UNIVERZA V LJUBLJANI

FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Siniša Grubor

Oddaljen dostop do namiznega računalnika

DIPLOMSKO DELO

UNIVERZITETNI ŠTUDIJSKI PROGRAM PRVE STOPNJE

RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: doc. dr. Tomaž Dobravec

Ljubljana 2013

Besedilo je oblikovano z urejevalnikom besedil MS Word.

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 01894/2013
Datum: 11.01.2013

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **SINIŠA GRUBOR**

Naslov: **ODDALJEN DOSTOP DO NAMIZNEGA RAČUNALNIKA**
REMOTE CONTROL FOR DESKTOP COMPUTERS

Vrsta naloge: Diplomsko delo univerzitetnega študija

Tematika naloge:

Program za oddaljen dostop do namiznega računalnika uporabniku omogoča uporabo računalnika, ki se nahaja na oddaljeni lokaciji. Tovrstni programi so zelo priljubljeni, saj omogočajo vzdrževanje, odpravljanje težav in uporabo vseh nameščenih programov na gostiteljskem računalniku.
Preglejte obstoječe programe za oddaljen dostop in jih med seboj primerjajte. Opišite prednosti in slabosti posameznega izdelka.
Izdelajte tudi svoj program za oddaljen dostop. Program naj omogoča povezavo dveh računalnikov na dva načina: preko direktne povezave in preko dodatnega strežnika. V delu predstavite prednosti in slabosti obeh pristopov. Raziščite tudi različne možnosti za prenos slike, jih implementirajte in med seboj primerjajte.

Mentor:

doc. dr. Tomaž Dobravec

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

IZJAVA O AVTORSTVU DIPLOMSKEGA DELA

Spodaj podpisani Siniša Grubor, z vpisno številko 63070106, sem avtor diplomskega dela z naslovom:

Oddaljen dostop do namiznega računalnika

S svojim podpisom zagotavljam, da:

- Sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Tomaža Dobravca,
- So elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela,
- Soglašam z javno objavo elektronske oblike diplomskega dela v zbirki »Dela FRI«.

V Ljubljani, dne 28. junija 2013 Podpis avtorja:

Zahvala

Iskreno se zahvaljujem doc. dr. Tomažu Dobravcu za vso izkazano pomoč pri izdelavi diplomske naloge.

Kazalo

Povzetek

Namen diplomske naloge je samostojna implementacija programa za oddaljen dostop do namiznega računalnika, ki bo deloval stabilno in ga ne bosta ovirala požarni zid ali usmerjevalnik.

V prvem delu bomo opisali splošno stanje programov za oddaljen dostop do namiznega računalnika, ki so dostopni na tržišču, in naredili krajšo primerjavo delovanja ter opis prednosti in slabosti le-teh. V drugem delu diplome se bomo osredotočili na izdelavo samostojnega programa za oddaljen dostop do namiznega računalnika. Natančno bomo opisali princip delovanja, kot tudi posamične module, ki so pomembni za njegovo delovanje. Pojasnili bomo tudi uporabo skozi grafični vmesnik, ki prav vsakemu uporabniku omogoča enostavno uporabo. V zaključku bomo naredili primerjavo našega programa z ostalimi, ki so na tržišču, in opisali, kje je še ostal prostor za njegove morebitne izboljšave.

Abstract

Our goal was to implement our own software for remote computer access. Our key goals were that software must be reliable and while using it, users will not have to change any firewall or router settings.

In first part, we are going to analyse current available programs for remote access. We will look at their pros and cons and try to determine, which of them are the best for the user.

In second part, we will focus on building our own program. We will describe in detail how it works and the main architecture of the program. Also we will look at the graphical user interface and how a user should use our software.

In conclusion, we will compare our product with current products on the market and also see if there is any room for improvements.

Ključne besede

Oddaljen dostop, primerjava slik, povezava med računalnikoma

Key words

Remote access, remote desktop, image comparison

Poglavje 1

1.1 Uvod

V diplomskem delu je predstavljena izdelava programa, ki omogoča oddaljen dostop do namiznega računalnika. Njegov osnovni namen je popoln nadzor uporabnika nad oddaljenim računalnikom.

Za to temo smo se odločili, ker se nam je zdelo zanimivo poskusiti implementirati svojo različico programa za oddaljen dostop do računalnika. Osnovna vodila, ki smo jih upoštevali pri sami implementaciji, so delovanje programa na vseh operacijskih sistemih, delovanje miške in tipkovnice ter delovanje v realnem času. Kot dober primer delovanja smo vzeli že obstoječi program TeamViewer in se poskušali z našo implementacijo čimbolj približati njegovemu delovanju. TeamViewer smo izbrali zato, ker je eden od vodilnih programov na tržišču in ima poseben način delovanja, pri katerem povezava med dvema računalnikoma ni neposredna, temveč poteka skozi vmesni strežnik. S tem se izognemo vsem problemom delovanja za požarnim zidom in usmerjevalniki. S tem se močno izboljša uporabniška izkušnja, saj povprečen uporabnik običajno nima dovolj znanja, da bi sam nastavil vse potrebne stvari za delovanje takšnih programov.

Zadane cilje nam je uspelo izpolniti, zato se lahko pohvalimo s kvalitetnim izdelkom, ki deluje s solidno hitrostjo in omogoča uporabo večine funkcionalnosti, ki jih omogočajo tudi ostali programi z enakim namenom. Program je implementiran v programskem jeziku Java in deluje enako dobro na vseh treh glavnih operacijskih sistemih, torej na Windowsih, Linuxu in OS X-u.

Poglavje 2

2.1 Obstoječi programi za oddaljen dostop do računalnika

Na trgu obstaja precej programov za oddaljen dostop do računalnika. Sodeč po članku, objavljenem na Wikipediji, obstaja prek 50 različic programa [5]. Ker vseh ne moremo testirati in analizirati, smo se odločili, da bomo glede na anketo, objavljeno na spletni stani lifehacker.com [1] , izbrali pet najbolj priljubljenih programov in jih med seboj primerjali. Anketo je do zaprtja teme rešilo kar 7978 uporabnikov, kar je dovolj velik vzorec, da lahko sklepamo na verodostojnost vprašalnika.

Programi, ki so bili izbrani, so UltraVNC, Windows Remote Desktop Connection, TeamViewer, TightVNC in LogMeIn. Prvo zanimivo dejstvo, ki ob pogledu na rezultate ankete vzbudi pozornost, je, da kar 89% uporabnikov uporablja enega od zgoraj naštetih programov za oddaljen dostop do računalnika. To potrjuje, da je naša izbira programov verjetno dobra.

