Using Conquest on LINUX – version 1.4.17d

The server core (**dgate.exe** = **dgate** under Linux) compiles and runs on Linux systems and Solaris. I develop primarily under Windows, but currently I test the code and scripts under the Ubuntu 10.10 64 bits server and Linux Mint 14 release in a virtual machine. The server also compiles and runs on a Raspberry Pi.

The Linux release of the server core works default with SqLite driver built in into the server (no ODBC). The DbaseIII driver is also supported. Piotr Filipczuk has added a PostGresSQL driver. Since version 1.4.15, a native MySQL interface also can be used. The graphical user interface has not been ported to Linux, but the WEB interface is provided. In this version, most options have been well tested – it is a stable release.

To use the server, one needs a valid version of the configuration files and put them in the same directory as the dgate executable. The easiest way to do this is to unpack conquestlinux1417d.gz with “tar xvf conquestlinux1417d.gz” that contains:

dgate	Server executable
dicom.ini	Server configuration (for sqlite)
dicom.ini.dbase	Template configuration for built in dbase driver
dicom.ini.sqlite	Template configuration for built in sqlite driver
dicom.ini.postgres	Template server configuration for postgres
dicom.ini.mysql	Template server configuration for mysql
dicom.ini.www	Template configuration for web server
dicom.sql	Database configuration (normalized)
dicom.sql.dbase	Template database configuration (denormalized)
dicom.sql.postgres	Template database configuration (normalized)
dicom.sql.sqlite	Template database configuration (normalized)
dicom.sql.mysql	Template database configuration (normalized)
acrnema.map	Configuration of know DICOM providers
dgate.dic	DICOM dictionary
dgatesop.lst	Accepted data and services (with jpeg)
dgatesop.lst.nojpg	Template accepted data and services (no jpeg)
dgatesop.lst.withjpg	Template accepted data and services (with jpeg)
maklinux	Shell script to compile and install dgate
maklinux_postgres	Idem but using Postgres as database
maklinux_sqlite	Idem but using SqLite as database
maklinux_mysql	Idem but using MySQL as database
maklinux_dbase	Idem but using DbaseIII as database
maklinux.bat	Used by mvh for collecting files in Linux distribution.
Makefile	Sample Makefile (unused)
DicomConformance_FilesLST_Changes.pdf	Part of manual
windowsmanual.pdf	Part of manual
linuxmanual.pdf	Part of manual (this file)
data/dbase/	Place for database (SqLite or DbaseIII)
data/samples/	Sample images
data/samples/HEAD_EXP_00097038/0001_002000_892665661.v2	
data/samples/HEAD_EXP_00097038/0001_003000_892665662.v2	

Plus all the sources needed to build the server: aaac.cxx, aaac.hpp, aarj.cxx, aarj.hpp, aarq.cxx, aarq.hpp, amap.cpp, array.tcc, array.thh, base.hpp, buffer.cxx, buffer.thh, cctypes.h, constant.h, dbsql.cpp, deivr.cxx, deivr.hpp, device.cpp, dgate.cpp, dgate.hpp, dgatefn.cpp, dicom.hpp, dimsec.cxx, dimsec.hpp, dimsen.cxx, dimsen.hpp, dprintf.cpp, dprintf.hpp, endian.cpd, endian.cxx, endian.hpd, endian.hpp, farray.thh, filepdu.cxx, flpdu.cxx, flpdu.hpp, gpps.cpp, gpps.hpp, lex.cpp, lex.hpp, loadddo.cpp, nkiqrsop.cxx, nkiqrsop.hpp, npipe.cpp, npipe.hpp, odbci.cpp, odbci.hpp, parse.cpp, pdata.cxx, pdata.hpp, pdu.cxx, pdu.hpp, queue.tcc, pqueue.thh, qrsop.cxx, qrsop.hpp, regen.cpp, rtc.cxx, rtc.hpp, safemem.h, socket.cxx, socket.hpp, sqlite3.c, sqlite3.h, storage.cxx, storage.hpp, total.cxx, trnsyn.cxx, uniq.cxx, unixsock.h, util.cxx, util.h, verify.cxx, verify.hpp, version.h, vrtosql.cpp, wintypes.hpp, xvgifwr.c

Plus the sources of a modified version of the IJG code: jpeg-6c, that allows run-time selection of bit depth (done by Bruce Barton), the jasper-1.900.1-6ct library (also modified by Bruce Barton), and lua_5.1.4 (original but with one macro renamed for Ubuntu 10.10 compatibility).

