

FreeRADIUS z PostgreSQL na systemie Sun Solaris 10(x86)

autor: Jakub Wartak <vnull@pcnet.com.pl>

Wstep:

Polskich literek jak zawsze nie bedzie, ale do rzeczy. Nie bede sie tutaj bawil w advocacy dlaczego PostgreSQL a nie MySQL (ktory jak bedzie mozna zobaczyc przy omawianiu FreeRADIUSa bedzie rowniez skompilowany plugin, bo Sun dostarcza razem z Solarisem takze MySQL – 4.1.XX).

MySQL pomimo swojej reputacji

Jest co prawda engine InnoDB (ale z moich doswiadczen wynika..

Troche informacji o systemie na ktorym byla przeprowadzana instalacja:

```
root@solek:/# uname -a
```

```
SunOS solek 5.10 Generic_118844-26 i86pc i386 i86pc
```

```
root@solek/# psrinfo -v
```

```
Status of virtual processor 0 as of: 02/27/2006 12:39:38
```

```
on-line since 02/27/2006 12:06:38.
```

```
The i386 processor operates at 667 MHz,
```

```
and has an i387 compatible floating point processor.
```

```
root@solek:/# prtconf -v | grep Memory
```

```
Memory size: 1024 Megabytes
```

A wiec widac, ze sprzet jest leciwy, ale do testow wystarcza (po 5 minutach, jak tylko wstanie ;))

Instalacja PostgreSQL:

PostgreSQL instalowany bedzie bezposrednio ze zrodel, instalowana wersja to 8.1.3 (najnowsza na dzien 28.02.2006), uzywanym kompilatorem bedzie GCC.

Wstepne przygotowanie srodowiska:

```
root@solek:/# export PATH=$PATH:/usr/sfw/bin:/usr/sfw/i386-sun-solaris2.10/bin
```

(bo gcc jest w /usr/sfw/bin/gcc, w i386-sun-solaris2.10/bin sa narzedzia typu strip, ar itd)

```
root@solek:/# mkdir ~vnull/pgsql; cd ~vnull/pgsql
```

```
root@solek:/users/vnull# wget ftp://ftp8.pl.postgresql.org/pub/postgresql/source/v8.1.3/postgresql-base-8.1.3.tar.bz2
```

```
root@solek:/users/vnull# bzip2 -d postgresql-base-8.1.3.tar.bz2; tar xf postgresql-base-8.1.3.tar
```

```
root@solek:/users/vnull# cd postgresql-8.1.3
```

Teraz mozna przeczytac sobie plik INSTALL i/lub README w celu lepszego zapoznania sie z procedura instalacyjna.

Wstepna konfiguracja

```
root@solek:/users/vnull# ./configure --prefix=/opt/pgsql --without-readline CFLAGS="-O3 -pipe"
```

Male rozjasnienie:

-O3 = wlacza maksymalna optymalizacje

-pipe = przyspiesza kompilacje przez niezapisywanie posrednich plikow wynikowych w /tmp lecz przez przekazywanie ich przez IPC

Umyslnie nie instaluje Postgresa w /usr/local (default), gdyz np. sunfreeware.com lubi tam upychac swoje paczki.

Po tym jak configure sie zakonczy pomyslnie, przystepujemy do kompilacji:

```
root@solek:/users/vnull# gmake
```

Jesli wszystko przebieglo pomyslnie , to powita nas:

All of PostgreSQL successfully made. Ready to install.

Teraz wlasciwa instalacja:

```
root@solek:/users/vnull# gmake install
```

I powinniśmy dostac:

PostgreSQL installation complete.

Teraz tworzymy nowego, dedykowanego dla bazy uzytkownika w systemie (kwestia bezpieczenstwa):

```
root@solek:/# groupadd postgres
```

```
root@solek:/# useradd -g postgres -d /opt/pgsql postgres
```

Tworzymy katalog gdzie beda przetrzymywane (domyslnie) bazy danych:

```
root@solek:/# cd /opt/pgsql
```

```
root@solek:/opt/pgsql# mkdir data
root@solek:/opt/pgsql# chown postgres:postgres data
```

Inicjalizujemy PostgreSQL:

```
root@solek:/opt/pgsql# su - postgres
Sun Microsystems Inc. SunOS 5.10 Generic January 2005
$ pwd
/opt/pgsql
$ id
uid=101(postgres) gid=101(postgres)
$
$ bash
bash-3.00$ bin/initdb -D /opt/pgsql/data
The files belonging to this database system will be owned by user "postgres". This user must also own the server process.
```