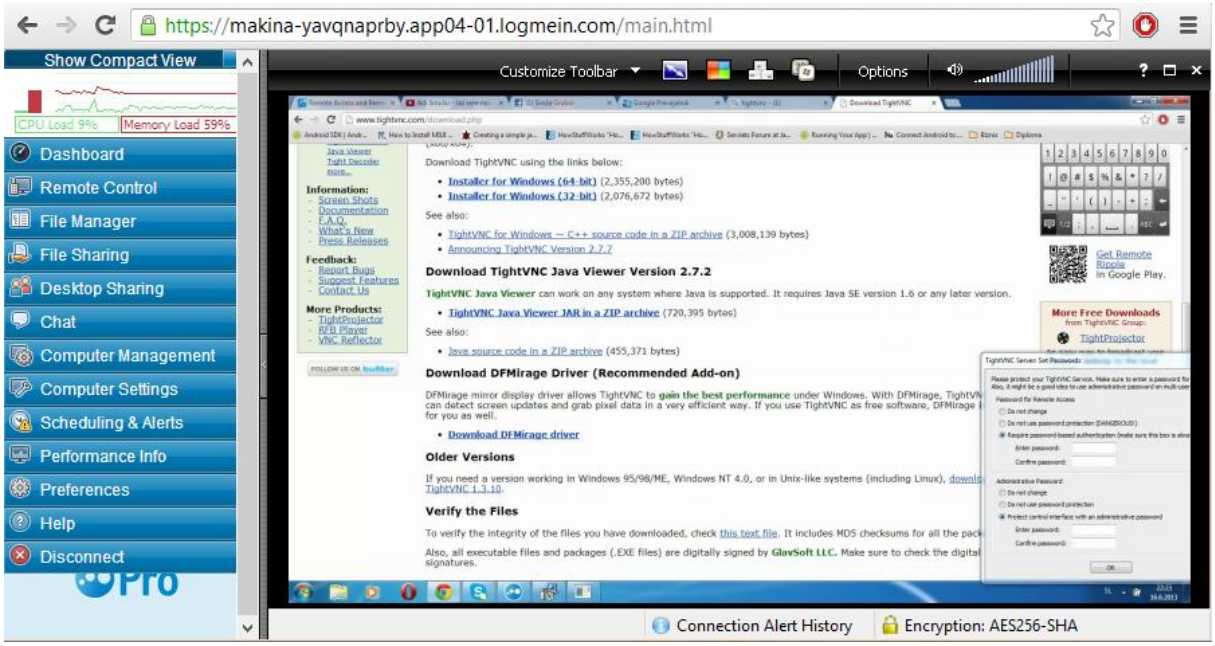
Druga zanimiva stvar je, da je bil od teh petih programov naš izbrani model TeamViewer uvrščen najslabše. Uporablja ga namreč zgolj 8.7% ljudi. Najbolje uvrščeni program je LogMeIn, ki ga uporablja kar 26.64% anketirancev. LogMeInu tesno sledi Windows Remote Desktop Connection, ki ga je izbralo 24.08% uporabnikov, za njim pa sta se znašla UltraVNC s 15.56% in TightVNC s 14.5% uporabniki.

Kot lahko vidimo, so razlike med posamičnimi programi na prvi pogled precejšnje. Zato si bomo ogledali vsak program posebej in ugotovili, kaj so slabosti in prednosti posameznih programov.

2.2 LogMeIn

LogMeIn je program, ki ga je izdelalo podjetje LogMeIn Inc. Dostopen je na njihovi spletni strani [3], kjer ponujajo dve različici programa – plačljivo in brezplačno. Mi smo preizkusili slednjo. Razlika med verzijama je v tem, da pro verzija ponuja nekaj dodatnih funkcionalnosti, kot so oddaljeno tiskanje, prenos datotek, prenos zvoka, deljenje namizja in tehnično podporo.

LogMeIn deluje po načelu, da je računalnik, na katerega namestiš program, preko spletnega vmesnika dostopen kjerkoli. Zaradi tega je program za uporabnika zelo prikladen. Ko ga enkrat namesti na svoj računalnik, lahko do njega dostopa od praktično kjerkoli, edini pogoj je povezava s spletom in poljuben brskalnik. Pa gre res za poljuben brskalnik? Sodeč po dokumentaciji na njihovi spletni strani, program podpira brskalnik Internet Explorer verzija 7.0+, Firefox 3.0+, Chrome 2.0+ ter Safari 4.0+. Glede na to mislimo, da je uporaba izraza »poljuben brskalnik« v tem primeru dovoljena, saj večina uporabnikov uporablja enega od navedenih brskalnikov, prav tako pa je podpora preteklih verzij brskalnikov zadostna.



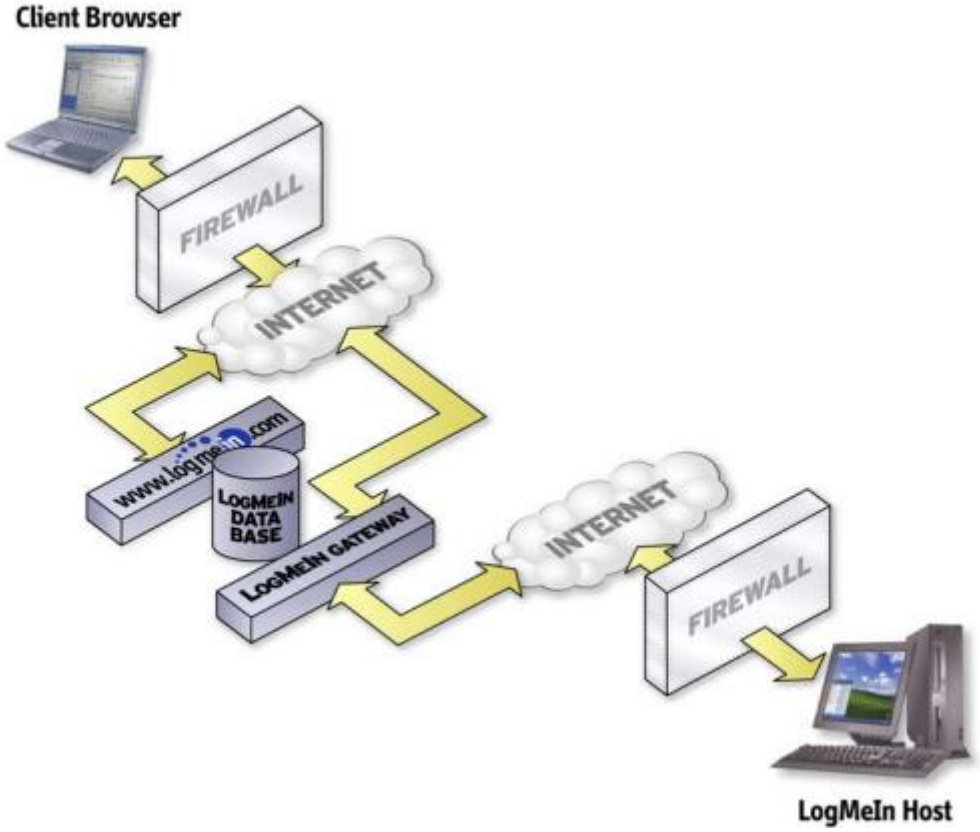
Slika 1: Prikaz delovanja LogMeIn programa v brskalniku Chrome

Gre za zelo dobro implementacijo programa za oddaljen dostop do računalnika, ki omogoča tudi več sej hkrati, kar pomeni, da se lahko več uporabnikov hkrati poveže na sistem. Vendar ima tudi LogMeIn nekaj pomanjkljivosti. Za najboljšo uporabniško izkušnjo mora uporabnik namestiti dodatek k brskalniku, ki ga uporablja. Skozi samo namestitev te sicer vodijo navodila, ki so objavljena na spletni strani, a kljub temu stvar ni najbolj preprosta za povprečnega uporabnika. Za naš test smo namestili dodatek brskalniku Google Chrome v. 27.0.1453.110, a je brskalnik avtomatsko namestitev zavrnil, ker je dodatek prepoznal kot škodljivo programsko opremo. Zato mora uporabnik dodatek namestiti ročno, kar za marsikoga lahko predstavlja nevšečnost. Seveda obstaja alternativna izbira, da dodatka ne namestimo. V tem primeru LogMeIn uporabi Javo za prikazovanje slike, kar sicer deluje, vendar izredno počasi in mnogo slabše kot z nameščenim dodatkom.

Poleg opisane ima LogMeIn eno zelo veliko pomanjkljivost, in sicer to, da z njim ne moremo dostopati do računalnikov, na katerih imamo nameščen operacijski sistem Linux. Glede na to,

da se danes veliko spletnih strežnikov poganja na Linuxu, lahko rečemo, da gre za veliko slabost sicer odlično izdelanega programa.

Po splošnem pregledu si oglejmo še tehnično plat programa. Slika 2 prikazuje njegovo arhitekturo.



Slika 2: Prikaz arhitekture LogMeIn programa [2]

Kot je razvidno iz slike 2, ni vzpostavljena direktna povezava med dvema računalnikoma, temveč je med računalniki strežnik. Taka arhitektura nam olajša skrbi glede požarnega zidu, saj ta ne bo blokiral oddaljene povezave, vendar se hkrati pojavi pomislek glede varnosti, saj gre naš celoten promet skozi tuj strežnik. Je torej uporaba LogMeIn programa res varna za uporabnika?

