SVN

Piskadło & Szymański*

Uniwersytet im. Adama Mickiewicza w Poznaniu

Referat w ramach przedmiotu *"Administracja Serwerami Sieciowymi Linux"*

Streszczenie

Dokument opisuje w jaki sposób działa i jakie funkcjonalności posiada system kontroli wersji Subversion (dalej zwanym SVN). W dokumencie poruszane są następujące tematy: sposoby podłączenia, używanie komend systemu, organizacja plików, tworzenie projektu, utrzymywanie kodu, zabezpieczanie, problemy związane z siecią, kopią zapasową repozytorium, jakie są popularne implementacje SVN, jak skonfigurować serwer SVN, jak on działa i jak zmienić parametry odpowiedzialne za jego działanie. Na końcu dokumentu znajduje się krótki tutorial pozwalający na zainstalowanie Subversion wraz z Apache i Trac.

I. Wprowadzenie

Subversion jest systemem kontroli wersji **(VCS: *version control system*)** na licencji Apache. Jest to oprogramowanie zarządzające plikami i katalogami oraz zmianami w nich wprowadzonymi, co pozwala na przywracanie konkretnych danych po wprowadzeniu niepożądanych zmian w strukturze repozytorium. Repozytorium jest głównym rdzeniem systemu kontroli wersji, to w nim przechowywane są informacje i dane systemu. Repozytorium zazwyczaj ma strukturę hierarchiczną na którą składają się katalogi i pliki. Dowolna liczba użytkowników, posiadająca uprawnienia, może zarówno edytować jaki i wyświetlać zawartość repozytorium. Wszelkie zmiany w repozytorium są rejestrowane w systemie, co pozwala kontrolować jego historię i pozwala na śledzenie kto jakie zmiany wprowadził. Repozytorium akceptując każdą zatwierdzoną zmianę użytkownika *(ang. commit)* tworzy nowy stan repozytorium. Każdą taką zatwierdzoną zmianę w repozytorium nazywamy wersją kontrolną lub rewizją *(ang. revision)*. Jednak aby nie operować na całym repozytorium, każdy użytkownik pracuje na swojej wersji roboczej. Oprócz danych użytkownika i struktury katalogów w repozytorium znajdują się także metadane, można je znaleźć w katalogach w katalogu .svn, natomiast od wersji 1.7 jest tylko jeden taki katalog dla całego repozytorium. Z systemami kontroli wersji wiąże się kilka problemów. Dotykającym każdego jest fundamentalny problem równoczesnego zapisu do pliku.

*Praca zbiorowa

II. Problemy i rozwiązania

Najpopularniejszym rozwiązaniem jest **lock-modify-unlock** czyli blokowanie dostępu do pliku na czas wprowadzenia zmian przez użytkownika. Jednak nie jest to najwygodniejsze rozwiązanie, przypuśćmy że nie odblokujemy pliku, wtedy żaden inny użytkownik nie może edytować pliku. Blokowanie dostępu do plików może stwarzać złudne poczucie bezpieczeństwa. Przykładem może być sytuacja gdy jeden z użytkowników edytuje i tym samym blokuje dostęp do jednego z plików. W tym samym czasie drugi z użytkowników repozytorium pracuje nad innym plikiem. Jest to jak znalazł przykład codziennego scenariusza codziennej pracy przy kodzie w repozytorium, jednak co jeśli jeden plik zależy od drugiego a po wprowadzonych i zatwierdzonych zmianach może wystąpić kolizja i niezgodność. Alternatywnym podejściem jest **copy-modify-merge**, rozwiązanie wykorzystywane zarówno w CVS jak i SVN. Polega to na pobraniu z repozytorium plików i pracy na ich lokalnych, prywatnych wersjach, by na końcu połączyć pliki z tymi znajdującymi się w repozytorium. Oczywiście system kontroli wersji

asystuje przy łączeniu *(ang. merge)* jednak to od człowieka zależy powodzenie tej operacji.

III. Podstawy podstaw

Ten rozdział będzie zbiorem podstawowych komend używanych przez klienta SVN wraz z opisem ich działania.

