

QGIS Application - Bug report #13271

QGIS fails to launch or hang while exiting due to thread issue with OWS servers

2015-08-26 08:28 PM - Mathieu Pellerin - nIRV

Status:	Closed	
Priority:	Severe/Regression	
Assignee:	Radim Blazek	
Category:	Browser	
Affected QGIS version:	master	Regression?: No
Operating System:	Ubuntu	Easy fix?: No
Pull Request or Patch supplied:	No	Resolution: fixed/implemented
Crashes QGIS or corrupts data:	No	Copied to github as #: 21331

Description

For a while now, my QGIS master builds would either fail to start due to a threading error. If it did start, the QGIS process would hang forever upon exit.

This morning, I ran a debug build to try and diagnose the issue. The debug output led me to think that it was WMS related (some sort of network check upon launching QGIS). After removing all my saved WMS servers, launching QGIS worked without threading warnings, and the QGIS process terminated properly.

Here's the relevant part of the debug output:

```
src/providers/wms/qgswmscapabilities.cpp: 1869: (downloadCapabilities) [5ms] [thread:0x66279e0] entering.
src/providers/wms/qgswmscapabilities.cpp: 1872: (downloadCapabilities) [0ms] [thread:0x66279e0] url =
http://www.ngd.la/ngd_public/wms?
src/providers/wms/qgswmscapabilities.cpp: 1886: (downloadCapabilities) [0ms] [thread:0x66279e0] getcapabilities:
http://www.ngd.la/ngd_public/wms?SERVICE=WMS&REQUEST=GetCapabilities
Warning: QObject: Cannot create children for a parent that is in a different thread.
(Parent is QgsNetworkAccessManager(0x2451bf0), parent's thread is QThread(0x2054420), current thread is
QThread(0x66279e0)
./qgis[0x4c7ced]
./qgis[0x4c7ef9]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(qt_message_output(QtMsgType, char const*)+0x31)[0x7f79b76f9991]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(+0x71df9)[0x7f79b76f9df9]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(qWarning(char const*, ...)+0xa1)[0x7f79b76fa071]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(+0x198a10)[0x7f79b7820a10]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(QObject::QObject(QObjectPrivate&, QObject*)+0xb8)[0x7f79b7829c28]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(QIODevice::QIODevice(QIODevicePrivate&, QObject*)+0x9)[0x7f79b7797949]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkReply::QNetworkReply(QNetworkReplyPrivate&,
QObject*)+0x9)[0x7f79b66b88c9]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(+0x7e8c8)[0x7f79b66bb8c8]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkAccessManager::createRequest(QNetworkAccessManager::Operation,
QNetworkRequest const&, QIODevice*)+0x2eb)[0x7f79b669f2eb]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(QgsNetworkAccessManager::createRequest(QNetworkAccessManager::Operation, QNetworkRequest const&, QIODevice*)+0x27b)[0x7f79b919c311]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkAccessManager::get(QNetworkRequest const&)+0x14)[0x7f79b669c214]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/qgis/plugins/libwmsprovider.so(QgsWmsCapabilitiesDownload::downloadCapabilities()
dCapabilities()+0x394)[0x7f793a5fe74a]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/qgis/plugins/libwmsprovider.so(QgsWMSConnectionItem::createChildren()+0x2a8)[0x7f793a66472e]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(QgsDataItem::populate(bool)+0x14d)[0x7f79b90c0df7]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(QgsDataItem::populate(bool)+0x14d)[0x7f79b90c0df7]
```

```

/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/qgis/plugins/libowsprovider.so(QgsOWSConnectionItem::createChildren()+0x4ee)[0x7f79440acc4e]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(QgsDataItem::runCreateChildren(QgsDataItem*)+0xda)[0x7f79b90c1050]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(+0x2f2826)[0x7f79b90cd826]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(+0x2f26f1)[0x7f79b90cd6f1]
