HylaFAX Enterprise Guide for ODBC

iFAX Solutions, Inc.

Version 3.4, September 2019

Contents

1. Preface

HylaFAX Enterprise will utilize an ODBC connection for storing data regarding both inbound and outbound faxing. In the case of inbound faxing, it is possible to route incoming faxes using database rows which, presumably, some automated provisioning agent would be adding/populating.

In the case of outbound faxing, ODBC usage enables a data warehouse for reporting on fax traffic including final transmission status, bitrate, duration of transmission, the receiver's station identifier, etc.

With clever use of stored procedures and triggers some business logic can added to both transmission and reception of faxes, and this is yet another point of integration in addition to the client/server protocol that some customers use to get data out of HylaFAX. For instance, if a received fax needs to be injected into some customer-facing application, you could execute a stored procedure to capture the fax (which can be stored in the database) and move it to a new table/schema.

This guide will explain how to use PostgreSQL and MariaDB on CentOS 7 64-bit with HylaFAX Enterprise

2. Configure ODBC Database Server

2.1. PostgreSQL installation

```
# yum install postgresql-server postgresql-odbc

# postgresql-setup initdb

# systemctl enable postgresql
```

2.1.1. Configure PostgreSQL

Edit */var/lib/pgsql/data/pg_hba.conf*:

Add the authentication line **BEFORE** the default ident lines:

```
local   hylafax        hylafax                                 md5
host    hylafax        hylafax       127.0.0.1/32              md5
host    hylafax        hylafax       ::1/128                   md5

# If you need network access, use something like:
host    hylafax        hylafax       192.168.1.0/24            md5
```

The file is well commented.

Edit */var/lib/pgsql/data/postgresql.conf*

```
Important variables are:
    listen_addresses = 'localhost'
    max_connections = 512
    shared_buffers = 1024

max_connections needed are (minimum):
    2 * N + 1 + n + H
where:
    N is the number of channels
    1 is for faxq
    n is the number of simultaneous faxq job preparations (n <= N)
    H is the number of simultaneous hfaxd connections
```

These minimum settings do not take into account any other necessary connections from non-hylafax related processes (like any other tools/scripts, or anything that faxrcvd might do, etc).

2.1.2. Start PostgreSQL

```
# systemctl start postgresql
```

2.1.3. Create HylaFAX User/Database for PostgreSQL

```
# su - postgres
```

```
$ createdb hylafax
```

```
$ createuser -A -D -P hylafax
```
Password: ee

```
$ createlang plpgsql hylafax
```

```
$ exit
```

2.1.4. Import HylaFAX Schema

```
# psql -U hylafax hylafax < /var/spool/hylafax/ifax/sql/postgres.sql
```

2.1.5. Setup ODBC

Edit */etc/odbc.ini*

```
[hylafax]
Driver      = /usr/lib64/psqlodbc.so
Database    = hylafax
Server      = localhost
UserName    = hylafax
Password    = ee
ReadOnly    = No
Debug       = 0
CommLog     = 0
UseServerSidePrepare = 0
```

2.2. MariaDB installation

```
# yum install mariadb mariadb-server mysql-connector-odbc

# systemctl start mariadb

# systemctl enable mariadb
```

2.2.1. Create HylaFAX User/Database on MariaDB

```
# mysqladmin create hylafax

# mysql mysql
```

Execute these SQL queries:

```
GRANT ALL PRIVILEGES ON hylafax.* TO 'hylafax'@'localhost' IDENTIFIED BY 'ee';
```

If you need network access, use something like:

```
GRANT ALL PRIVILEGES ON hylafax.* TO 'hylafax'@'209.166.32.52' IDENTIFIED BY 'ee';
```

Update privileges:

```
FLUSH PRIVILEGES;
exit;
```

2.2.2. Import HylaFAX Schema

```
# mysql -uhylafax -pee hylafax < /var/spool/hylafax/ifax/sql/mysql.sql
```

2.2.3. Setup ODBC

Edit */etc/odbc.ini*

```
[hylafax]
Driver      = /usr/lib64/libmyodbc5.so
Database    = hylafax
Server      = localhost
User        = hylafax
Password    = ee
;ReadOnly   = No
```