V dokumentaciji navajajo, da je največja varnostna luknja celotnega sistema uporabnik sam, s čimer se seveda strinjamo. Nato pojasnijo, da so njihove storitve za uporabnika povsem varne, saj jih vodi pet varnostnih ciljev:

1. overjanje ciljnega računalnika;
2. overjanje uporabnika, ki se poveže na njihov strežnik;
3. overjanje strežnika ciljnemu računalniku;
4. overjanje ciljnega računalnika strežniku;
5. enkripcija podatkov.

Več podrobnosti o zaščiti lahko najdemo v dokumentaciji [2]. Hiter povzetek petih točk pokaže, da gre za overjanje vseh vpletenih enot v proces na osnovi SSL certifikatov. Na ta način se lahko tudi enkriptira podatke, zato lahko sklenemo, da je uporaba LogMeIn storitev načeloma varna [13]. Vseeno pa nekaj pomislekov ostaja. Te so pojasnjeni v članku »The Most Dangerous Code in the World: Validating SSL Certificates in Non-Browser Software« [12], kjer avtorji članka pojasnijo, da lahko z izbiro neprimernih knjižnic ali z njihovo napačno uporabo ogrozimo našo varnost.

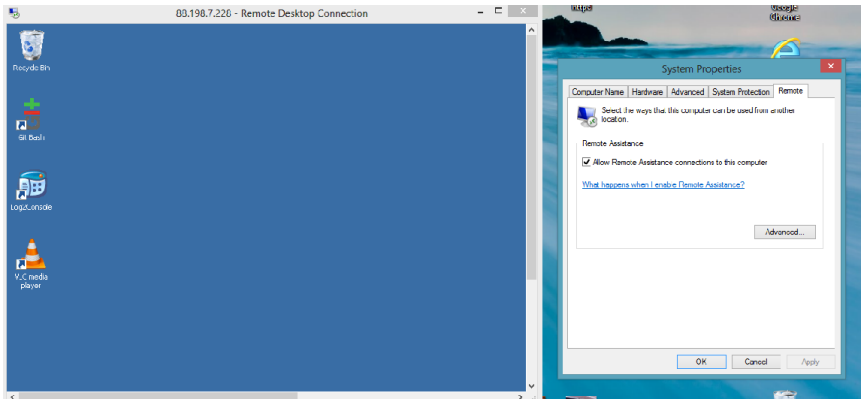
Gre torej za odličen program, ki si vsekakor zasluži mesto na lestvici najboljših petih programov za oddaljen dostop do računalnika. In vendar je njegova uporabnost nekoliko slabša, ker ne podpira oddaljenega dostopa do Linux sistemov.

2.2 Windows Remote Desktop Connection

Od vseh petih verjetno najbolj znan program je Windows Remote Desktop Connection. Program je avtomatsko vgrajen v večino različic Windowsov. Prvi Windowsi, ki so ga ponujali, so bili Windows XP Professional. Od takrat so ga z izjemo Home različic obdržali v vseh drugih, torej tudi v Windows 8 Enterprise, Windows 8 Pro, Windows 7 Professional, Windows 7 Enterprise, Windows 7 Ultimate, Windows Vista Business, Windows Vista Ultimate in Windows Vista Enterprise.

Ker je program zelo poznan, bomo le na kratko razložili njegove prednosti in slabosti. Ena glavnih prednosti je ta, da ga imamo zelo verjetno prednaloženega na računalniku. Tudi samo delovanje je hitro in enostavno. Kljub temu ima program obilico pomanjkljivosti. Omogoča nam zgolj povezavo med Windows sistemi, ne omogoča več sej hkrati, potrebne so dodatne nastavitve za uporabo znotraj lokalnega omrežja. Prav tako mora uporabnik sam poskrbeti za izjemo v požarnem zidu.

Omenjene pomanjkljivosti so pomembno dejstvo, zato menimo, da ni ravno najboljša izbira za navadnega uporabnika, saj se ob vseh nastavitvah, ki jih je potrebno nastaviti, ta lahko večkrat zmede. Res pa je, da je progam zelo dobro dokumentiran in tako lahko uporabnik hitro najde vsa navodila na spletu. Priporočili bi ga za administracijo oddaljenih Windows strežnikov, saj je zelo dobro integriran v operacijski sistem.



Slika 3: Prikaz delovanja programa Windows Remote Desktop Connection

2.4 UltraVNC

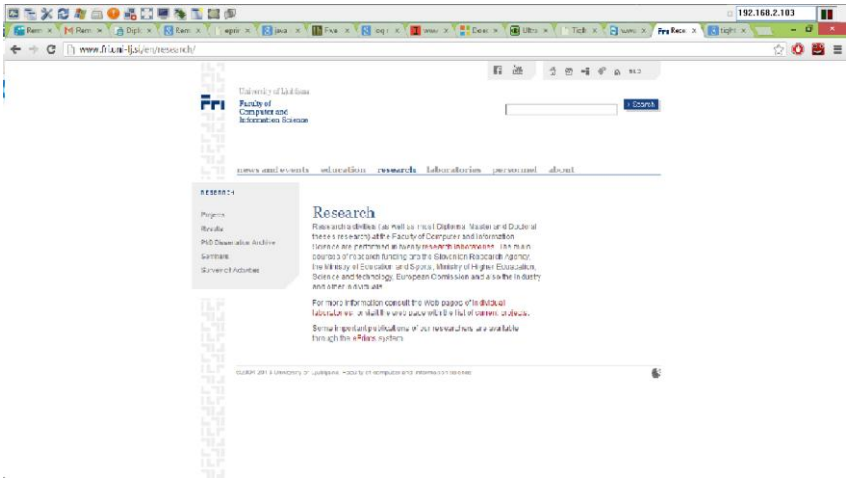
UltraVNC program je nastal leta 2005 pod GPL licenco. Torej gre za program, ki je popolnoma brezplačen. To je jasno že ob prihodu na njihovo spletno stran [4], ki z estetskega vidika ni ravno najlepša, v njen prid pa je potrebno omeniti, da ima zelo dobro dokumentacijo.

Princip delovanja, ki so ga zasnovali, je neposredna povezava med dvema računalnikoma. Uporabnik ima ob namestitvi možnost izbire, ali želi namestiti zgolj strežnik ali zgolj opazovalnik. Seveda lahko namesti tudi oboje.

Zaradi direktne povezave med računalnikoma, je varnost verjetno manj diskutabilna kot pri LogMeIn programu, a to hkrati pomeni, da je potrebno vložiti veliko več časa v nastavljanje programa. Delovanje za požarnim zidom in usmerjevalnikom pride z dodatnimi navodili, ki jih je potrebno upoštevati.

Program podpira delovanje zgolj na Windows sistemih, kar je še ena velika pomanjkljivost. Tudi sama izkušnja se nam ne zdi ravno prijazna do povprečnega uporabnika, saj se mora ta spoprijeti s široko izbiro nastavitvenih možnosti, ki niso vedno jasno razložene.