Plus ZeroBraneStudio integration files, webserver sample files, sample lua scripts and a placeholder clibs folder for binary modules used by lua scripting.

INSTALLATION

Prerequisites: 1) a running Linux system. 2) sudo installed and enough rights to perform sudo. If not, the script will not be able to install the server as web service for apache and you need to copy the files by hand. 3) Installed G++; 4) Check /usr/lib/cgi-bin/ exists and is enabled in apache2.conf.

These packages needed to be installed in a plain Linux Mint14 release for a release using SQLite:

```
sudo apt-get update
sudo apt-get install g++
sudo apt-get install apache2
```

The following steps illustrate a minimal installation (maklinux_xxx may need adjustments for your local installation):

```
(ps)ftp the gz file to linux system (e.g., into your home directory)    get the files there
mkdir conquest
cd conquest                                                              to there
tar xvf ../conquestlinux1417c.tar.gz                                     unpack all files

cd jpeg-6c                                                               make the IJG library
./configure
sudo make
sudo make install
cd  ..

cd jasper-1.900.1-6ct                                                    make the jasper library
./configure
sudo make
sudo make install
cd ..
```

```
./maklinux                                                              compile and install web access
(or: maklinux_dbase, or: maklinux_postgres, or: maklinux_mysql)

dgate -v -r                                                             regenerate the database
dgate -v &                                                              run the server (for ever)
```

Now the server should be running and localhost/cgi-bin/dgate should provide a working web interface.

ZerobraneStudio IDE

To install and use ZeroBrane Studio with the conquest DICOM server under Linux, take these steps. First download ZeroBraneStudioEduPack-xxx-linux.sh. Then in a command prompt run:

```
chmod 777  ZeroBraneStudioEduPack-xxx-linux.sh
./ZeroBraneStudioEduPack-xxx-linux.sh
```

After installation is done run ZeroBrane Studio from the command prompt as “sudo zbstudio” and run the install script /dicomserver/ZeroBraneStudio/install.lua in ZeroBrane Studio as described in this file. After running the conquest install script as root, ZeroBraneStudio can be run as a normal user.

As, in linux, the socket library is not linked into the dgate binary, you have to copy (for a 64 bits conquest) “/opt/zbstudio/bin/X64/clibs/socket/core.so to” “/dicomserver/clibs/socket” or (for a 32 bits conquest): “/opt/zbstudio/bin/X86/clibs/socket/core.so” to “/dicomserver/clibs/socket”.

Note: to enable the use of shared libraries and io.popen, I had to add -DLUA_USE_DLOPEN and -DLUA_USE_POSIX to the gcc line compiling lua/all.c in all ./maklinuxXXX scripts.

Some of the scripts make use of external binaries:

I installed cmake and cmake-qt-gui to build and install nifty_reg
Then I installed mricron as 'small' nifti viewer
and finally p7zip.full to enable use of 7za as decompressor

CONFIGURATION

Configuration files under Windows and Linux are the same except for the use of a forward slash instead of back slash in directory paths. The following essential entries are therefore different for Linux (these are the defaults):

```
SQLServer        =        ./data/dbase/conquest.db3
MAGDevice0       =        ./data/
```

See the Windows manual for more details about the configuration files (you need at least to edit **acrnema.map** to define DICOM systems that will be retrieving information from your server). All configurations options in **dicom.ini** (e.g., for DICOM routing) are listed in **windowsmanual.pdf**. You probably also need to edit the web server configuration file **/usr/lib/cgi-bin/dicom.ini** to set the correct IP address of the machine. If not the web server will only partly function.

After copying the files, if needed, regenerate the database with "conquest/dgate –v –r" then run the server with "conquest/dgate –v" or "conquest/dgate -^serverstatus.log". NOTE: regeneration is only needed after an upgrade if **dicom.sql** is updated. If you want to avoid regeneration do NOT replace **dicom.sql**

To automatically start the server at boot time create a shell script in /etc/rc5.d called Z99Conquest, that contains, e.g.,:

```
cd /home/marcel/conquest
dgate -^serverstatus.log
```

The building process for the server was tested with gcc 3.3.5, Ubuntu 8.10 and on Solaris 10. Both 32 and 64 bit OS's are supported. Warnings (many 'multi-character character constant' and one 'fattach is not implemented and will always fail') are produced but these do not impact server operation.