The database cluster will be initialized with locale C.

```
fixing permissions on existing directory /opt/pgsql/data ... ok
creating directory /opt/pgsql/data/global ... ok
creating directory /opt/pgsql/data/pg_xlog ... ok
creating directory /opt/pgsql/data/pg_xlog/archive_status ... ok
creating directory /opt/pgsql/data/pg_clog ... ok
creating directory /opt/pgsql/data/pg_subtrans ... ok
creating directory /opt/pgsql/data/pg_twophase ... ok
creating directory /opt/pgsql/data/pg_multixact/members ... ok
creating directory /opt/pgsql/data/pg_multixact/offsets ... ok
creating directory /opt/pgsql/data/base ... ok
creating directory /opt/pgsql/data/base/1 ... ok
creating directory /opt/pgsql/data/pg_tblspc ... ok
selecting default max_connections ... 100
selecting default shared_buffers ... 1000
creating configuration files ... ok
creating template1 database in /opt/pgsql/data/base/1 ... ok
initializing pg_authid ... ok
```

enabling unlimited row size for system tables ... ok
initializing dependencies ... ok
creating system views ... ok
loading pg_description ... ok
creating conversions ... ok
setting privileges on built-in objects ... ok
creating information schema ... ok
vacuuming database template1 ... ok
copying template1 to template0 ... ok
copying template1 to postgres ... ok

WARNING: enabling "trust" authentication for local connections
You can change this by editing pg_hba.conf or using the -A option the
next time you run initdb.

Success. You can now start the database server using:

```
bin/postmaster -D /opt/pgsql/data
```

or

```
bin/pg_ctl -D /opt/pgsql/data -l logfile start
```

```
bash-3.00$ ^D
```

Wychodzimy sobie z su, aby zmodyfikowac katalog domowy usera postgres na /opt/pgsql/data,
shella tez przy okazji, aby bylo wygodniej:

```
root@solek:/opt/pgsql# usermod -s /bin/bash -d /opt/pgsql/data postgres
```

Jeszcze troche tunningu ;)

```
root@solek:/opt/pgsql# su - postgres
```

```
Sun Microsystems Inc. SunOS 5.10 Generic January 2005
```

```
postgres@solek:~$ cat > .bash_profile
```

```
export PATH=$PATH:/opt/pgsql/bin
```

```
postgres@solek:~$ logout
```

```
root@solek:/opt/pgsql# su - postgres
```

```
Sun Microsystems Inc. SunOS 5.10 Generic January 2005
```

```
postgres@solek:~$ echo $PATH
```

```
/usr/local/bin:/usr/bin:/bin:/usr/bin/X11:/usr/games:/usr/bin::/opt/pgsql/bin
```

Ok, przechodzimy do pierwszego odpalenia bazy (tymczasowego) oraz bardzo pobieznego sprawdzenia jej wydajnosci:

```
postgres@solek:~$ postmaster -D /opt/pgsql/data
```

```
LOG: could not bind IPv6 socket: Cannot assign requested address
```

```
HINT: Is another postmaster already running on port 5432? If not, wait a few se  
conds and retry.
```

```
LOG: database system was shut down at 2006-02-27 12:49:04 CET
```

```
LOG: checkpoint record is at 0/33F25C
```

```
LOG: redo record is at 0/33F25C; undo record is at 0/0; shutdown TRUE
```

```
LOG: next transaction ID: 565; next OID: 10794
```

```
LOG: next MultiXactId: 1; next MultiXactOffset: 0
```

```
LOG: database system is ready
```

```
LOG: transaction ID wrap limit is 2147484146, limited by database "postgres"
```

```
^Z
```

```
[1]+ Stopped postmaster -D /opt/pgsql/data
```

```
postgres@solek:~$ bg %%
```

```
[1]+ postmaster -D /opt/pgsql/data &
```

```
postgres@solek:~$ psql
```

```
Welcome to psql 8.1.3, the PostgreSQL interactive terminal.
```

```
Type: \copyright for distribution terms
```

```
  \h for help with SQL commands
```

```
  \? for help with psql commands
```

```
  \g or terminate with semicolon to execute query
```

```
  \q to quit
```

```
postgres=# select version();
```

```
          version
```

```
-----  
-----
```

```
PostgreSQL 8.1.3 on i386-pc-solaris2.10, compiled by GCC gcc (GCC) 3.4.3 (csl-s  
ol210-3_4-branch+sol_rpath)
```

```
(1 row)
```

```
postgres=# \l
```

```
List of databases
```

```
Name | Owner | Encoding
-----+-----+-----
postgres | postgres | SQL_ASCII
template0 | postgres | SQL_ASCII
template1 | postgres | SQL_ASCII
(3 rows)
```

Czyli wstepnie jest wszystko OK.

```
root@solek:/# cd ~vnull/pgsql/postgres-8.1.3/contrib/pgbench
```

```
root@solek:/# gmake
```

```
root@solek:/# gmake install
```