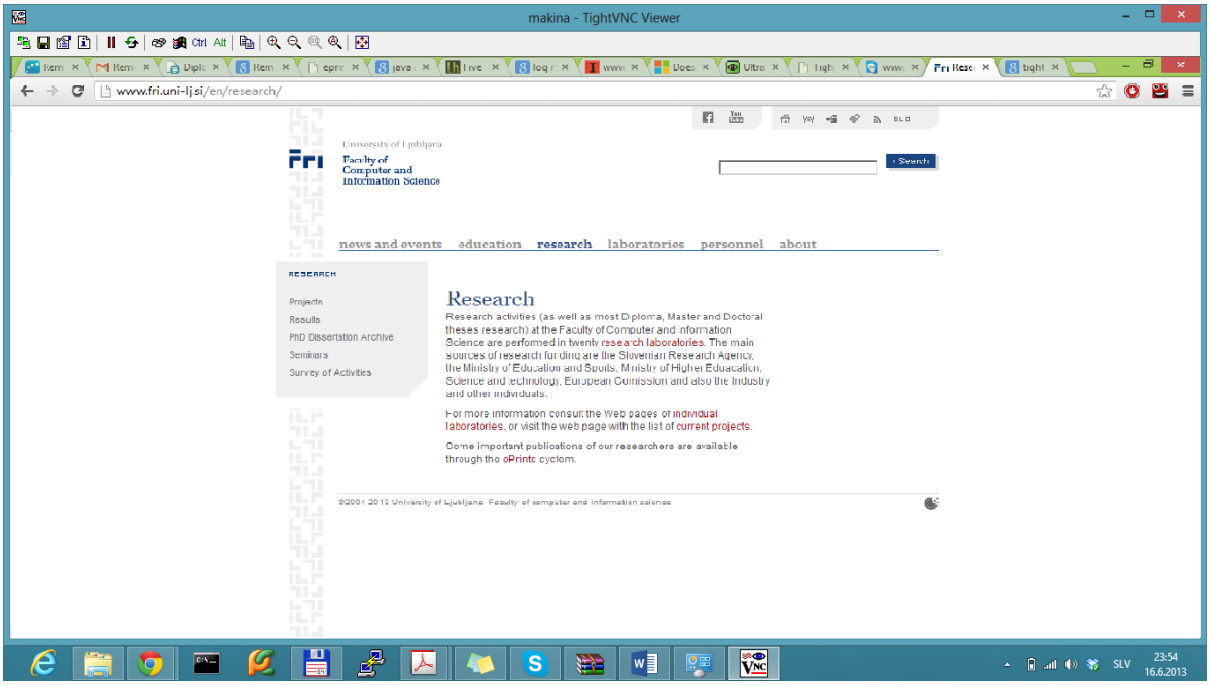
Sam prenos slike je sicer kvaliteten, a precej počasnejši kot pri prvih dveh programih.



Slika 4: Prikaz delovanja programa UltraVNC

2.5 TightVNC

TightVNC je program, ki ga je razvil Constantin Kaplinsky leta 2001 [6]. Gre za program, ki deluje po podobnem principu kot UltraVNC, zato ni presenetljivo, da je razlika v oceni med njima precej majhna. Oba uporabljata RFB (Remote frame buffer) protokol, treba pa je izpostaviti, da TightVNC za razliko od UltraVNC podpira tudi Linux sisteme. Oba programa sta za uporabnika zelo zapletena in ob nameščanju zahtevata kar nekaj znanja. Tako kot UltraVNC ima tudi TightVNC GPL licenco. Sam prenos slike je sprejemljiv, a vidno slabši od prvih dveh opisanih programov.



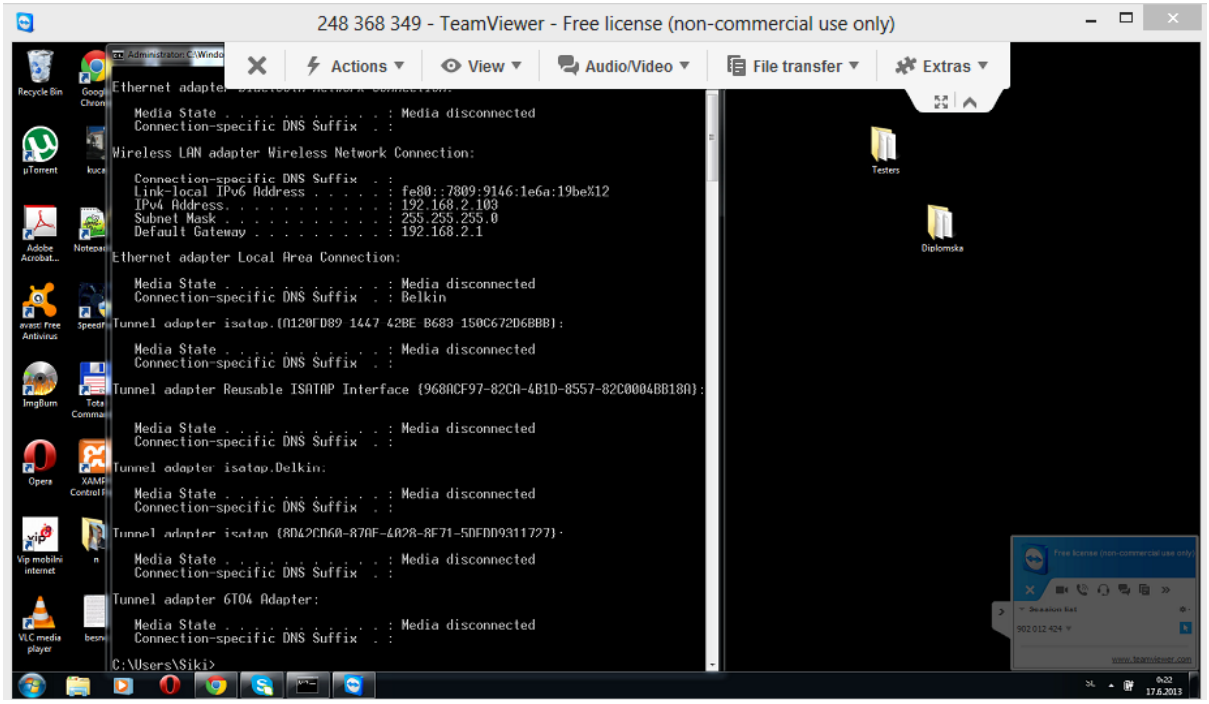
Slika 5: Prikaz delovanja programa TightVNC

2.6 TeamViewer

TeamViewer je odličen program, ki ga je leta 2005 razvilo podjetje TeamViewer GmbH. Do uporabnika zelo prijazen, saj je enostaven za uporabo. Ob zagonu se uporabniku odpre nezapleten uporabniški vmesnik z geslom in uporabniškim imenom. Vsak uporabnik dobi enotno identifikacijsko številko, po kateri ga TeamViewerjev strežnik prepozna. Za povezavo z drugim računalnikom je tako potrebno imeti zgolj ti dve informaciji.