src/core/qgsmessagelog.cpp: 45: (logMessage) [3ms] [thread:0x65ebfb0] 2015-08-27T10:15:15 Qt[1] QObject: Cannot create
children for a parent that is in a different thread.
(Parent is QgsNetworkAccessManager(0x2451bf0), parent's thread is QThread(0x2054420), current thread is
QThread(0x65ebfb0)
./qgis[0x4c7ced]
./qgis[0x4c7ef9]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(qt_message_output(QtMsgType, char const*)+0x31)[0x7f79b76f9991]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(+0x71df9)[0x7f79b76f9df9]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(qWarning(char const*, ...)+0xa1)[0x7f79b76fa071]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(+0x198a10)[0x7f79b7820a10]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(QObject::QObject(QObjectPrivate&, QObject*)+0xb8)[0x7f79b7829c28]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(QIODevice::QIODevice(QIODevicePrivate&, QObject*)+0x9)[0x7f79b7797949]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkReply::QNetworkReply(QNetworkReplyPrivate&,
QObject*)+0x9)[0x7f79b66b88c9]
src/providers/wms/qgswmscapabilities.cpp: 1872: (downloadCapabilities) [0ms] [thread:0x6619e00] url =
http://64.91.228.155:8181/geoserver/Infrastructure/wms?version=1.1.0&layers=Infrastructure:map_percentage_of_houses_with_electricity&s
electricity&styles=&bbox=211418.14440275182,1144592.2830999997,784614.8614973782,1625604.7785997128&width=512&height=429
height=429&srs=EPSG:32648&
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(+0x7e8c8)[0x7f79b66bb8c8]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkAccessManager::createRequest(QNetworkAccessManager::Operation,
QNetworkRequest const&, QIODevice*)+0x2eb)[0x7f79b669f2eb]
src/providers/wms/qgswmscapabilities.cpp: 1886: (downloadCapabilities) [0ms] [thread:0x6619e00] getcapabilities:
http://64.91.228.155:8181/geoserver/Infrastructure/wms?version=1.1.0&layers=Infrastructure:map_percentage_of_houses_with_electricity&s
electricity&styles=&bbox=211418.14440275182,1144592.2830999997,784614.8614973782,1625604.7785997128&width=512&height=429
height=429&srs=EPSG:32648&SERVICE=WMS&REQUEST=GetCapabilities
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(QgsNetworkAccessManager::createRequest(QNetworkAcces
etworkAccessManager::Operation, QNetworkRequest const&, QIODevice*)+0x27b)[0x7f79b919c311]
Warning: QObject: Cannot create children for a parent that is in a different thread.
(Parent is QgsNetworkAccessManager(0x2451bf0), parent's thread is QThread(0x2054420), current thread is
QThread(0x6619e00)
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkAccessManager::get(QNetworkRequest const&)+0x14)[0x7f79b669c214]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/qgis/plugins/libwmsprovider.so(QgsWmsCapabilitiesDownload::downloadCapabilitie
dCapabilities()+0x394)[0x7f793a5fe74a]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/qgis/plugins/libwmsprovider.so(QgsWMSConnectionItem::createChildren()+0x2a8)[0x7f793a66472e]
n()+0x2a8)[0x7f793a66472e]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(QgsDataItem::runCreateChildren(QgsDataItem*)+0xda)[0x7f79b90c1050]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(+0x2f2826)[0x7f79b90cd826]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(+0x2f26f1)[0x7f79b90cd6f1]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(+0x6fb70)[0x7f79b76f7b70]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(+0x7c6ff)[0x7f79b77046ff]
./qgis[0x4c7ced]
./qgis[0x4c7ef9]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(qt_message_output(QtMsgType, char const*)+0x31)[0x7f79b76f9991]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(+0x71df9)[0x7f79b76f9df9]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(qWarning(char const*, ...)+0xa1)[0x7f79b76fa071]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(+0x198a10)[0x7f79b7820a10]