- **$ svn help** – zwraca zainstalowaną wersję Subversion oraz listę dostępnych komend
- **$ svn checkout http://svn.asl24.pl/svn/repos/assl** – tworzy kopie operacyjną repozytorium
- **$ svn update** – pobiera aktualną rewizję repozytorium
- **$ svn add myveryimportantdir** – dodaje pliki, katalogi lub linki symboliczne do repozytorium
- **$ svn import ./me/myimportantfile.pl http://svn.asl24.pl/svn/repos/assl/project** – umieszcza plik lub katalog dotychczas niepodlegający zarządzaniu wersją w repozytorium, nie wymaga kopii operacyjnej repozytorium
- **$ svn cleanup** – czyści repozytorium, np. z zablokowanych plików, nie zakończonych zmian itp
- **$ svn commit** – wysyła zmiany z kopi roboczej do repozytorium tworząc nową rewizję (dzięki parametrowi **-m** można dodać komentarz do rewizji)
- **$ svn revert** – cofa wszystkie zmiany dokonane lokalnie w repozytorium.
- **$ svn lock veryimportantfile.pl** – blokuje dostęp do pliku
- **$ svn unlock veryimportantfile.pl** – odblokowanie dostępu do pliku
- **$ svn log** – wyświetla log zmian w repozytorium
- **$ svn list http://svn.asl24.pl/svn/repos/assl** – wyświetlanie zawartości repozytorium
- **$ svn blame http://svn.asl24.pl/svn/assl/referat.tex** – wyświetla użytkownika odpowiedzialnego za zmiany i komentarz do pliku w określonej rewizji (parametr **-r**)
- **$ svn status veryimportantdir** – wyświetla status katalogu lub pliku w repozytorium
- **$ svnversion** – program raportujący stan kopii operacyjnej repozytorium

IV. Administrator ma zawsze racje

W tym rozdziale poruszona zostanie tematyka administracji repozytorium SVN. Aby utworzyć nowe repozytorium można skorzystać z narzędzi dostarczonych przez Subverion. Jednym z najważniejszych narzędzi administratora SVN jest svnadmin. Nowe repozytorium można stworzyć wpisując: **$ svnadmin create /var/svn/repos** Utworzone zostanie repozytorium o zerowej rewizji a w jego właściwościach będzie jedynie data jego utworzenia. Repozytorium SVN składa się z następujących katalogów: conf – w którym znajdują się pliki konfiguracyjne, db - . . . , format – przechowujący liczbę określającą rewizję, hooks – skrypty, locks – pliki blokady repozytorium oraz plik README.txt. Przy tworzeniu repozytorium możemy również określić z jakiego typu „systemu plików" ma korzystać. Do wyboru mamy BDB *(Berkeley DB)* oparty o model bazy danych oraz FSFS *(wymowa: ang. "fuzz-fuzz")* wykorzystujący strukturę plików. Istnieją różnice między nimi, jednak domyślnym, począwszy od SVN 1.2, jest FSFS.

- Zalety FSFS:
  - niezależny od platformy
  - użyteczny w sieciowych systemach plików
  - mniejsze zużycie przestrzeni dyskowej
  - szybszy w przypadku katalogów z dużą ilością plików
  - w SVN poniżej wersji 1.6 pliki zmian w repozytorium znajdowały się w różnych ścieżkach co powodowało problemy w sieciowych systemach plików oraz generowały straty w wykorzystaniu przestrzeni dyskowej, jednak od wersji 1.6 wszystkie

zmiany, na życzenie administratora mogą być przechowywane w jednym pliku

- Wady FSFS:
  - większe *commity* mogą powodować timeouty u użytkownika gdyż FSFS zwalnia przy finalizacji *commitów*
  - problemy ze skalowaniem starszych systemów plików z katalogami w których jest dużo plików
- Zalety BDB:
  - duża liczba rewizji nie powodują problemów
  - większe *commity* są wolniejsze jedna koszt jest amortyzowany podczas cyklu zatwierdzania zmian
  - izolacja transakcji oraz brak zjawiska przypadkowego „przebijania" między zmianami wprowadzanymi przez użytkowników
  - możliwość uskutecznienia backupu bez konieczności ustawiania repozytorium w stan „offline"
  - wszelkie zmiany są najpierw zapisywane w *logfile* przed przeprowadzeniem
- Wady BDB:
  - zależny od platformy
  - nie można używać w trybie „tylko do odczytu"
  - generalnie nie nadaje się w sieciowych systemach plików
  - większe zużycie przestrzeni dyskowej