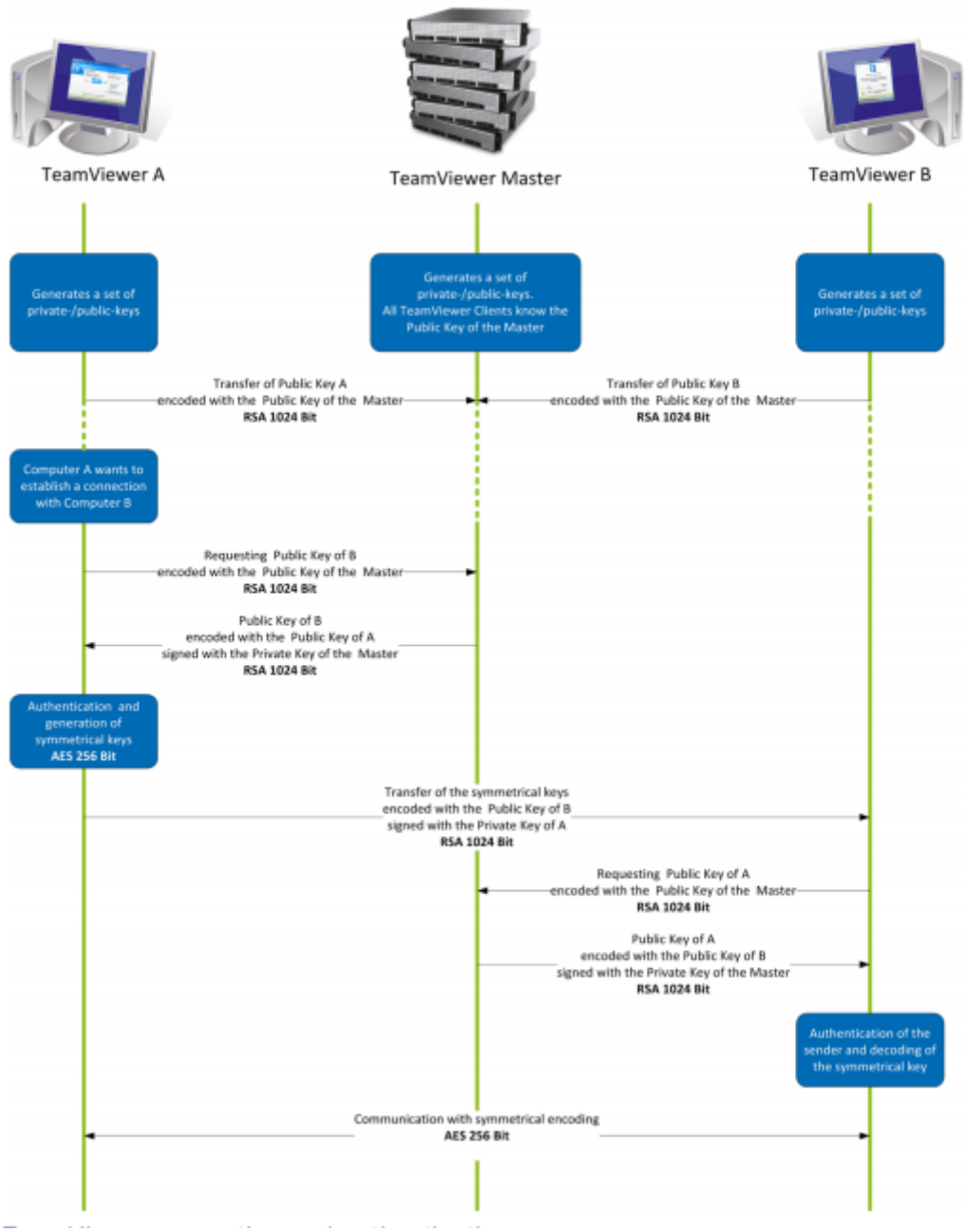
Podobno kot LogMeIn, tudi TeamViewer deluje preko njihovega strežnika, kar naredi program enostavnejši za uporabo, saj se uporabniku ni potrebno ukvarjati z nastavitvami

usmerjevalnika in požarnega zidu. Program omogoča tudi uporabo na vseh treh glavnih operacijskih sistemih. Samo delovanje in prenos slike sta zelo hitra in zanesljiva.



Slika 6: Prikaz delovanja programa TeamViewer

Prav tako je poskrbljeno tudi za varnost, saj so vse povezave kriptirane. Za več informacij o varnosti si lahko ogledamo sliko 7 ali preberemo dokumentacijo na njihovi spletni strani [6,7].



Slika 7: Prikaz enkripcije in delovanja programa TeamViewer [7]

Po našem mnenju je močno presenetljivo, da se je tako kvaliteten program po mnenju uporabnikov znašel na repu lestvice. Možen razlog za tak rezultat je, da je bila anketa izvedena leta 2008 [1], ko TeamViewer še ni bil močno razširjen na trgu. Verjamemo namreč, da bi bil danes rezultat raziskave bolj kot ne enak, le da bi se TeamViewer zavihtel s četrtega na prvo mesto. Zaradi vsega povedanega menimo, da je ta program prava referenca za primerjavo z našim in da je smiselno upoštevati tudi dobra vodila, ki so si jih zastavili razvijalci TeamViewerja. Najpomembnejše vodilo je implementacija povezave s strežnikom med dvema računalnikoma.

Poglavje 3

3.1 Razvoj programa

V tem poglavju bomo natančno predstavili notranjo zgradbo programa in komponente, ki ga sestavljajo. Opisali bomo implementacijo nekaterih komponent in tudi orodja ter tehnologije, ki smo jih uporabili pri načrtovanju in implementaciji.

3.2 Razvojna orodja

NetBeans je integrirano razvojno okolje (IDE), ki je namenjeno razvoju programov. Uporabljamo ga lahko za razvoj upravljalnih aplikacij, grafičnih vmesnikov, spletnih strani, spletnih storitev in mobilnih aplikacij. NetBeans omogoča programiranje v večih programskih jezikih. Za razvoj našega programa smo ga uporabili, ker močno olajša razvijanje Javanskih aplikacij. Olajša predvsem implementacijo uporabniškega grafičnega vmesnika, saj NetBeans z enostavnim »drag and drop« pristopom omogoča, da grafični vmesnik hitro in enostavno zgradimo.

Glavni elementi razvojnega okolja so sledeči:

3.2.1 Urejevalnik kode

Napreden urejevalnik kode z vgrajeno tehnologijo Intel-liSense, ki nam med programiranjem prikaže razrede in metode, ki jih imamo na voljo. Preurejevalnik kode (angl. Refactoring) omogoča izboljšavo berljivosti in strukturo kode.

3.2.2 Izdelava grafičnih vmesnikov

Enostavno izdelovanje zaslonskih vmesnikov po principu WYSIWYG (kar vidiš, to dobiš).

3.2.3 Razhroščevalnik

Ena izmed najbolj pomembnih lastnosti NetBeansa je možnost izvajanja sprehajanja po aplikaciji vrstico za vrstico. Tako hitreje odpravimo napake v kodi.

3.2.4 Organiziranost

Organizacija kode v projekte.

Ko želimo program dobro strukturirati, je uporaba integriranih razvojnih orodij močno priporočljiva.

3.3 Pomembni Javanski razredi

Ker smo želeli, da naš program na vseh operacijskih sistemih deluje enako, smo se odločili, da ga napišemo v Javi. V tem poglavju si bomo na hitro ogledali pomembnejše uporabljene razrede.

3.3.1 Socket

Ta razred nudi implementacijo t.i. »client« vtičnika. Z drugimi besedami to pomeni, da je to vstopna oziroma izstopna točka za komunikacijo med dvema računalnikoma [9].

3.3.2 ServerSocket

Ta razred nam nudi implementacijo t.i. »server« vtičnika. Ta vtičnik čaka na zahtevo »clienta« in ko jo prejme, izvede zahtevano akcijo in običajno vrne rezultat le-te nazaj. [10]

3.3.3 Robot

Razred robot je del zbirke awt (Abstract Window Toolkit). Uporablja se za generiranje vhodnih dogodkov sistema. Običajno se uporablja povsod, kjer potrebujemo generiranje pritiskov tipkovnice ali miške. Tako lahko, ko ustvarimo objekt robot, dejansko sprožimo pritisk na levi ter desni miškin gumb, vseh tipk na tipkovnici, drsenje (scroll) miške, kot tudi zajemamo trenutno sliko ekrana [11]. Gre za najpomembnejšega od vseh razredov, ki so bili uporabljeni v implementaciji programa.

Seveda je bilo uporabljenih še veliko več razredov, vendar so omembe vredni predvsem ti trije. Prva dva nam povesta, da povezava med računalniki poteka po zanesljivem TCP/IP protokolu, razred Robot pa nam razkriva, kako se lahko v Javi avtomatsko generira uporabniške poteze.

3.4 Implementacija

Zgradbo programa lahko razdelimo na tri dele:

- Strežnik
- Odjemalec
- Gostitelj

V naslednjih poglavjih si bomo pogledali zgradbo vsakega od teh treh delov.

3.4.1 Strežnik

Strežnik je samostojni del programa, skozi katerega poteka komunikacija med dvema računalnikoma. Prednost uporabe strežnika za komunikacijo med dvema računalnikoma je razložena že v prejšnjih poglavjih, a vseeno jo na kratko ponovimo. Z uporabo strežnika se izognemo situaciji, v kateri bi moral eden od računalnikov, ki med seboj komunicirata, delovati kot strežnik. Zaradi tega nam požarni zid ne blokira naše povezave in tudi dodatna konfiguracija usmerjevalnika ni potrebna.