```

```
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(QObject::QObject(QObjectPrivate&, QObject*)+0xb8)[0x7f79b7829c28]
/usr/lib/x86_64-linux-gnu/libQtCore.so.4(QIODevice::QIODevice(QIODevicePrivate&, QObject*)+0x9)[0x7f79b7797949]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkReply::QNetworkReply(QNetworkReplyPrivate&,
QObject*)+0x9)[0x7f79b66b88c9]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(+0x7e8c8)[0x7f79b66bb8c8]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkAccessManager::createRequest(QNetworkAccessManager::Operation,
QNetworkRequest const&, QIODevice*)+0x2eb)[0x7f79b669f2eb]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(QgsNetworkAccessManager::createRequest(QNetworkAccessManager::Operation, QNetworkRequest const&, QIODevice*)+0x27b)[0x7f79b919c311]
/usr/lib/x86_64-linux-gnu/libQtNetwork.so.4(QNetworkAccessManager::get(QNetworkRequest const&)+0x14)[0x7f79b669c214]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/qgis/plugins/libwmsprovider.so(QgsWmsCapabilitiesDownload::downloadCapabilities()dCapabilities()+0x394)[0x7f793a5fe74a]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/qgis/plugins/libwmsprovider.so(QgsWMSConnectionItem::createChildren()+0x2a8)n()+0x2a8)[0x7f793a66472e]
/home/webmaster/dev/cpp/QGIS/bm-qtcreator/output/lib/libqgis_core.so.2.11.0(QgsDataItem::runCreateChildren(QgsDataItem*)+0xda)[0x7f79b90c1050]
0xda)[0x7f79b90c1050]
```

The thread warning (Warning: QObject: Cannot create children for a parent that is in a different thread) appears when QGIS is trying to connect to my saved WMS servers.

It's a pretty serious issue, esp. so as it leaves active processes eating CPU% for as long as the users don't manually kill the process.

(I have no idea if this is a regression in master only)

Related issues:

Related to QGIS Application - Bug report # 14401: QGIS Crashes if WFS credential...

Closed

2016-03-01

History

#1 - 2015-08-26 09:35 PM - Mathieu Pellerin - nIRV

- Category changed from Web Services clients/WMS to Browser

After further investigation, the problem is with the *browser panel*.

If WMS server nodes are expanded when QGIS is launched, it will trigger a server capability / layer check, which creates a threading warning and lead to zombie processes upon exit, and can prevent QGIS for launching altogether.

If a WMS server node is closed, and expanded during a QGIS session, it'll also trigger a thread warning.

There must be a problem with how the browser handles the network requests to WMS servers.

#2 - 2015-08-27 03:50 AM - Radim Blazek

There is a note about the warning <https://github.com/qgis/QGIS/blob/1dfe005/src/providers/wms/qgswmscapabilities.cpp#L1887>

The problem was with unfinished requests. I have fixed that in commit:c285ed169, but maybe too complicated. It might be enough to move (moveToThread) mCapabilitiesReply to main thread and set its parent to QgsNetworkAccessManager::instance() which probably aborts all request when destroyed.

The fix has to be done also for WFS and WCS.