Temat systemu plików będzie jeszcze kilkakrotnie poruszany w zależności od omawianego zagadnienia.
W katalogu *db* można znaleźć, w zależności od wybranej wersji „systemu plików" repozytorium, jego pliki konfiguracyjne. Dla BDB jest to plik *DB_CONFIG*. Korzystając z niego można ustawić takie własności jak: maksymalną liczbę blokad nadanych plikom czy maksymalny rozmiar pliku księgowania. Tutaj należy nadmienić iż wprowadzone zmiany w *DB_CONFIG* nie zostaną zastosowane dla repozytorium dopóty nie zostanie przywrócone środowisko repozytorium (korzystając z polecenia: **svnadmin recover**). Dla FSFS takim plikiem konfiguracyjnym jest *fsfs.conf*. Dla FSFS można ustawić wiele parametrów poprawiających wydajność repozytorium czy też zmniejszyć wykorzystanie przestrzeni dyskowej. Przykładem z życia wziętym może być problem pojawiający się przy aktualizacji SVN do wersji 1.6, który w swych plikach posiada zapis o wykorzystywanej wersji (format=4, poniżej wersji 1.6 było: format=3). Co to zmienia? Wykorzystywana jest wtedy opcja repo-sharing, która wykorzystuje do działania SQLite co nie działa poprawnie na sieciowym systemie plików AFS. Wystarczy więc w pliku *fsfs.conf* ustawić: ***repo-sharing off***. W katalogu *hook* naszego repozytorium znajdują się skrypty opisujące reguły w zależności od scenariusza działania repozytorium, można je porównać do triggerów w bazach danych. Głównym podziałem „scenariuszy" jest czas ich wykonania, dlatego wyróżniamy zdarzenia które mają się zdarzyć „pre hooks" oraz te które już się wydarzyły „post hooks". W hooks znajdziemy przykładowe szablony, takich skryptów jak: *post-commit.tmpl, post-lock.tmpl, post-revprop-change.tmpl, post-unlock.tmpl, pre-commit.tmpl, pre-lock.tmpl, pre-revprop-change.tmpl, pre-unlock.tmpl, start-commit.tmpl* Jednak Subversion wychodzi naprzeciw administratorom, oferując zestaw narzędzi ułatwiających pracę przy utrzymaniu repozytorium.

i **svnadmin** – główne narzędzie administratora repozytorium pozwalające na np. utworzenie nowego repozytorium.

ii **svnlook** – narzędzie wykorzystywane przy diagnostyce rewizji czy konkretnych transakcji.

iii **svndumpfilter** - narzędzie pomocne w operacjach na historii zmian repozytorium.

iv **svnsync** – narzędzie ma tak naprawdę tylko jedno zadanie – przenieść historię zmian z jednego repozytorium do drugiego.

v **fsfs-reshard.py** – jeśli repozytorium korzysta z fsfs to jest niezastąpione narzędzie przydające się do porządkowania repozytorium, najczęściej po aktualizacji do wersji wyższej niż 1.6. Narzędzie pozwala na przejście na nową strukturę w której metadane przechowywane są w jednym katalogu, a nie jak poprzednio w każdym podkatalogu. Co ważne przy przejściu z wersji 1.5 do nowszej, ta operacja nie zostanie wykonana automatycznie przez narzędzie aktualizujące. Skrypt Pythona można znaleźć w tools/server-side.

vi **mod_dav_svn** – rozszerzenie do serwera Apache, zapewniające dostępność repozytorium przez sieć.

vii **svnserve** – program działający jak proces daemon lub uruchamiany w przez SSH, zapewniający dostępność repozytorium przez sieć.

viii **db_dump** i **db_load** – narzędzia dostępne dla BDB, służą do migracji repozytorium między maszynami (pamiętając o wymogu analogii środowiska systemowego w przypadku DBD), o ile można korzystać svnadmin dump i svnadmin load, które działają tak samo, to db_dump i db_load są od nich szybsze.