Shell script **maklinux** is available that compiles dgate, copies it to the cgi-bin directory for web access, and sets up (*overwrites*) **dicom.ini** and **dicom.sql** for SqLite operation. The SqLite driver is built-in.

Also a shell script **maklinux_dbase** is available that compiles dgate with dbaseIII support and copies it to the cgi-bin directory for web access. It also sets up (*overwrites*) **dicom.ini** and **dicom.sql** for dbaseIII operation. The dbaseIII driver is built-in.

Also a shell script **maklinux_mysql** is available that compiles dgate with MySQL support and copies it to the cgi-bin directory for web access. It also sets up (*overwrites*) **dicom.ini** and **dicom.sql** for SqLite operation. It requires creating a DB called "conquest" with phpmyadmin and installing libmysqlclientdev with: "*apt-get install libmysqlclient-dev*" before running maklinux_mysql. These are the settings in dicom.ini for MySQL:

```
SQLHost          = localhost
SQLServer        = conquest
Username         = root
Password         =
Mysql            = 1
```

```
DoubleBackSlashToDB = 1
```

Also a shell script **maklinux_postgres** is available that compiles dgate and copies it to the cgi-bin directory for web access. It also makes sure the postgres shared libraries can be found, and sets up (*overwrites*) **dicom.ini** and **dicom.sql** for PostGres operation. The PostGres system (I used postgresql-8.1beta1.tar.bz2) most be setup to the defaults, and a database named '*conquest'* made. For postgres to work you need to check some values in dicom.ini (using the default postgres account assuming password postgres, note that parameter '*SQLServer*' sets the database to conquest). A copy from **dicom.ini.postgres** to **dicom.ini** would set the following values:

```
SQLHost                 = localhost
SQLServer               = conquest
Username                = postgres
Password                = postgres
PostGres                = 1
DoubleBackSlashToDB = 1
UseEscapeStringConstants = 1
```

It is advised to use a normalized database (as defined in **dicom.sql**) for postgres operation, e.g., by copying **dicom.sql.postgres** to **dicom.sql** and a denormalized database for DbaseIII, e.g., by copying **dicom.sql.dbase** to **dicom.sql** . The following are donated scripts by Mark Pearson for start/stop and rotating logfiles:

To install this script (it is in the distribution as conquest-pacs.sh) do:

```
sudo cp conquest-pacs.sh /etc/init.d/
sudo chmod 755 /etc/init.d/conquest-pacs.sh
sudo apt-get install authbind
sudo /etc/init.d/conquest-pacs.sh start
```

```
#!/bin/bash
#
# conquest-pacs.sh      SysV init script for Conquest PACS.
#
#       Written by Miquel van Smoorenburg <miquels>.
#       Modified for Debian GNU/Linux by Ian Murdock <imurdock>.
#       Customized for Conquest by Mark Pearson <markp>
#
#       HOME and PACSUSER should be the only variables that may need to be
modified.
#
PATH=/sbin:/bin:/usr/sbin:/usr/bin

# Modify HOME to suit your environment.
HOME=/usr/local/conquest
# This is the user to run as. Modify it if you don't use username conquest.
PACSUSER=conquest

DAEMON=$HOME/dgate
INI=$HOME/dicom.ini
NAME=conquest_pacs.sh
```

```
# All defaults here will be overridden by values from $HOME/dicom.ini
STATUSLOG=$HOME/serverstatus.log
PORT=104
DESC="Conquest PACS Server"

STOPPACS=$HOME"/dgate --quit:"
STARTAS=$DAEMON

test -f $DAEMON || echo "Cannot find $DAEMON" exit 0
test -f $INI || echo "Cannot find $INI" exit 0

set -e

if  grep "TCPPort" $INI > /dev/null ; then
        PORT=`egrep -i '^*TCPPort *= ' $INI | sed 's/\r//' | awk '{ print  $3}'`
fi

if [ $PORT -le 1024 ]; then
        test -f /usr/bin/authbind || echo "authbind is needed for access to ports <
1024" exit 0
        STARTAS="/usr/bin/authbind "
fi

if  grep -is "^ *StatusLog" $INI > /dev/null ; then
        STATUSLOG=`egrep -i '^*StatusLog' $INI | sed 's/\r//'  | awk '{ print
$3}'`
fi


PIDFILE=/var/run/$NAME.$PORT.pid
if [ $STARTAS = $DAEMON ]; then
        ARGS=" -^$STATUSLOG"
else
        ARGS="$DAEMON -^$STATUSLOG"
fi

case "$1" in
  start)
        if [ -f $HOME/disable_autostart ]; then
                echo "Not starting $DESC: disabled via $HOME/disable_autostart"
                exit 0
        fi


        echo -n "Starting $DESC: "
        start-stop-daemon --start --quiet --pidfile $PIDFILE \
                --chuid $PACSUSER --chdir $HOME --exec $DAEMON \
                --startas $STARTAS --background -- $ARGS
        echo "$NAME."
        ;;
  stop)
        echo -n "Stopping $DESC: "
        cd $HOME
        $STOPPACS

        start-stop-daemon --oknodo --stop --quiet --pidfile $PIDFILE \
                --exec $DAEMON -- $ARGS
        echo "$NAME."
        echo
        ;;

  restart|force-reload)
        echo -n "Restarting $DESC: "
        start-stop-daemon --stop --oknodo --quiet --pidfile $PIDFILE  \
                --exec $DAEMON -- $ARGS
        sleep 1
        start-stop-daemon --start --quiet --pidfile $PIDFILE \
                --chuid conquest --chdir $HOME --exec $DAEMON -- $ARGS
        echo "$NAME."
        ;;
  *)
        N=/etc/init.d/$NAME
        echo "Usage: $N {start|stop|restart|force-reload}" >&2
        exit 1
        ;;
esac

exit 0
```