The crash on startup: it happens randomly if more WMS servers are expanded:

```
#0 0x00007f3ca9476107 in __GI_raise (sig=sig@entry=6) at ../nptl/sysdeps/unix/sysv/linux/raise.c:56
#1 0x00007f3ca94774e8 in __GI_abort () at abort.c:89
#2 0x00007f3ca94b4204 in __libc_message (do_abort=do_abort@entry=1, fmt=fmt@entry=0x7f3ca95a6fe0 "**** Error in '%s': %s: 0x%s ***\n")
    at ../sysdeps/posix/libc_fatal.c:175
#3 0x00007f3ca94b99de in malloc_printerr (action=1, str=0x7f3ca95a71c8 "double free or corruption (fasttop)", ptr=<optimized out>)
    at malloc.c:4996
#4 0x00007f3ca94ba6e6 in _int_free (av=av@entry=0x7f3c18000020, p=<optimized out>, p@entry=0x7f3c180111e0,
have_lock=have_lock@entry=1)
    at malloc.c:3840
#5 0x00007f3ca94b710 in _int_realloc (av=av@entry=0x7f3c18000020, oldp=oldp@entry=0x7f3c180111e0, oldsize=oldsize@entry=80,
nb=nb@entry=144) at malloc.c:4340
#6 0x00007f3ca94bd7b9 in __GI___libc_realloc (oldmem=0x7f3c180111f0, bytes=128) at malloc.c:3029
#7 0x00007f3cac86417f in QListData::realloc (this=this@entry=0x7f3c1400c850, alloc=13) at tools/qlist.cpp:221
#8 0x00007f3cac864230 in QListData::append (this=0x7f3c1400c850, n=n@entry=1) at tools/qlist.cpp:243
#9 0x00007f3cac86428a in QListData::append (this=<optimized out>) at tools/qlist.cpp:253
#10 0x00007f3cab7d484d in QList<QPair<QByteArray, QByteArray>>::append (this=this@entry=0x7f3c1400c850, t=...)
    at ../../include/QtCore/../../src/corelib/tools/qlist.h:519
#11 0x00007f3cab80dbfd in operator>><QPair<QByteArray, QByteArray>> > (s=..., l=...)
    at ../../include/QtCore/../../src/corelib/io/qdatastream.h:253
#12 0x00007f3cab80d6d8 in QNetworkCacheMetaDataPrivate::load (in=..., metaData=...) at access/qabstractnetworkcache.cpp:395
#13 0x00007f3cab80d849 in operator>> (in=..., metaData=...) at access/qabstractnetworkcache.cpp:357
#14 0x00007f3cab8118c7 in QCacheltem::read (this=this@entry=0x28af498, device=device@entry=0x7f3c1eff1420,
readData=readData@entry=false)
    at access/qnetworkdiskcache.cpp:710
#15 0x00007f3cab811b64 in QNetworkDiskCache::fileMetaData (this=this@entry=0x2a1d2e0, fileName=...) at
access/qnetworkdiskcache.cpp:390
#16 0x00007f3cab811c1e in QNetworkDiskCache::metaData (this=0x2a1d2e0, url=...) at access/qnetworkdiskcache.cpp:373
#17 0x00007f3cab7f457f in QNetworkAccessHttpBackend::loadFromCacheIfAllowed (this=this@entry=0x7f3c1400c520, httpRequest=...)
    at access/qnetworkaccesshttpbackend.cpp:246
#18 0x00007f3cab7f83cd in QNetworkAccessHttpBackend::postRequest (this=0x7f3c1400c520) at access/qnetworkaccesshttpbackend.cpp:447
#19 0x00007f3cab7ee3ae in QNetworkAccessBackend::start (this=0x7f3c1400c520) at access/qnetworkaccessbackend.cpp:426
#20 0x00007f3cab808b7f in QNetworkReplyImplPrivate::_q_startOperation (this=this@entry=0x7f3c1400c060) at
access/qnetworkreplyimpl.cpp:93
#21 0x00007f3cab809380 in QNetworkReplyImplPrivate::setup (this=this@entry=0x7f3c1400c060,
op=op@entry=QNetworkAccessManager::GetOperation, req=..., data=data@entry=0x0) at access/qnetworkreplyimpl.cpp:374
#22 0x00007f3cab7ea38c in QNetworkAccessManager::createRequest (this=0x299c1c0, op=QNetworkAccessManager::GetOperation, req=...,
outgoingData=0x0) at access/qnetworkaccessmanager.cpp:1010
#23 0x00007f3cae4f70a3 in QgsNetworkAccessManager::createRequest (this=0x299c1c0, op=QNetworkAccessManager::GetOperation,
req=...,
outgoingData=0x0) at /home/radim/devel/qgis/src/core/qgsnetworkaccessmanager.cpp:166
#24 0x00007f3cab7e7224 in QNetworkAccessManager::get (this=0x299c1c0, request=...) at access/qnetworkaccessmanager.cpp:598
#25 0x00007f3c2e3f9aaa in QgsWmsCapabilitiesDownload::downloadCapabilities (this=0x50e9af0)
    at /home/radim/devel/qgis/src/providers/wms/qgswmscapabilities.cpp:1906
#26 0x00007f3c2e3f96f6 in QgsWmsCapabilitiesDownload::downloadCapabilities (this=0x50e9af0, baseUrl=..., auth=...)
    at /home/radim/devel/qgis/src/providers/wms/qgswmscapabilities.cpp:1877
#27 0x00007f3c2e45d8aa in QgsWMSConnectionItem::createChildren (this=0x50e98a0)
    at /home/radim/devel/qgis/src/providers/wms/qgswmsdataitems.cpp:69
```

```
#28 0x00007f3cae4211c5 in QgsDataItem::runCreateChildren (item=0x50e98a0) at /home/radim/devel/qgis/src/core/qgsdataitem.cpp:383
#29 0x00007f3cae42d3c7 in QtConcurrent::StoredFunctionCall1<QVector<QgsDataItem*>, QVector<QgsDataItem*> (*) (QgsDataItem*),
QgsDataItem*>::runFunctor (this=0x5109db0) at /usr/include/qt4/QtCore/qtconcurrentstoredfunctioncall.h:267
#30 0x00007f3cae42d2b6 in QtConcurrent::RunFunctionTask<QVector<QgsDataItem*> >::run (this=0x5109db0)
    at /usr/include/qt4/QtCore/qtconcurrentrunbase.h:106
#31 0x00007f3cac838b80 in QThreadPoolThread::run (this=0x5109fb0) at concurrent/qthreadpool.cpp:108
#32 0x00007f3cac84570f in QThreadPrivate::start (arg=0x5109fb0) at thread/qthread_unix.cpp:349
#33 0x00007f3ca922c0a4 in start_thread (arg=0x7f3c1eff3700) at pthread_create.c:309
#34 0x00007f3ca952704d in clone () at ../sysdeps/unix/sysv/linux/x86_64/clone.S:111
```

