu. Získané informácie zo súboru sa uložia do štruktúry typu *s_message.* Timeout je nastavený na 5 sekúnd.

1 Minimálna hodnota expires = 3*timeout príjmania paketu. Je zvolená tak, aby chyby pri príjmaní a odosielaní správ neosposobili vypršanie registrácie klienta.

4.2 Registrácia

Registráciu zabezpečuje funkcia *registerSip()*, ktorá zabezpečuje odoslanie a spracovanie prijatých registračných paketov, potrebných k registrácii na server.

4.2.1 Registračný paket č.1

Klient SIP vygeneruje pomocou funkcie g*enSIPRegisterPacketStruct()* vygeneruje základný registračný paket a odošle ho na server. Tento paket obsahuje v request line REGISTER sip:uri protokol.

4.2.2 Odpoveď na registračný paket č.1

Na tento paket klient dostane odpoveď s kódom 401 Unauthorized a v riadku WWW-Authenticate nám server oznámi, akým šifrovacím algoritmom treba šifrovat posielané prihlasovacie údaje klienta. V rovnakom riadku nájdeme políčko nonce, ktoré je unikátne a používa sa pri šifrácii prihlasovacích údajov. Tento paket spracuje funkcia *processPacket().*

4.2.3 Registračný paket č.2

S informáciami, ktoré nám server poslal v riadku WWW-Authenticate vygenerujeme rovnaký registračný paket ako v 1. kroku, ktorý navyše obsahuje riadok Authorization s potrebnými údajmi a odpovedou, ktorá obsahuje zašifrované heslo. Túto odpoved zašifrujeme pomocou funkcie *MD5()* z kniznice OpenSSL.

4.2.4 Odpoveď na registračný paket č.1

Po zaslaní správneho registračného paketu č.2 nám server odpovie paketom s kódom 200 OK. Pri zadaní nesprávneho hesla server odpovedá s kódom 403 Forbidden. Po spracovaní tohoto paketu klient vypíše na štandardný výstup správu s daným kódom a IP zdroja a destnácie.

4.3 Správy

Klient sa pokúsi zaslať všetky správy pomocou funkcie *sendMessages()* a po odoslaní všetkých správ sa prepne do režimu príjmania správ pomocou funkcie *service().*

4.3.1 Odosielanie správ

Odosieľanie správ má na starosti funkcia *sendMessages()*, ktorá v parametre dostane vektor so správami a príjmateľmi správ. Všetky správy v tomto sa postupne pripravia ako pakety, a následne sa

odošlú tieto pakety. Po odslaní klient čaká na odpoveď s kódom 202 Accepted od príjmateľa. Pokiaľ príde potvrdenie správy alebo do 5 sekúnd príjmateľ neodpovie, klient pokračuje odosielaním ďalších správ.

4.3.2 Príjmanie správ

Príjmanie správ má na starosti funkcia *service()*, ktorá po prijatí správy zistí funkciou *isMessageForMe()* či je klient správny príjemca správy. Po tejto kontrole sa správa spracuje pomocou funkcie *processPacket()* a vypíše sa na štandardný výstup.Ďalej sa odošle paket s kódom 202 Accepted ako odpoveď. Klien čaká na prijatie správy 5 sekúnd, pokial žiadnu správu nedostane reštartuje príjmanie správy.

4.3 Udržiavanie spojenia so serverom

Funkcia *service()*, má okrem príjmania správ na starosti aj udržiavanie užívateľa zaregistrovaného na server. V prípade že sa užívateľovi podarilo zaregistrovať na server, funkcia dáva pozor či zostávajúci čas prihlásenia je dostatočne dlhý. Po presiahnutí hranice minimálneho zostávajúceho času (10sekúnd) klient znova podstúpi registračnú procedúru čím predĺži čas svojej registrácie. Minimálny zostavajúci čas je zvolený, tak aby chyba pri príjmaní a odosielaní paketu neosposobila vypršanie registrácie klienta.

5. Použitie aplikácie

Program počúva prichádzajúce pakety na porte 5060.

Parametre spúštania programu:

-h, --help

Vypíše nápovedu.

-p options.txt

Špecifikuje meno súboru, v ktorom sa nachádzajú údaje potrebné na pripojenie

k SIP serveru.

-m messages.txt

Špecifikuje meno súboru, v ktorom sa nachádzajú údaje potrebné pre odoslanie správ.

Súbor musí mať nasledujúci formát:

meno@server:port TELO SPRAVY.

6 Záver

Aplikácia pomocou SIP protokolu sa dokáže zaregistrovať na SIP server a taktiež posielať a príjmať správy protokolu SIP. Aplikácia je prekladaná pomocou g++ a používa štandardy C++11. Testovanie prebiehalo pomocou predpripravených virtuálnych obrazov systému Ubuntu.

Referencie

RFC3261 - SIP: Session Initiation Protocol (http://tools.ietf.org/html/rfc3261)

RFC3428 - SIP: Extension for Instant Messaging (http://www.tools.ietf.org/html/rfc3428)

MD5 autentizace protokolu SIP (http://www.kapejod.org/en/2013/02/reversing-sip-digest-authentication-in-javascript-node-js)