Using Conquest on LINUX – version 1.4.19 Feb 8, 2017

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 Linux Ubuntu 16.04 in a virtual machine. I also had the server compiled on a Raspberry Pi but without a lot of the extras.

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 PostGresQL driver. The 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 **dicomserver1419.zip** with "unzip dicomserver1419".

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.

These packages needed to be installed in a plain Linux system (e.g. Mint or Ubuntu) for a release using SQLite or DbaseIII:

```
sudo apt-get update
sudo apt-get install g++
sudo apt-get install apache2
sudo a2enmod cgi
sudo service apache2 restart
```

The following steps illustrate a minimal installation:

```
(ps)ftp the zip file to linux system (e.g., into your home directory)   get the files there
mkdir conquest
cd conquest                                                             to there
unzip ../dicomserver1419.zip                                            unpack all files
chmod 777 maklinux

./maklinux                                                              compile and install web access
choose option 3                                                         SqLite

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.

To install with Postgres as database, these commands are needed to install and setup Postgres:

sudo apt-get install libpq-dev	Postgres development tools
sudo apt-get install postgresql	Postgres database
sudo su	become superuser
su – postgres	become postgres user
psql	set the passwork to postgres
\password	
postgres	
postgres	
\q	
createdb conquest	create database conquest
exit	
exit	
./maklinux	compile and install web access
choose option 2	Postgres

The build process always often a few error messages that can be ignored:

/usr/bin/install: cannot create regular file '/usr/local/man/man1/cjpeg.1': No such file or directory

Makefile:200: recipe for target 'install' failed

mkdir: cannot create directory ‘data/dbase’: File exists

During database creation (dgate -v -r) there can be error messages about non-existing databases, e.g. for postgress:

```
osboxes@osboxes:~/Desktop/distribution$ ./dgate -v -r
Regen Database
Step 1: Re-intialize SQL Tables
*** ERROR:  relation "dicomworklist" does not exist
LINE 1: SELECT DICOMWorkList.PatientID FROM DICOMWorkList
                                            ^

Dropping Existing tables (if-any)
Worklist is empty
Dropping worklist
*** ERROR:  table "dicomworklist" does not exist
***Failed PGSQLExec : DROP TABLE DICOMWorkList
….
***Error: ERROR:  table "uidmods" does not exist