S tem uporabniku omogočimo takojšnjo uporabo našega programa, ki je primeren za vse vrste uporabnikov in ne zgolj za tiste s tehničnim predznanjem. Ker pa še vedno želimo ohraniti delno prilagodljivost programa, ima uporabnik možnost izbire in se lahko odloči, da na primer ne želi vzpostaviti povezave preko strežnika, ampak neposredno med dvema računalnikoma. Kasneje bomo tudi pokazali, kako trivialno je to omogočiti ali ne.

Za potrebe našega programa smo torej implementirali enostaven Javanski strežnik, ki je zmožen procesirati neomejeno število povezav in usmerjati promet med dvema računalnikoma.

3.4.1.1 Delovanje

Strežnik deluje tako, da nenehno teče in posluša na poljubno izbranih vratih. Ko uporabnik poda zahtevo za povezavo, se ustvari nova nit, ki poveže dva uporabnika med seboj in skrbi za njuno komunikacijo. Ob prekinitvi povezave se nit ustavi in računalniki, vpleteni v povezavo, se izbrišejo iz strežnikovega spomina.

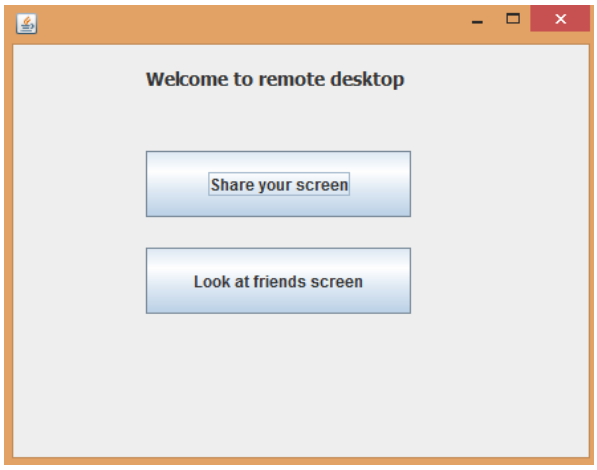
Strežnik lahko teče na poljubnem računalniku, na poljubih vratih. Edina zahteva je, da je dosegljiv z zunanjega spleta. Z drugimi besedami, če je strežnik povezan na usmerjevalnik, moramo poskrbeti, da usmerjevalnik posreduje promet do strežnika. Prav tako moramo dodati izjemo v požarni zid.

Za konfiguracijo strežnika poskrbimo razvijalci programa, tako da za končnega uporabnika to dejansko ni pomembno.

3.4.1.2 Odjemalec

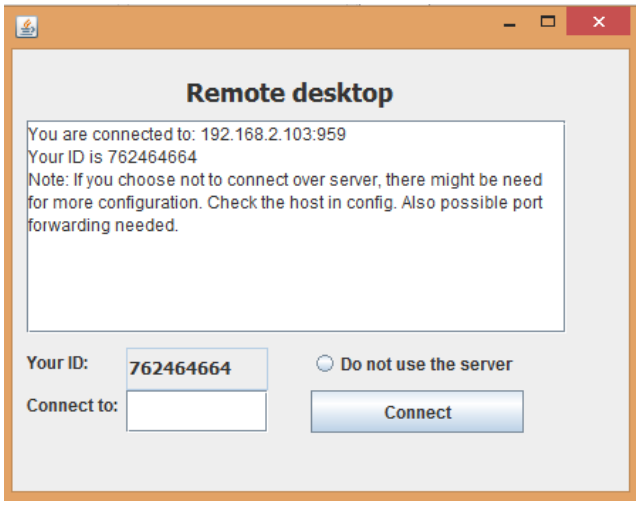
Z besedo odjemalec smo označili računalnik, na katerem se prikazuje vsebina drugega računalnika.

Ob zagonu programa uporabnik zagleda grafični vmesnik (slika 8), ki mu omogoča izbiro med tem, ali bo delil vsebino svojega ekrana ali pa bo gledal vsebino drugega računalnika oziroma upravljal z oddaljenim računalnikom.



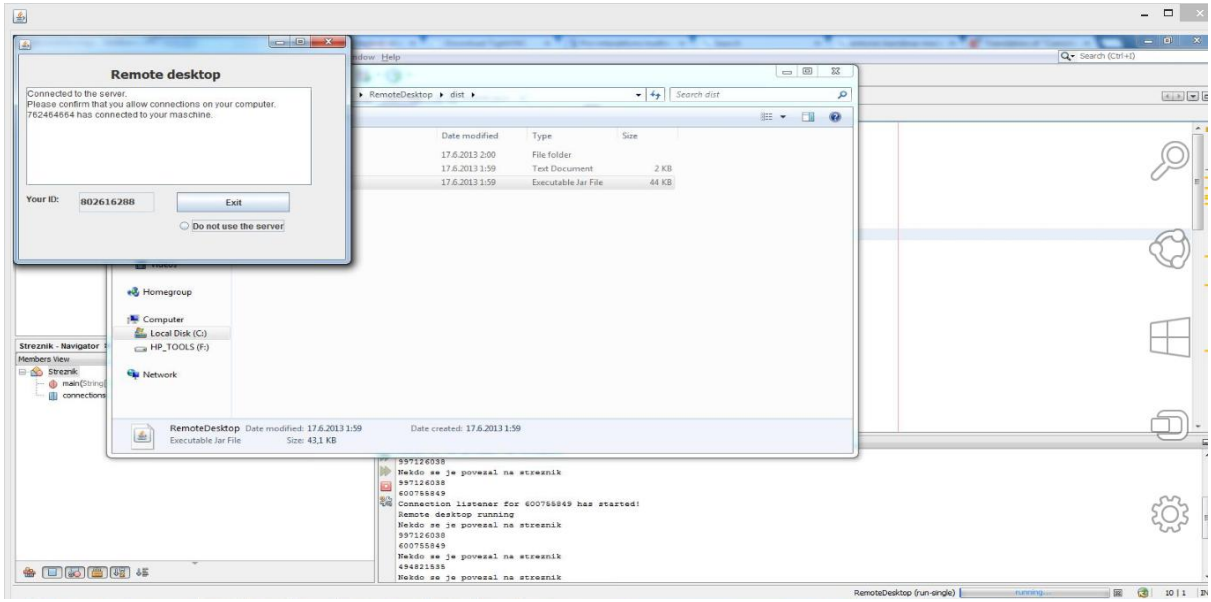
Slika 8: Grafični uporabniški vmesnik našega programa

Ob izbiri druge možnosti se uporabniku odpre novo okno, ki ga lahko vidimo na sliki 9.



Slika 9: Grafični uporabniški vmesnih našega programa – pogled odjemalca

Iz slike 9 lahko vidimo, da smo v tem trenutku že povezani na strežnik z našim enotnim identifikacijskim žetonom, ki omogoča strežniku, da poveže dva računalnika med seboj. Edino, kar mora uporabnik še storiti, da lahko začne oddaljeno upravljati z drugim računalnikom, je, da v polje, predvideno za to, vpiše enotni identifikacijski žeton računalnika, na katerega se želi povezati, in nato pritisne gumb »Poveži«. Ko se to zgodi, mora uporabnik na ciljnem računalniku še potrditi povezavo in oddaljeno upravljanje je omogočeno. Uporabnikovo okno nato izgleda tako, kot nam kaže slika 10.