V. Dobre rady

Rozdział poświęcony wskazówkom dotyczącym operacji wykonywanych na repozytorium czy z nim związanych. W pewnym momencie administratora repozytorium może najść ochota na zmianę logu wiadomości (własność: svn:log) towarzyszącym commitom w rewizji nr 299. W tej sytuacji, zakładając że w pliku mylog.txt posiadamy nowy plik logu, można postąpić następująco:

```
$ svnadmin setlog repozytorium mylog.txt
-r 299
```

Podczas korzystania z SVN mogą zdarzyć się przypadki błędów podczas commitów. W takim wypadku może wystąpić zjawisko martwiej transakcji, które samo w sobie nie jest szkodliwe jednak powoduje straty przestrzeni dyskowej. Aby wyświetlić tego typu transakcje:

```
$ svnadmin lstxns repozytorium
```

Aby sprawdzić kto jest za nie odpowiedzialny można użyć svnlook z parametrem –transaction (**-t**). Jednak aby posprzątać martwe transakcje należy posłużyć się svnadmin rmtxns a za wejście podać wynik **svnadmin lstxns**.

```
$ svnadmin rmtxns repozytorium ‘svnadmin
lstxns repozytorium‘
```

W Subversion 1.6 pojawiła się ciekawa opcja dla FSFS, pozwalająca zaoszczędzić miejsce na dysku, kompresując i pakując wszystkie nie podlegające zmianie pliki w jeden.

```
$ svnadmin pack /var/svn/repozytorium
```

Migracja repozytorium:

```
$ svnadmin dump repozytorium > dumpfile
```

```
$ svnadmin load repozytorium < dumpfile
```

Tworzenie backupu:

```
$ svnadmin hotcopy /var/svn/repos
/var/svn/repos-backup
```

VI. Serwer SVN

Subversion zostało zaprojektowane tak by dostęp programowy do serwera mógł odbywać się w sposób różny wraz z różną implementacją, wykorzystując oferowane API. Teoretycznie więc istnieje nieskończenie wiele implementacji sieciowych. Jednak w praktyce wykorzystywane są dwa najpopularniejsze rozwiązania.

mod_dav_svn jest modułem oferowanym wraz z serwerem Apache, pozwalającym na dostęp do repozytorium przy użyciu protokołu WebDAV/DeltaV, rozszerzenia protokołu HTTP. Apache zapewnia takie funkcjonalności jak: szyfrowanie SSL, logowanie zmian, limitowany dostęp do repozytorium przez „web”.

svnserve jest programem korzystającym z własnego protokołu a jego główną zaletą jest szybkość. Do autoryzacji może wykorzystywać SASL zapewniając różnorodność szyfrowania czy autoryzacji. Zazwyczaj wykorzystywany z tunelami SSH. Jest również przydatny dla zapewniania dostępu do repozytorium użytkownikom należącym do konkretnej grupy, korzystając z lokalnego protokołu file://. Na końcu dokumentu znajduje się tabela porównująca dostępne rozwiązania. Zalety i wady:

- Zalety svnserve:
  - łatwy w konfiguracji
  - protokół sieciowy szybszy niż WebDAV
  - nie ma potrzeby konfigurowania kont użytkowników
  - hasło nie jest przesyłane przez sieć
- Wady svnserve:
  - domyślnie oferuje jeden sposób autoryzacji, protokół sieciowy nie jest szyfrowany, a hasła przechowywane są na serwerze w formie plaintext
  - nie posiada zaawansowanych metod logowania zmian
  - nie posiada wbudowanego modułu pozwalającego na przeglądanie przy użyciu przeglądarki internetowej
- Zalety SVN przez SSH:
  - protokół sieciowy szybszy niż WebDAV
  - wykorzystuje konta i strukturę użytkowników SSH
  - szyfrowany ruch sieciowy
- Wady SVN przez SSH:
  - tylko jeden sposób autoryzacji
  - nie posiada zaawansowanych metod logowania zmian
  - wymaga aby użytkownicy byli w tej samej grupie lub używali dzielonego klucza SSH
- Zalety Apache i mod_dav_svn:
  - zapewnia wiele sposobów autoryzacji – wszystkie dostępne w Apache
  - nie ma potrzeby tworzenia kont użytkowników w systemie
  - system logowania zmian
  - ruch sieciowy szyfrowany przy użyciu SSL
  - wbudowany moduł obsługi przez przeglądarkę internetową
  - repozytorium może zostać zamontowane jako udział sieciowy
- Wady Apache i mod_dav_svn:
  - wolniejszy od svnserve
  - konfiguracja początkowa może sprawiać problemy

VII. *svnserve*

Svnserve jest lekkim programem operującym na na własnym protokole poprzez TCP/IP. Klienci kontaktują się z serwerem przy użyciu URL: *svn://* lub *svn+ssh://*. Svnserve można uruchomić na różne sposoby, jako demon nasłuchujący zapytań, jako Uniksowy demon **inetd** nasłuchujący na konkretnym porcie, poprzez tunel SSH, jako usługę Microsoft Windows czy przy użyciu **launchd**. Aby uruchomić svnserve jako demon:

```
$ svnserve -d
```

aby zmienić domyślny port *(3690)* można posłużyć się flagą **–listen-port** lub **–listen-host**. Aby ograniczyć dostęp do katalogów użyj flagi **-r /var/repos**. Po uruchomieniu demona wszystkie repozytoria są dostępne (chyba że użyliśmy flagi -r) a ich dostęp można uzyskać znając ich ścieżkę absolutną, np.

```
svn://host.example.com/var/svn/asslproject
```

Uruchamiając svnserve poprzez inetd należy w pliku */etc/services* dopisać dwie linie:

```
svn 3690/tcp


svn 3690/udp
```

następnie w */etc/inetd.conf* dopisać:

```
svn stream tcp nowait svnowner
/usr/bin/svnserve svnserve -i
```

co uruchomi svnserve przez inetd a serwer i klient przy komunikacji używać będą *stdin* i *stdout*. Aby uruchomić svnserve jako usługę Microsoft Windows należy użyć:

```
C:\> sc create svn
binpath= "C:\svn\bin\svnserve.exe -service -r
C:\repos"
displayname= "Subversion Server"
depend= Tcpip
start= auto
```

Co utworzy nową usługę o nazwie „svn" uruchamiającą plik svnserve.exe, który uruchomi serwer snvserve ustawiając ścieżkę repozytorium na „C:\repos". Należy zwrócić uwagę na spację przy key= value w tym poleceniu, która jest wymagana do poprawnego działania. Gdy już zdefiniowano usługę, może ona być być uruchamiana i zatrzymywana przy użyciu: **net start svn** i **net stop svn**. Po połączeniu klienta z svnserve, klient wybiera repozytorium, serwer sprawdza *conf/svnserve.conf* w celu autoryzacji i weryfikacji użytkownika. Domyślnie svnserve korzysta wyłącznie z CRAM-MD5, polega to na wysłaniu do klienta małej ilości danych, następnie użytkownik, korzystając z algorytmu funkcji skrótu MD5 użytego na otrzymanych danych i swoim haśle przesyła wynik do serwera, który to sprawdza otrzymany fingerprint i zezwala bądź nie, na dostęp do repozytorium. Plik *svnserve.conf* jest centralnym mechanizmem kontroli, weryfikacji i autoryzacji. Plik podzielony jest na sfery (w pliku oznaczone przez *[nazwa sfery]*) a parametry i wartości mają następującą formę: *key = value*. Wykorzystanie SASL jest dobrym sposobem zabezpieczenia svnserve. SASL zapewnia kilka różnych sposobów autoryzacji, np. Kerberos (GSSAPI), NTLM, One-Time-Passwords (OTP), DIGEST-MD5, LDAP, Secure-Remote-Password (SRP) i inne. Aby skonfigurować SASL dla svnserve, w svnserve.conf należy dopisać:

```
[sasl]
use-sasl = true
```