WorkList Database
```

Patient Database

Study Database

Series Database

Image Database

Step 2: Load / Add DICOM Object files

Regen Device 'MAG0'

[Regen] ./data/0009703828/1.3.46.670589.5.2.10.2156913941.892665339.860724_0001_002000_14579035620000.dcm

-SUCCESS

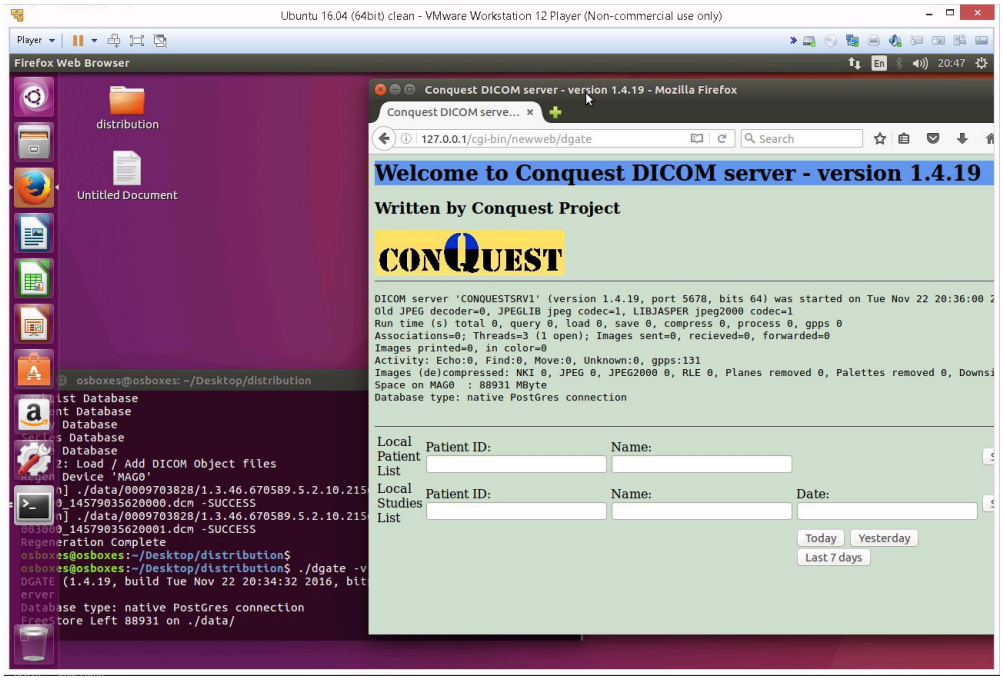
[Regen] ./data/0009703828/1.3.46.670589.5.2.10.2156913941.892665339.860724_0001_003000_14579035620001.dcm -SUCCESS

Regeneration Complete

osboxes@osboxes:~/Desktop/distribution$ ./dgate -v

DGATE (1.4.19, build Tue Nov 22 20:34:32 2016, bits 64) is running as threaded server

Database type: native PostGres connection



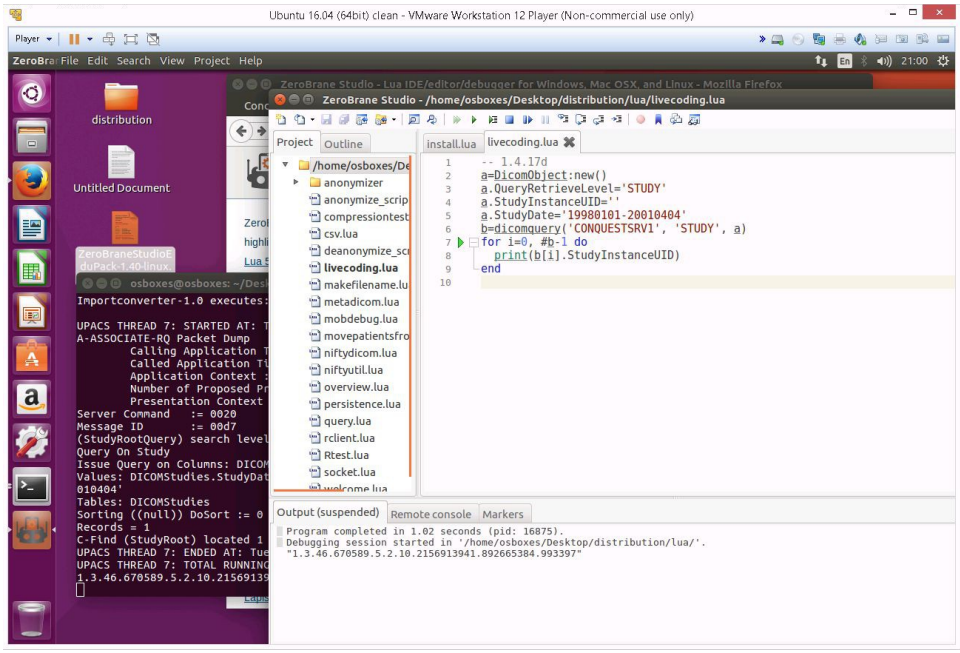
Conquest in action on Ubuntu16.04, with web interface

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
sudo ./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.



Integration with Zerobrane studio

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.

Also MySQL support is provided. 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
```

The PostGres system can 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 nconquest-pacs.sh) do:

```
sudo cp nconquest-pacs.sh /etc/init.d/
sudo chmod 755 /etc/init.d/nconquest-pacs.sh
sudo apt-get install authbind
sudo /etc/init.d/nconquest-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 (not recently tested):

```
 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
```