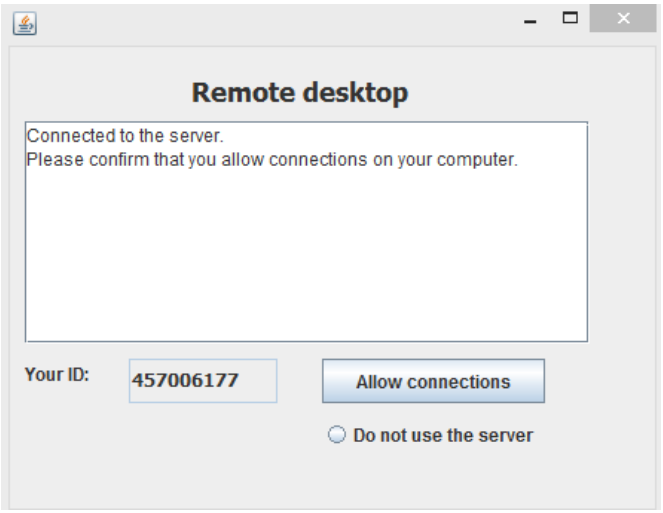


Slika 10: Aktivna povezava med dvema računalnikoma

V tem trenutku je odjemalec prevzel popolno kontrolo nad drugim računalnikom. Za upravljanje ima na voljo miško, tipkovnico in sliko, ki se prenaša v realnem času. Povezavo lahko v kateremkoli trenutku prekineta odjemalec ali gostitelj.

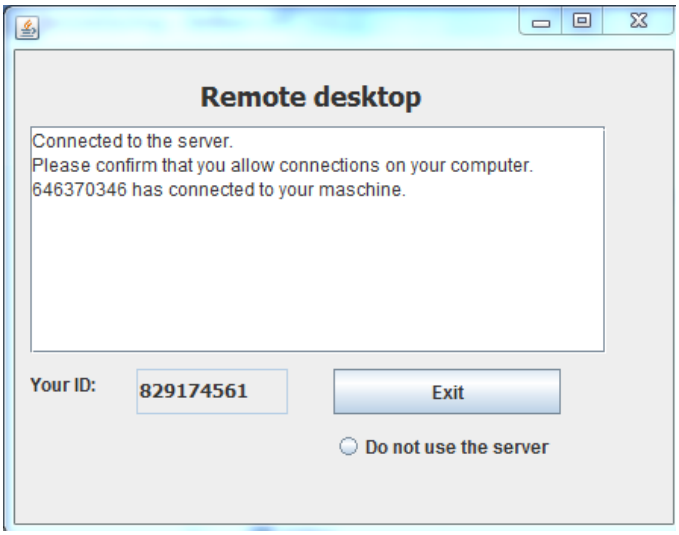
3.4.1.3 Gostitelj

Z besedo gostitelj označujemo računalnik, do katerega želimo dostopati. Ko želi uporabnik deliti svoj ekran z drugim uporabnikom, se odloči za vlogo gostitelja. Proces poteka podobno kot pri odjemalcu, le s to razliko, da si pri sliki 8 uporabnik izbere prvo možnost. Torej možnost deljenja svojega ekrana.



Slika 11: Grafični uporabniški vmesnih našega programa – pogled gostitelja

Podobno kot odjemalec, tudi gostitelj dobi svoj identifikacijski žeton, ki ga uporablja za prepoznavo in povezovanje. Gostitelj mora za tem le še potrditi prihajajočo povezavo. Ko to stori, se lahko odjemalec poveže na njegov računalnik in ga upravlja. Slika 12 prikazuje, kako izgleda gostiteljev ekran.



Slika 12: Grafični uporabniški vmesnih - pogled gostitelja

Uporabnik je obveščen o tem, če in kdo se je povezal na njegov računalnik. Iz slike 12 je razvidno tudi, da gostitelj s pritiskom na gumb »Exit« lahko prekine povezavo v kateremkoli trenutku, možnost »Do not use the server« pa izbere, če ne želi, da gre njegova povezava čez strežnik, opisan v poglavju 3.3.1.

3.4.2 Logika

3.4.2.1 Pošiljanje slike

Po predstavitvi glavnih delov programa si moramo ogledati še ozadje delovanja. V osnovi program za oddaljen dostop do računalnika deluje tako, da v neskončni while zanki gostitelj pošilja sliko svojega ekrana odjemalcu, odjemalec pa pošilja koordinate miške in pritiske miškinih gumbov ter tipk na tipkovnici nazaj gostitelju. Ob modernih ekranih ter računalnikih (visoka resolucija) so seveda posredovane slike lahko precej velike, zato prvi strah, s katerim se razvijalec spopade, nastopi ob misli na hitrost: ali bo lahko zagotovil ustrezno hitrost prenosa podatkov, torej slike, da bo vse skupaj še potekalo v realnem času. S podobno težavo smo se spopadli tudi sami in dobili nekaj idej, kako jo rešiti. V tem poglavju si bomo ogledali ideje, kasneje pa še implementacijo rešitve.

3.4.2.1.1 Slabe rešitve

Da bi lahko razlikovali dobre rešitve od slabih, si bomo ogledali najprej dve slabi rešitvi. Povprečna velikost slike zajema ekrana je lahko od nekaj 100 kb do dobrega MB. Najbolj enostavna rešitev je seveda ta, da z vsako iteracijo pošljemo celo sliko strežniku.

Po hitrem testu lahko vidimo, da ta rešitev deluje solidno, za kar se gre prej zahvaliti modernim internetnim povezavam kot naši domiselnosti. Takoj ko preizkusimo našo rešitev na počasnejši internetni povezavi, vidimo, da je prostora za optimizacijo še ogromno.

Druga rešitev, ki se ponuja, je ta, da pred pošiljanjem zmanjšamo kvaliteto slike in na ta način prihranimo pri količini prenosa podatkov. A ta rešitev nam ni všeč, ker s tem po nepotrebnem zmanjšamo kvaliteto storitve, saj bo slika na odjemalčevem računalniku precej slabša, kot bi lahko bila. Vidimo torej, da se tu splača malo ustaviti in premisliti. To smo tudi storili in dobili nekaj dobrih idej za rešitev problema.

3.4.2.1.2 Dobre rešitve

Dobrih rešitev je verjetno več, mi pa bomo v tem delu opisali dve od njih, ki sta si med seboj zelo podobni.

Osnovna ideja je pri obeh rešitvah enaka. Po krajšem razmisleku smo namreč prišli do zaključka, da se v vsakem trenutku spremeni zgolj del ekrana in se le redko kdaj zamenja celotna slika. Zakaj bi torej pošiljali celo sliko, ko pa lahko pošiljamo samo del slike, ki se je zamenjal? Tovrstno razmišljanje je ključno za dobro implementacijo takšnih programov. Poglejmo si, kako lahko to dejstvo izkoristimo.

Z vsako iteracijo zajamemo sliko na gostiteljevem ekranu. Sliko nato razrežemo na več enakih delov in te dele shranimo. Nato z vsako naslednjo iteracijo dele slike med seboj primerjamo in ko zaznamo spremembo, pošljemo spremenjeni košček na strežnik. Ta princip nam znatno zmanjša prenos podatkov med računalniki in pospeši delovanje programa.

Dve rešitvi za hitro primerjavo slik med seboj sta primerjava kontrolne vsote in primerjava naključno izbranih pikslov. Za vsak košček v vsaki iteraciji izračunamo kontrolno vsoto in če se ta spremeni, se je spremenila tudi naša slika. Drugače rečeno, v vsaki iteraciji vzamemo naključno število pikslov, ki ga primerjamo s prejšnjo sliko in ko le en piksel spremeni svojo barvo, takoj označimo sliko za pošiljanje.

Obe metodi sta zelo učinkoviti. V našem programu smo se odločili za implementacijo metode »naključni piksli«, sicer pa je metoda s kontrolno vsoto tudi zelo dobra, popularna in pogosto uporabljena v modernih programih. To izjavo žal ni mogoče podkrepiti z dostopno dokumentacijo, ker podjetja ne rada razkrivajo podrobno delovanje svojih programov, saj se tako obvarujejo pred konkurenco.

Seveda pri metodi »naključni piksli« lahko nekdo pomisli, da bi bilo nujno potrebno primerjati prav vse piksle, da bi zagotovo vedeli, če se je slika spremenila ali ne. To je sicer res, a ker se program poganja v neskončni zanki, lahko izberemo tako število pikslov, ki nam daje ugodno verjetnost za zaznavo spremembe. V kolikor spremembe ne zaznamo v prvi iteraciji, jo bomo zagotovo v kateri od naslednjih, tako da večje škode za delovanje programa s tem ne bomo povzročili.