Następnie utworzyć w katalogu, gdzie znajdują się biblioteki i moduły SASL, plik o nazwie *svn.conf* i następującej zawartości:

```
pwcheck_method: auxprop
auxprop_plugin: sasldb
sasldb_path: /etc/my_sasldb
mech_list: DIGEST-MD
```

gdzie plik */etc/my_sasldb* można utworzyć korzystając z

```
$ saslpasswd2 -c -f /etc/my_sasldb -u
realm username
```

Aby korzystać z svnserve przez SSH wystarczy używać URLu: *svn+ssh://* do połączenia z repozytorium. Warte zapamiętania jest iż ten rodzaj połączenie nie wymaga działającego demona snvserve po stronie serwerowej, a jego instancja powoływana jest w momencie powoływania połączenia przez SSH.

VIII. *httpd*

W tym rozdziale zostanie poruszona problematyczna tematyka związana z konfiguracją serwera Apache do działania z SVN. Przed rozpoczęciem należy sprawdzić czy posiadaną wersją serwera httpd jest wersja 2.0 lub wyższa oraz moduły: mod_dav i mod_dav_svn. Korzystając a2enmod dav_svn lub edytując pliki konfiguracyjne Apache (*LoadModule dav_svn_module modules/mod_dav_svn.so* i *LoadModule dav_module modules/mod_dav.so*) należy aktywować moduły. Teraz należy dodać lokalizację repozytorium do ustawień Apache.

```
<Location /repos>

DAV svn

SVNPath /var/svn/repository

</Location>
```

Jeśli w katalogu będzie składowane kilka repozytoriów dostępnych dla Apache, można wskazać „rodzica" korzystając z

```
SVNParentPath /var/svn
```

. Tak ustawiony serwer Apache będzie zezwalał na dostęp wszystkim użytkownikom znającym jego adres. W celu wprowadzenia zabezpieczeń należy utworzyć plik z użytkownikami i ich hasłami przy użyciu

```
htpasswd -c -m /etc/svn-auth.htpasswd
username
```

. Następnie edytować pliki z lokalizacją repozytorium, dopisując następujące linie:

```
AuthName "Subversion repository"

AuthType Basic

AuthUserFile /etc/svn-auth.htpasswd

Require valid-user
```

Można również zaszyfrować połączenie z serwerem używając SSL.

IX. Ostatnich słów kilka o konfiguracji SVN

W pliku servers znajdują się opcje konfiguracyjne serwera Subversion związanego z warstwą sieciową. Występuje tutaj podział na sekcje globalną, dotyczącą wszystkich składowych serwera *[global]* jak i sekcję opisującą grupy *[groups]* i tym samym same grupy *[group name]*. Parametry serwera jakie można ustawić, to:

a *http-auth-types* – dostępne tryby uwierzytelniania użytkowników, dostępne wartości to: *basic*, *digest* i *negotiatie*.

b *http-compression* – *yes* jest domyślną opcją, oznacza kompresję zapytań do serwerów obsługujących DAV.

c *http-timeout* – czas oczekiwania serwera na odpowiedź

d *ssl-authority-files* – wskazanie na folder z certyfikatami (HTTPS)

e *ssl-client-cert-file* – ustawienie certyfikatu SSL dla Subversion

f *store-passwords* – cache'owanie haseł

g *store-plaintext-passwords* – przechowywanie haseł w niezaszyfrowanej postaci, dotyczy tylko systemów Uniksowych

