2 LABORATORINIS DARBAS

Atliko: Artūras Jonkus, EKSfm-16

1. *Repozitorijų kūrimas*

Tam, kad būtų galima skelbti duomenis į serverį, reikia sukurti atitinkamus aplankus tiek kliento, tiek serverio pusėje, bei tarp jų sudaryti tinkamą ryšį. Tada galima siųsti duomenis (*git push*) bei gauti duomenis (*git pull*) arba netgi nusikopijuoti visą repozitoriją su visais kada nors atliktais ir dokumentuotais pakeitimais (*git clone*). Esant poreikiui taip pat galima išskirti atšakas (angl. *branches*). Prieš kuriant repozitorijas reikia užtikrinti, kad git programinė įranga instaliuota. Jeigu ne, tai padaryti galima terminalo lange parašius *sudo apt-get update && sudo apt-get install git-core*.

Serverio pusėje sukurti repozitoriją nesudėtinga, o pavyzdys pateikiamas toliau.

Reikalinga sukurti atskirą vartotoją, kuris bus naudojamas kaip vietinis serveris. Tai gali būti jau esamas vartotojas, tačiau šiuo atveju sukuriamas naujas vartotojas *git*. Sukūrimui naudojama *sudo adduser git komanda*. Čia paprašoma parinkti slaptažodį, šiuo atveju jis – *git,* kaip ir vartotojo vardas. Visus kitus parametrus galima praleisti. Prie sukurto vartotojo failų galima prieiti surinkus *sudo su* bei suvedus esamo vartotojo (to, kuris prisijungęs dabar) slaptažodį. Nuo šiol *sudo* vartoti nereikia ir netgi negalima, kadangi vartotojas *git* nėra įtrauktas į *sudoers* grupę, todėl neturi leidimų naudotis šios funkcijos privalumais.

Po minėtų veiksmų namų aplanke */home/git/* sukuriamas *server_side/gitrepo/* aplankas. Tam reikalinga *mkdir server_side && cd server_side && mkdir gitrepo*. Tam, kad šiame aplanke būtų galima inicijuoti *git* direktoriją, naudojama *cd gitrepo && git –bare init*. Tai visi reikalingi žingsniai iš šio vartotojo (serverio) pusės. Grįžti atgal galima su *exit* && *sudo su arturas* (*arturas* - studento vardas). Taip pat galima tiesiog paleisti terminalo langą, uždarant sukurtąjį seniau.

Tam, kad būtų galima patikrinti, ar galima skelbti failus, reikia kažkur sukurti aplanką ir jame sukurti bent vieną failą. Tai atlieka *mkdir [projekto aplanko vieta] && cd [projekto aplanko vieta] && touch [failo vardas]* (skliaustuose nurodytos vertės priklauso nuo vartotojo pasirinkimo). Pavyzdžiui, *mkdir client_side && cd client_side && mkdir gitrepo && cd gitrepo & touch text.txt*. Atsiradusiame aplanke taip reikia sukurti *git* aplanką, su *git init* komanda. Sukurtas text.txt failas gali būti pridėtas *git add -A* komanda. Pirmą kartą kreipiantis sistema paprastai paprašo autentifikavimo teikiant failus versijavimui, todėl prieš tai reikėtų nurodyti elektroninį paštą ir vartotojo vardą. Jiems aprašyti atitinkamai reikalingos *git config --global user.email [vartotojo elektroninis paštas]* ir *git config --global user.name [vartotojo vardas]* komandos. Po šių komandų galima teikti failus versijavimui su *git commit –m "[pageidaujama žinutė]"*. Taigi, įrašius pirmąją eilutę į tekstinį failą atlikamas pateikimas serveriui *git commit –m „First commit"*.

Failas pateiktas versijavimui, tačiau dar neatiduotas lokaliam serveriui. Apskritai, tarp dviejų sukurtų aplankų dar kol kas nėra ryšio, jį reikia sukurti. Tam reikalinga *ssh* (*secure shell*) komunikaciją inicijuojanti programa OpenSSH (įrašoma su *sudo apt-get update && sudo apt-get install openssh-server*), po šios komandos sėkmingo įvykdymo reikėtų kviesti *git remote add origin git@localhost:server_side/gitrepo*, taip nurodant, kad pastarasis aplankas bus vieta, kur serveryje bus saugomi pakeitimai. Galiausiai pakeitimus pateikti serveriui galima su *git push origin master* bei sutinkant su visais pasirodančiais pasirinkimais. Suvedus *git status* turėtų pasirodyti užrašas, kad visi pakeitimai buvo atlikti ir duomenys yra sinchronizuoti (*nothing to commit, working directory is clean*). Tai reiškia, kad vietinis serveris yra sukurtas ir veikia tinkamai, o pirmoji žinutė jau yra serveryje.

2. *Duomenų iš repozitorijos gavimas ir pakeitimų darymas*

Tam, kad būtų dirbama su paskutiniais duomenimis, reikalinga pastoviai atnaujinti juos. Tai atlikti padeda *git pull* komanda.