3.4.2.2 Ostala logika

Preostanek logike je lažji, vendar vseeno ne zanemarljiv del implementacije. Verjetno je smotrno omeniti, da se zaradi uporabe razreda Robot in zgolj čiste Jave moramo sprijazniti z nekaterimi programskimi pomanjkljivostmi, a na ta račun pridobimo program, ki dela enako

dobro na vseh operacijskih sistemih. Pri pomanjkljivostih mislimo na zaznavanje pritiska več tipk hkrati, drsenje z miško in težjo implementacijo šumnikov. Ampak večino stvari se na oddaljenem računalniku še vedno da uporabljati z uporabo »zaslonske tipkovnice«, ki jo imajo vsi operacijski sistemi vgrajeno, tako da končni rezultat ni slab.

3.4.2.4 Programska koda

V tem poglavju si bomo pogledali programsko kodo in dejansko implementacijo pošiljanja slike med dvema računalnikoma.

```
public static void sliceAndSend(BufferedImage i, DataOutputStream dos)
throws Exception {
    int chunkWidth = i.getWidth() / rows;
    int chunkHeight = i.getHeight() / colums;
    int count = 0;
          BufferedImage temp = new BufferedImage(chunkWidth,
  chunkHeight, i.getType());
    int pixelTemp;
    int pixel;
    int randomx;
    int randomy;
    boolean diff = true;
    for (int x = 0; x < rows; x++) {
        for (int y = 0; y < colums; y++) {
            //Initialize the image array with image chunks
            if (imgs[count] != null) {

                temp = new BufferedImage(chunkWidth, chunkHeight,
                i.getType());

                Graphics2D gr = temp.createGraphics();
                    gr.drawImage(i, 0, 0, chunkWidth, chunkHeight,
              chunkWidth * y, chunkHeight * x, chunkWidth * y +
              chunkWidth, chunkHeight * x + chunkHeight, null);
                gr.dispose();

                if(methodNumber==1){
                    for (int m = 0; m < 500; m++) {
                        randomx = (int) (Math.random() * chunkWidth);
                        randomy = (int) (Math.random() * chunkHeight);
                        pixelTemp = temp.getRGB(randomx, randomy);
                        pixel = imgs[count].getRGB(randomx, randomy);

                        if (pixelTemp != pixel) {
                            diff = true;
                            break;
                        }
                    }

                    if (diff == true) {
                        imgs[count] = temp;
                        diff = false;
                        dos.writeBoolean(true);
```

```
                            ImageIO.write(imgs[count], "jpg", baos);
                            baos.flush();
                            imageInByte = baos.toByteArray();
                            baos.close();
                            dos.writeInt(imageInByte.length);
                            dos.write(imageInByte, 0, imageInByte.length);
                            dos.flush();
                            baos.reset();

                        } else {

                            dos.writeBoolean(false);
                        }
                    }


                    count++;

                } else {
                  imgs[count] = new BufferedImage(chunkWidth, chunkHeight,
            i.getType());
                    // draws the image chunk
                    Graphics2D gr = imgs[count++].createGraphics();
                                gr.drawImage(i, 0, 0, chunkWidth,
            chunkHeight, chunkWidth * y, chunkHeight * x, chunkWidth * y +
            chunkWidth, chunkHeight * x + chunkHeight, null);
                    gr.dispose();

                }
            }
        }
    }
```

Torej metoda SliceAndSend() dela točno to, kar je opisano v poglavju 3.3.4.1.2. Kot vhodne parametre dobi instanco razreda BufferedImage in DataOutputStream. BufferedImage je dejansko naša trenutna slika zaslona, DataOutputStream pa je tok za pošiljanje podatkov na strežnik. Metoda je tipa void, kar pomeni, da ne vrača ničesar.

Za nadaljnje delo bi bilo mogoče dopolniti seznam metod in jih med seboj primerjati.

3.4.2.5 Konfiguracija programa

Za lažjo konfiguracijo programa smo ustvarili posebno config.txt datoteko, iz katere program prebere nastavitve, kot so ip naslov strežnika in vrata, na katerih strežnik posluša. Format datoteke je

ključna beseda : vrednost ; oziroma, primer vrednosti je

port number:959;

host:192.168.2.103.

Poglavje 4

4.1 Zaključek

Kot smo pojasnili skozi prejšnja poglavja, obstajata dve skupini programov za oddaljen dostop do računalnika. To so programi, ki vzpostavijo neposredno povezavo med dvema računalnikoma, ter programi ki vzpostavijo povezavo med dvema računalnikoma, ki poteka preko nekega vmesnega strežnika. Pomembna razlika med omenjenima skupinama je prikladnost uporabe za končnega uporabnika. Glavna ugotovitev je, da je za končnega uporabnika vsekakor lažja uporaba programov, katerih povezava poteka skozi vmesni strežnik. Razlog za to je, da se tako uporabniku ni potrebno ukvarjati z nastavitvami požarnega zidu, ter usmerjevalnika. S tem tudi ne ogrozimo hitrosti delovanja programa. Edina potencialna slabost je varnost prenosa, vendar glede na napredne možnosti zaščite, ki so na voljo danes, vidimo da tudi tu ni upravičenega razloga za strah.

Zato smo se tudi pri naši implementaciji odločili za različico z vmesnim strežnikom. Sam projekt je bil zelo zanimiv in po njegovi izvedbi smo mnenja, da nam je uspelo izdelati solidno verzijo programa. Če ga primerjamo z ostalimi programi, vidimo, da za TeamVieawerjem malce zaostaja, prav tako za LogMeIn, vendar gre v obeh primerih za industrijsko naravnan program, ki ga skupina razvijalcev nenehno izboljšuje, zato smo z našim izdelkom lahko zadovoljni. Prostora za izboljšavo je seveda še nekaj, lahko bi dodali še kakšne funkcionalnosti, analizirali več metod delovanja in podobno. A uspelo nam je narediti program, ki je zelo prijazen za uporabo in deluje z določeno stopnjo zanesljivosti na vseh operacijskih sistemih. Program za eno iteracijo porabi približno 200ms, kar nam omogoča prenos do 8 iteracij na sekundo.

Po vsem raziskanem in povedanem lahko sklenemo, da je bil projekt uspešen, in upamo, da se bo tudi v prihodnje našel kdo, ki bo želel zadevo obuditi in projekt pripeljati do popolnosti.

Upamo tudi, da smo bralcu uspeli pokazati, da je implementacija programa za oddaljen dostop do računalnika zanimiv izziv, ki pa ni nepremagljiva ovira.

Literatura

1. http://lifehacker.com/5080121/five-best-remote-desktop-tools
2. http://www.infosecurityproductsguide.com/technology/2007/wp_lmi_security.pdf
3. https://secure.logmein.com/
4. http://www.uvnc.com/
5. http://en.wikipedia.org/wiki/Comparison_of_remote_desktop_software
6. http://www.tightvnc.com/
7. http://www.teamviewer.com/images/pdf/TeamViewer_SecurityStatement.pdf
8. http://www.teamviewer.com/download/teamviewer_manual.pdf
9. http://docs.oracle.com/javase/1.4.2/docs/api/java/net/Socket.html
10. http://docs.oracle.com/javase/6/docs/api/java/net/ServerSocket.html
11. http://docs.oracle.com/javase/7/docs/api/java/awt/Robot.html
12. Martin Georgiev, Subodh Iyengar, Suma Jana, Rishita Anubhai, Dan Boneh, Vitaly Shmatikov. The Most Dangerous Code in the World: Validating SSL Certificates in Non-Browser Software. Zbornik konference ACM 2012 o računalniški in komunikacijski varnosti, 38-49, 2012
13. The Information Security Forum. Securing Remote Access by Staff – Implemetation Guide. https://www.igt.hscic.gov.uk/KnowledgeBaseNew/ISF_Security%20Staff%20Remote%20Access_Implementation%20Guide.pdf, 1999