libnss-pgsql - Name Service Switch module for use with PostgreSQL

Jörg Wendland <jorgland@sol.wohnheim.uni-ulm.de>
Bret Mogilefsky <mogul@gelatinous.com>

1.0.0

Abstract

This document covers setup and use of the Name Service Switch (NSS) Module `libnss-pgsql.so`. Its purpose is replace the flatfile user and group database in `/etc` with a relational database on a PostgreSQL server. It is highly configurable to fit onto most existing databases.

Contents

Chapter 1

Prerequisites

For installation instructions please see the files `README` and `INSTALL` in the source distribution.

To use this module you will need a PostgreSQL database containing some sort of user account information. See 'Database Schema' on page 7 for an example database.

Chapter 2

Module and Database Setup

2.1 required database structure

To use this module with a database you will need at least three tables in that database. One for account data (the information usually stored in `/etc/passwd`), one for group data (`/etc/group`) and another one storing information about groupmembership (there is a m:n relation between passwd and group so you need this weak entity). If you have an existing database you do not want to modify you can use views or table expressions (see 'Confile File' on page 8), of course.

2.1.1 tables and fields

There have to be some required fields in those tables. They are described below.

passwd table (see `getpwnam(3)` for more information)

- login name
- password
- numerical user id (uid)
- numerical primary group id (gid)
- login shell
- description (gecos)
- home directory

group table (see `getgrnam(3)`)

- group name
- group password
- numerical group id (gid)

An example database schema can be found in 'Database Schema' on page 7. See there for how to construct the groupmember table.

2.2 configuring the module

Having a working database, you must write your configfile `/etc/nss-pgsql.conf`. This file has a simple syntax:

```
LINE      := COMMENT | EMPTY | STATEMENT
COMMENT   := '#' <text>
EMPTY     := ''
STATEMENT := KEYWORD '=' <value>
```

With `KEYWORD` being one of the following (all are required):

`host` Hostname or IP address of PostgreSQL server.
`port` Port on which the Postmaster listens.
`database` The database name.
`login` Username for logging in to the PostgreSQL server.
`passwd` Password for logging in.
`passwdtable` Name of the table storing the account data.
`grouptable` Table storing group data.
`groupmembertable` Table storing the 'links' between account and group data.
`passwd_name` Name of field in `passwdtable` storing the user name.
`passwd_passwd` Field storing the password.
`passwd_uid` Field storing the numerical user id.
`passwd_dir` Field storing the `$HOME` directory.
`passwd_shell` Field storing the user's shell.
`passwd_gecos` Field storing additional information about this user, e.g. his full name and phone number.
`passwd_gid` Field storing the numerical primary group id.
`group_name` Name of the field in `grouptable` storing the group name.
`group_passwd` Field storing the groups password.
`group_gid` Field storing the group id.
`group_member` Name of field in `groupmembertable` storing the user name being member of the queried group.

Chapter 3

Name Service Switch Setup

WARNING: You can do serious damage to your system if you do not know what you are doing! If you are trying changes in the Name Service Switch for the very first time, consider doing that in a chroot environment or a test machine. Have at least an editor with `/etc/nsswitch.conf` open to revert your changes easily. You have been warned!

Setting up the Name Service Switch is fairly easy, given a working installation of the libnss-pgsql module. Just add an entry `pgsql` the `passwd` and `group` lines of `/etc/nsswitch.conf`. You can tune the behaviour with options between the entries to get a fully working system. Useful lines in `/etc/nsswitch.conf` would look like these:

```
passwd:          compat [SUCCESS=continue] pgsql
group:           compat [SUCCESS=continue] pgsql
```

This will make your libc look into the standard `passwd` and `group` files first, then lookup names via NIS (that is integrated in the libnss-compat modules) and then use libnss-pgsql. The option `[SUCCESS=continue]` ensures that all accounts or groups are retrieved when using the iteration functions `getpwent(3)` and `getgrent(3)`.

Appendix A

Examples

A.1 Database Schema

```
CREATE TABLE "groups" (
        "gid" serial NOT NULL,
        "name" character varying(16) NOT NULL,
        "passwd" character varying(20),
        PRIMARY KEY ("gid")
);

CREATE TABLE "accounts" (
        "uid" serial NOT NULL,
        "login" character varying(8) NOT NULL,
        "passwd" character varying(30) NOT NULL,
        "shell" character varying DEFAULT '/bin/bash' NOT NULL,
        "homedir" character varying NOT NULL,
        PRIMARY KEY ("login")
);

CREATE TABLE "usergroups" (
        "gid" int4 NOT NULL,
        "uid" int4 NOT NULL,
        PRIMARY KEY ("gid", "uid"),
        CONSTRAINT "ug_gid_fkey" FOREIGN KEY ("gid")
            REFERENCES "groups"("gid"),
        CONSTRAINT "ug_uid_fkey" FOREIGN KEY ("uid")
            REFERENCES "accounts"("uid")
);
```

A.2 Confile File

```
# database parameters
host             = 127.0.0.1
port             = 5432
database         = benutzer
login            = postgres
passwd           = foo

# the tables used (or tableexpressions!)
passwdtable      = accounts
grouptable       = groups
# the next line is wrapped for readability though
# this is not legal in a config file
groupmembertable = accounts JOIN usergroups ON
     accounts.uid=usergroups.uid JOIN groups ON
     usergroups.gid=groups.gid

# fields used in passwd functions
passwd_name      = login
passwd_passwd    = passwd
passwd_uid       = uid
passwd_dir       = homedir
passwd_shell     = shell
passwd_gecos     = login
passwd_gid       = uid

# fields used in group functions
group_name       = name
group_passwd     = passwd
group_gid        = gid
group_member     = login
```