The problem is, that all functions in this QNetworkAccessManager are reentrant, which means that the functions can [simultaneously] be called safely from multiple threads, as long as each thread uses a different instance of the class. Unfortunately we are currently in QGIS using everywhere QGIS QgsNetworkAccessManager::instance(). So either we have to use one instance per thread or add mutex to QgsNetworkAccessManager::get().

#4 - 2015-09-01 07:44 AM - Radim Blazek

In commit:e95bf6d is implemented solution which I believe is the right one. QgsNetworkAccessManager::instance() is used only through signals/slots. That should guarantee that its methods won't be called from different threads.

Maybe overcomplicated? Could it be done in a simpler way?

It is now implemented in WMS provider, it should be moved to a generic QgsDownloader and used also in WCS and WFS, maybe also where new QgsNetworkAccessManager() is used instead of QgsNetworkAccessManager::instance().

Please test.

#5 - 2015-10-05 03:39 AM - Jürgen Fischer

- Status changed from Open to Feedback

#6 - 2015-10-05 06:03 PM - Mathieu Pellerin - nIRV

- Assignee set to Radim Blazek
- Crashes QGIS or corrupts data changed from No to Yes
- Resolution set to fixed/implemented
- % Done changed from 0 to 100
- Status changed from Feedback to Closed

Confirmed as fixed, thanks again Radim.

#7 - 2015-10-05 11:01 PM - Radim Blazek

- Subject changed from QGIS fails to launch or hang while exiting due to thread issue with WMS servers to QGIS fails to launch or hang while exiting due to thread issue with OWS servers
- Status changed from Closed to Reopened

I have reopened and changed the subject to OWS because it has to be fixed also for WCS and WFS.

#8 - 2015-11-07 11:26 AM - Giovanni Manghi

- *Target version deleted (Version 2.12)*

#9 - 2016-03-05 04:29 AM - Jürgen Fischer

- *Status changed from Reopened to Closed*

Fixed in changeset commit:"2eb82430bbdb02b0789b3ffe80d5b6fd747fa8cc".