Pirmiausia reikalinga sukurti aplanką *client2_side/gitrepo*. Kadangi žingsniai yra identiški vartotojo aplanko kūrimui pirmojoje dalyje, kūrimo proceso apibūdinimas praleidžiamas.

Sukūrus naują aplanka reikalinga gauti pakeitimus, kadangi kol kas aplankas tuščias. Tai reikėtų atlikti su *git pull origin master*. Sėkmės atveju aplanke turėtų atsirasti *text.txt* failas. Be jo dar aplanke taip pat gali būti paslėptų failų, pavyzdžiui *.gitignore*, kurio turinyje nurodoma, kurių failų pakeitimų serveriui pateikti nevalia.

3. *Konfliktų sprendimas*

Jeigu keletas vartotojų dirba su tuo pačiu projektu ir dėl to keičia tuos pačius failus, tikėtina, kad vienas iš jų atliks pakeitimus, kuriuos galbūt jau atliko kitas asmuo, arba ištrins būtent tai, ką pridėjo kitas. Toks nutikimas vadinamas konfliktu. Dėl šios priežasties paskutinis vartotojas negalės pilnai pateikti duomenų, prieš tai turės ištaisyti susidariusį konfliktą – ištrinti, pridėti arba pataisyti atitinkamas eilutes.

```
arturas@arturas-VirtualBox:~/Documents/client2_side/gitrepo$ git push origin mas
ter
git@localhost's password:
To git@localhost:server_side/gitrepo/
 ! [rejected]        master -> master (fetch first)
error: failed to push some refs to 'git@localhost:server_side/gitrepo/'
hint: Updates were rejected because the remote contains work that you do
hint: not have locally. This is usually caused by another repository pushing
hint: to the same ref. You may want to first integrate the remote changes
hint: (e.g., 'git pull ...') before pushing again.
hint: See the 'Note about fast-forwards' in 'git push --help' for details.
arturas@arturas-VirtualBox:~/Documents/client2_side/gitrepo$
```

2.1 pav. *git push origin master* komandos rezultatai

Laboratorinio darbo metu konfliktas taip pat nutiko. Antrajam vartotojui įrašius naują eilutę į failą, kurį jau pateikė ir pakeitimus į serverį įrašė pirmasis vartotojas. Antrasis vartotojas prieš atlikdamas pakeitimus nepaprašė gauti naujausių pakeitimų *git pull origin master* komanda, todėl neturi teisės be leidimo ignoruoti naujausių pakeitimų.

Kadangi vartotojas nepaprašė naujiausių pakeitimų, *git push origin master* išveda klaidos sąlygą (2.1 paveikslas). Vartotojas tada turi suvesti *git pull origin master* ir ekrane pamatęs, kad nutiko konfliktas, šiuos konfliktus išspręsti.

```
arturas@arturas-VirtualBox:~/Documents/client2_side/gitrepo$ git pull origin mas
ter
git@localhost's password:
remote: Counting objects: 3, done.
remote: Compressing objects: 100% (2/2), done.
Unpacking objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0)
From localhost:server_side/gitrepo
 * branch            master     -> FETCH_HEAD
   54b8133..b13b74b  master     -> origin/master
Auto-merging text.txt
CONFLICT (content): Merge conflict in text.txt
Automatic merge failed; fix conflicts and then commit the result.
arturas@arturas-VirtualBox:~/Documents/client2_side/gitrepo$
```

2.2 pav. *git pull origin master* komandos rezultatai

Konfliktų sprendimas nėra sudėtingas – *git* programa pažymi, kuriose vietose yra nesutapimai, tad galima ranka peržiūrint failus šiuos konfliktus išspręsti ištrinant nereikalingas eilutes.

Apie visus atliktus pakeitimus, kuriuos vartotojai perdavė serveriui, galima sužinoti *git log* komanda. 2.3 paveiksle parodyti rezultatai parodo, kad buvo atlikti visi laboratoriniame reikalingi pakeitimai serveryje (angl. *commits*). Viršuje pateikiami naujausi pakeitimai. Pirmuoju pateikimu pateikta pirmoji eilutė, antru pridėta antroji eilutė (antro vartotojo), trečiu pirmasis vartotojas pridėjo trečią eilutę, ketvirtu atliktas pakeitimas, sukėlęs konfliktą, kurį galiausiai išsprendė paskutinis pateiktas pakeitimas (*Fixed conflict*)

```
arturas@arturas-VirtualBox:~/Documents/client2_side/gitrepo$ git log --all --one
line
c56ea03 Fixed conflict
40b9b76 Fourth commit from second client
b13b74b Third commit from the first client
54b8133 Second commit from second client
f3c88d8 First commit
arturas@arturas-VirtualBox:~/Documents/client2_side/gitrepo$
```

2.3 pav. Visų atliktų pakeitimų sąrašas