Teraz w /opt/pgsql/bin mamy narzedzie, ktorym mozemy przeprowadzic bardzo, ktotki i pobiezny test, ja tego tutaj uzywam do sprawdzenia czy baza sie nie wysypie, czy opcje kompilacji nie zaszkodzily zbytnio stabilnosci itd. UWAGA: w warunkach produkcyjnych **nalezy** sprawdzic PostgreSQL'a jego regress-testem (wiecej w INSTALL).

```
postgres@solek:~$ createdb radius
```

```
LOG:  transaction ID wrap limit is 2147484146, limited by database "postgres"CRE
ATE DATABASE
```

Pozwalamy uzytkownikowi radius na maksymalnie 60 jednoczesnych polaczen.

```
postgres@solek:~$ createuser -c 60 radius
```

```
Shall the new role be a superuser? (y/n) n
```

```
Shall the new role be allowed to create databases? (y/n) n
```

```
Shall the new role be allowed to create more new roles? (y/n) n
```

```
CREATE ROLE
```

Inicjalizacja bazy do mini-benchmarku:

```
postgres@solek:~$ pgbench -i radius
```

```
ERROR: table "branches" does not exist
```

```
ERROR: table "tellers" does not exist
```

```
ERROR: table "accounts" does not exist
```

```
ERROR: table "history" does not exist
```

```
creating tables...
```

```
10000 tuples done.
```

20000 tuples done.
30000 tuples done.
40000 tuples done.
50000 tuples done.
60000 tuples done.
70000 tuples done.
80000 tuples done.
90000 tuples done.
100000 tuples done.

set primary key...

NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "branches_pkey"
" for table "branches"

NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "branches_pkey"
" for table "branches"

NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "tellers_pkey"
for table "tellers"

NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "tellers_pkey"
for table "tellers"

NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "accounts_pkey"
" for table "accounts"

NOTICE: ALTER TABLE / ADD PRIMARY KEY will create implicit index "accounts_pkey"
" for table "accounts"

vacuum...LOG: transaction ID wrap limit is 1073742425, limited by database "radius"

LOG: transaction ID wrap limit is 1073742425, limited by database "radius"
done.

A teraz wlasciwy test:

postgres@solek:~\$ pgbench -c 8 -t 200 radius

starting vacuum...end.

transaction type: TPC-B (sort of)

scaling factor: 1

number of clients: 8

number of transactions per client: 200

number of transactions actually processed: 1600/1600

tps = 86.450537 (including connections establishing)

tps = 87.330767 (excluding connections establishing)

UWAGA: ta baza nie byla tuningowana, stanadardowo PostgreSQL startuje z takimi ustawieniami, zeby mogli sie podniesc na kazdym sprzecie; jako ze temat optymalizacji PostgreSQLa jest rozlegly nie bedzie tutaj omawiany – jest masa publikacji w internecie na ten temat.

Czas posprzatac:

radius=# \dt

List of relations

Schema	Name	Type	Owner
public	accounts	table	postgres
public	branches	table	postgres
public	history	table	postgres
public	tellers	table	postgres

(4 rows)

radius=# drop table accounts;

DROP TABLE

radius=# drop table branches

radius=# ;

DROP TABLE

radius=# drop table history;

DROP TABLE

radius=# drop table tellers;

DROP TABLE

radius=#

Zmuszenie PostgreSQL do wstawiania razem z system-em pozostawiam czytelnikowi :)

Podsumowujac: mamy dzialajacego juz PostgreSQL, wiemy ze w miare to to dziala. Teraz nalezy go jakos wykorzystac...

Instalacja FreeRADIUSa:

```
root@solek:/users/vnull# wget ftp://ftp.freeradius.org/pub/radius/freeradius-1.1.0.tar.gz
root@solek:/users/vnull# gzip -d freeradius-1.1.0.tar.gz; tar xf freeradius-1.1.0.tar
root@solek:/users/vnull# export PATH=$PATH:/usr/sfw/bin:/usr/sfw/i386-sun-solaris2.10/bin
root@solek:/users/vnull# cd freeradius-1.1.0
```

UWAGA: brak biblioteki GDBM (i jej plikow naglowkowych) moze skutkowac brakiem nastepujacych modutow:

```
    rlm_pool
    rlm_dbm
    rlm_counter
```

A teraz magiczne zaklecie:

```
root@solek:/users/vnull/freeradius-1.1.0# CFLAGS="-pipe -I/opt/pgsql/include" LDFLAGS="-L
/opt/pgsql/lib -L/usr/sfw/lib" ./configure --prefix=/opt/freerad --with-logdir=/var/log/radius --with-
mysql-include-dir=/usr/sfw/include/mysql --with-mysql-lib-dir=/usr/sfw/lib --with-postgresql-lib-
dir=/opt/pgsql/lib --with-postgresql-include-dir=/opt/pgsql/include
```