W pliku *config* znajdują się ustawienia dotyczące działania *svnserve*. Sekcja *[auth]* zawiera ustawienia autoryzacji w SVN.

a *password-stores* – można określić z jakiego, jeśli w ogóle, systemowego zarządcy haseł ma korzystać Subversion, dostępne opcje to: *gnome-keyring*, *kwallet*, *keychain*, *windows-crypto-api*

Sekcja *[helpers]* służy pomocy zewnętrznym aplikacjom, by te mogły poprawnie wykonywać swoje zadania.

a *diff-cmd* – wskazuje ścieżkę do programu porównującego

b *editor-cmd* – wskazuje ścieżkę do edytora tekstowego z którego będzie korzystał użytkownik w nagłych wypadkach

c *merge-tool-cmd* – wskazuje ścieżkę do programu łączącego pliki

Sekcja *[miscellany]* zbiera wszystkie pozostałe parametry.

a *Log-encoding* – ustawienie kodowania logów

b *no-unlock* – automatycznie usuwa wszystkie blokady w momencie commita

Jest również sekcja *[tunnels]* odpowiedzialna za ustalenie schematów połączeń przez *svnserve* i *ssh://*.

X. Tutorial

Instalacja subversion:

```
sudo apt-get update

sudo apt-get install subversion
```

Tworzenie repozytorium:

```
cd /var

sudo mkdir svn

sudo svnadmin create /var/svn/repos
```

Dodanie użytkownika *svn*

```
sudo adduser svn
```

Nadanie własności do katalogu *svn*

```
sudo chown -R svn.svn svn
```

Dodanie do użytkownika *mateusz* do grupy:

```
sudo vigr
```

W pliku dodaj:

```
admin:x:110:mateusz

svn:x:1001:mateusz
```

Instalacja serwera SSH:

```
sudo apt-get install openssh-server
```

Test połączenia:

```
svn co
svn+ssh://username@machinename/var/svn/repos
```

Teraz w katalogu *.ssh* i pliku *authorized_keys2* nadaj uprawnienia:

```
chmod 0700 .ssh

chmod 0600 .ssh/authorized_keys2
```

W */etc/ssh/sshd_config* dodaj linię ***PasswordAuthentication*** *no*. Następnnie używając *sshkeygen* utwórz klucze. Do konfiguracji Apache potrzebne będzie *libapache2-svn*, więc zainstaluj:

```
sudo apt-get install libapache2-svn
```

Następnie edytuj zawartość pliku *000-default*

```
cd /etc/apache2/sites-enabled

sudo vi 000-default
```

I dodaj następujące linie:

```
<Location /svn/repos>

DAV svn

SVNPath /var/svn/repos

AuthType Basic

AuthName "Subversion Repository"

AuthUserFile /etc/apache2/passwords

Require valid-user

</Location>
```

Dodaj plik haseł:

```
sudo htpasswd -cb /etc/apache2/passwords
mateusz dgjan08
```

Restart Apache:

```
sudo /etc/init.d/apache2 force-reload
```

Instalacja Trac:

```
sudo apt-get install trac python-setuptools
libapache2-mod-python enscript
```

Tworzenie bazy Trac:

```
sudo mkdir /var/www/trac

sudo trac-admin /var/www/trac/repos
initenv

cd /var/www

sudo chown -R www-data.svn trac
```

Po raz kolejny edytuj *000-default* dodając:

```
<Location /trac/[[:alnum]]+/login">

 AuthType Basic

 AuthName "Subversion Repository"

 AuthUserFile /etc/apache2/passwords

 Require valid-user

 </Location>

 <Location /trac>

SetHandler mod_python

PythonInterpreter main_interpreter

PythonHandler trac.web.modpython_frontend

PythonOption TracEnvParentDir /var/www/trac

PythonOption TracUriRoot /trac

</Location>
```

Na koniec restart Apache:

```
sudo /etc/init.d/apache2 force-reload
```

Literatura

[B. Collins-Sussman, B. W. Fitzpatrick, C. M. Pilato, 2011] Version Control with Subversion. For Subversion 1.7.