For security reasons I have added a user "conquest" and the package authbind to allow access to priveleged ports. I added the following entries to dicom.ini:
HomeDir = /usr/local/conquest
StatusLog = /var/log/conquest/NMPACS.serverstatus.log
TroubleLog = /var/log/conquest/NMPACS.PacsTrouble.log

The file /etc/cron.weekly/conquest_rotate does weekly log rotation for me.

```
#!/bin/bash

# conquest_rotate      Cron script to rotate conquest log files.
#     Keep files for 365 days
#     Read filenames from dicom.ini
#
#
#                Written by Mark Pearson 20070711 <markp>.
#

# Modify this line to suit your environment
HOMES=(/usr/local/conquest /usr/local/conquest-icon)
for i in ${HOMES[@]}; do

        INI=${i}/dicom.ini
        STATUSLOG=${i}/serverstatus.log
        TROUBLELOG=${i}/PacsTrouble.log

        set -e

# defaults will be overridden by values from ${i}/dicom.ini
        if  grep -is "^ *StatusLog" $INI > /dev/null ; then
                STATUSLOG=`egrep -i '^*StatusLog' $INI | sed 's/\r//'  | awk
'{ print  $3}'`
        fi
        if  grep -is "^ *TroubleLog" $INI > /dev/null ; then
                TROUBLELOG=`egrep -i '^*TroubleLog' $INI | sed 's/\r//'  | awk
'{ print  $3}'`
```

```
        fi

        if [ -s $TROUBLELOG ]; then
                savelog -p -c 365 -n -q $TROUBLELOG
        fi

        if [ -s $STATUSLOG ]; then
                savelog -p -c 365 -n -q $STATUSLOG
        fi
done
```

This copes with multiple pacs instances on the same host. The advantage of using savelog is that old logfiles are compressed. It should be quite simple to edit the files to have executable or log in /opt. Also, don't forget to set the appropriate file permissions for the user that runs conquest.

Finally, Here are the command lines to compile the server under OS X xcode using 10.4u sdk on a PowerPC:

g++ -isysroot /Developer/SDKs/MacOSX10.4u.sdk -arch ppc -Wno-multichar -I/usr/local/mysql/include -L/usr/local/mysql/lib -DDARWIN -DUSEMYSQL -DHAVE_LIBJASPER -DHAVE_LIBJPEG -DB_DEBUG -o dgate total.cxx -lpthread -lgcc_s.10.4 -lstdc++.6 -lmysqlclient -lz

And to compile under SOLARIS 10:

/usr/sfw/bin/g++ -DUNIX -DNATIVE_ENDIAN=1 -DHAVE_LIBJASPER -DHAVE_LIBJPEG -DSOLARIS total.cxx -o dgate -lpthread -lsocket -lnsl -lposix4