Wszystko powyzsze jest w jednej linijce. Co gorsza nie udalo mi sie znalezc sposobu na przekazanie do rlm_sql_postgresql odpowiednich parametrow tak, aby juz samo ./configure dobrze wszystko poustawialo (chyba ze jakos przez INCLTDL czy cos podobnego). Kompilacja bedzie przebiegac w II czesciach, pierwsza – do bledu w rlm_sql_postgresql; druga – polegajaca na poprawce w Makefile od rlm_sql_postgresql – po bledzie :)

Mozliwe takze, ze komus moze sie nie udac samo juz ./configure, dlatego proponuje sporobowac przed ./configure ustawic zmienna SHELL na /bin/bash (gdyby configure wypisane wyzej nie zadzialalo) :

```
root@solek:/users/vnull/freeradius-1.1.0# SHELL="/bin/bash" CFLAGS="-pipe..." i cala reszta jak
wyzej...
```

Ok, zakladam teraz ze ./configure bylo OK:

```
root@solek:/users/vnull/freeradius-1.1.0# gmake
```

I po jakims czasie nam sie wysypie na rlm_sql_postgresql, wiec wykonujemy co nastepujace:

```
root@solek:/users/vnull/freeradius-1.1.0# cd src/modules/rlm_sql/drivers/rlm_sql_postgresql/
root@solek:/users/vnull/freeradius-1.1.0# vi Makefile
```

Modyfikujemy LDFLAGS tak by znalazlo sie tam “-L/opt/pgsql/lib” (bez cudzyslow).

Zapisujemy, wycohdzimy i wracamy do /users/vnull/freeradius-1.1.0, po czym dajemy (znow!):

```
root@solek:/users/vnull/freeradius-1.1.0# gmake
```

a potem (po pomyslniej kompilacji):

```
root@solek:/users/vnull/freeradius-1.1.0# gmake install
```

I chcialoby sie uwierzyc, ze mamy dzialajacego FreeRADIUSa... ale niestety tak nie jest. Dlaczego?

```
root@solek:/opt/freerad/lib# ldd rlm_sqlcounter.so
```

```
libssl.so.0.9.7 => (file not found)
libcrypto.so.0.9.7 => (file not found)
libnsl.so.1 => /lib/libnsl.so.1
libresolv.so.2 => /lib/libresolv.so.2
libsocket.so.1 => /lib/libsocket.so.1
librt.so.1 => /lib/librt.so.1
libpthread.so.1 => /lib/libpthread.so.1
libc.so.1 => /lib/libc.so.1
libmp.so.2 => /lib/libmp.so.2
libmd5.so.1 => /lib/libmd5.so.1
libscf.so.1 => /lib/libscf.so.1
libaio.so.1 => /lib/libaio.so.1
libdoor.so.1 => /lib/libdoor.so.1
libuutil.so.1 => /lib/libuutil.so.1
libm.so.2 => /lib/libm.so.2
```

```
root@solek:/opt/freerad/lib# ldd rlm_sql_postgresql.so
```

```
libpq.so.4 => (file not found)
libc.so.1 => /lib/libc.so.1
libm.so.2 => /lib/libm.so.2
```

Jak widac linker nie moze znalezc pewnych bibliotek:

```
libpg.so
libssl.so
libcrypto.so ( libcrypto.so jest wymagane przez libssl.so )
```

Wiec to naprawiamy szybko:

```
root@solek:/# for d in "/opt/pgsql/lib /opt/freerad/lib /usr/sfw/lib"; do crle -u -l $d done
```

Po tej operacji, wszystkie zaleznosci powinny zostac spelnione:

```
root@solek:/opt/freerad/lib# ldd *.so | grep -i not
```

```
root@solek:/opt/freerad/lib#
```

Czyli nic nie powinno nam sie pokazac ;)

Teraz probnie mozemy odpalic FreeRADIUSa (na defaultowym configu sie nam podniesie, aczkolwiek nic pozytecznego robic nie bedzie ...):

```
root@solek:/opt/freerad/lib# cd ../sbin; ./radiusd -X
```

```
[masa smiecia]
```

```
Listening on authentication *:1812
```

```
Listening on accounting *:1813
```

```
Ready to process requests.
```

Wlasnie to sie powinno pokazac, nie powinno byc segfault-ow czy tez informacji ze nie mozna wystartowac radiusd z powodu braku mozliwosci zaladowania jakiegos modulu dynamicznego.