3. Test the ODBC Connection

Test the connection using `isql hylafax`

```
[root@hylafax ~]# isql hylafax
+---------------------------------------+
| Connected!                            |
|                                       |
| sql-statement                         |
| help [tablename]                      |
| quit                                  |
|                                       |
+---------------------------------------+
SQL> SELECT count(*) FROM hylafax_statements;
+-----------------+
| count           |
+-----------------+
| 13              |
+-----------------+
SQLRowCount returns 1
1 rows fetched
SQL> quit
```

4. Configure HylaFAX Enterprise with ODBC

Create */var/spool/hylafax/etc/CONFIG/database* with the following content:

```
DataInitialize:    "odbc: dsn=hylafax"
#DataOption:       "odbc: username=hylafax"
#DataOption:       "odbc: password=ee"
DataSource:        "FaxRecvInfo: odbc"
DataSource:        "FaxAcctInfo: odbc"
```

Afterwards, configure your modem devices to include that file. For example, in */var/spool/hylafax/etc/CONFIG/boston* add:

```
Include:           etc/CONFIG/database
```

Next, restart `faxgetty` and/or `btgetty` proccesses in order to connect them to ODBC:

```
# for i in /var/spool/hylafax/FIFO.*; do echo Q > $i; done
```

If there is an ODBC connection problem, the faxgetty/btgetty processes will fail to start properly.

You can verify whether there were errors while connecting to the database by executing:

```
# grep -i odbc /var/log/messages
```

If HylaFAX is having difficulty connecting to ODBC, try uncommenting the 'DataOption' lines above.

Should you need to disable ODBC, simply comment out the Include line you added above.

5. Retrieving Data from ODBC

Here are sample SQL queries for retrieving records from the 'communication' table:

```
SELECT tstart, command, commid, jobid, mailaddr, dest, csi, npages, status
FROM communication WHERE dest = '12155558774';
```

```
SELECT tstart, commid, device, jobid, npages, status FROM communication
ORDER BY tstart DESC LIMIT 40;
```

A description of each column can be found in the *xferfaxlog* manpage.

6. Populating the Database

You can use the `xferfaxlog2db.php` script from http://people.ifax.com/~david/scripts/ to export your xferfaxlog data into the ODBC database. Only records not currently available in the database will be created.

7. Checking a job's final status

The HylaFAX job 'status' is available in the 'status' column of the 'communication' table. On success, the status field will be empty. A call that ended with an error will report the error in the 'status' column.

Please note that the */var/spool/hylafax/etc/xferfaxlog* data and the 'communication' database table contain call detail records. If you want to know the final status of a job, it's better to add that column to the database and then update it via the FaxNotify script.

For example, you'd add the following to */var/spool/hylafax/etc/FaxNotify*:

```
# Add final status (e.g. 'done', 'failed', etc) to the communication table
# See `man notify` for list of statuses
DB=hylafax
USER=hylafax
PASS=ee

case "$WHY" in
  blocked)    ;;
  requeued)   ;;
  *) echo "UPDATE communication SET finalstatus='$WHY' WHERE commid = '$COMMID'" | isql
$DB $USER $PASS
   ;;
esac
```

You must apply the following schema change for both PostgreSQL and MariaDB:

```
ALTER TABLE communication ADD finalstatus VARCHAR(20) DEFAULT NULL;
```

You must also apply the following schema change for MariaDB:

```
ALTER TABLE communication CHANGE tstart tstart TIMESTAMP DEFAULT CURRENT_TIMESTAMP;
```

Once you've applied these changes, you'll be able to query the database regarding jobs where the finalstatus is not null and get the true final status of the job.

```
SELECT * FROM communication WHERE command = 'SEND' AND finalstatus is not null;
```

You can use the following queries to retroactively set the `finalstatus` column:

```
UPDATE communication SET finalstatus='done' WHERE command = 'SEND' AND finalstatus is
null AND status is null;
```

```
UPDATE communication SET finalstatus='failed' WHERE command = 'SEND' AND finalstatus is
null AND status like '%too many%';
```