DCMTK - Bug #683

Data race conditions in ofstd/tests/tthread.cc

2016-04-29 16:48 - Marco Eichelberg

Status:	WontFix	**Start date:**	2016-04-29
Priority:	Low	**Due date:**	
Assignee:	Marco Eichelberg	**% Done:**	0%
Category:	Testing	**Estimated time:**	0:00 hour
Target version:			
Module:	ofstd	**Compiler:**	
Operating System:			

Description

When checking the DCMTK unit tests with Pareon Verify, 25 data races are reported in

```
[M0108] Data race(s) detected:
  the static object of size 4 allocated as `rw_cond1' at dcmtk/ofstd/tests/tthread.cc:223
  is concurrently accessed by
  the write in
    function RWLockerT1::run() at dcmtk/ofstd/tests/tthread.cc:347
    called from function thread_stub at dcmtk/ofstd/libsrc/ofthread.cc:94
    ^^^ thread start ^^^
    called from function pthread_create
    called from function OFThread::start() at dcmtk/ofstd/libsrc/ofthread.cc:142
    called from function rwlocker_test() at dcmtk/ofstd/tests/tthread.cc:423
    called from function OFTestofstd_thread::run() at dcmtk/ofstd/tests/tthread.cc:583
    called from function OFTestTest::runAndReturn() at dcmtk/ofstd/include/dcmtk/ofstd/oftest.h:10
3
    called from function OFTestManager::runTests(OFList<OFTestTest*> const&, char const*) at dcmtk
/ofstd/include/dcmtk/ofstd/oftest.h:184
    called from function OFTestManager::run(int, char**, char const*) at dcmtk/ofstd/include/dcmtk
/ofstd/oftest.h:291
    called from function main at dcmtk/ofstd/tests/tests.cc:81
    ^^^ application start ^^^
  performing 1 access of size 4 at the start of the object
  and the read in
    function rwlocker_test() at dcmtk/ofstd/tests/tthread.cc:430
    called from function OFTestofstd_thread::run() at dcmtk/ofstd/tests/tthread.cc:583
    called from function OFTestTest::runAndReturn() at dcmtk/ofstd/include/dcmtk/ofstd/oftest.h:10
3
    called from function OFTestManager::runTests(OFList<OFTestTest*> const&, char const*) at dcmtk
/ofstd/include/dcmtk/ofstd/oftest.h:184
    called from function OFTestManager::run(int, char**, char const*) at dcmtk/ofstd/include/dcmtk
/ofstd/oftest.h:291
    called from function main at dcmtk/ofstd/tests/tests.cc:81
    ^^^ application start ^^^
  performing 1 access of size 4 at the start of the object
```

Similar races are reported for the following variables:

```
  the static object of size 4 allocated as `mtx_cond1' at dcmtk/ofstd/tests/tthread.cc:39
  the static object of size 4 allocated as `mtx_cond2' at dcmtk/ofstd/tests/tthread.cc:40
  the static object of size 4 allocated as `mtx_cond3' at dcmtk/ofstd/tests/tthread.cc:41
  the static object of size 4 allocated as `rw_cond1' at dcmtk/ofstd/tests/tthread.cc:223
  the static object of size 4 allocated as `rw_cond2' at dcmtk/ofstd/tests/tthread.cc:224
  the static object of size 4 allocated as `rw_cond3' at dcmtk/ofstd/tests/tthread.cc:225
  the static object of size 4 allocated as `rw_cond4' at dcmtk/ofstd/tests/tthread.cc:226
  the static object of size 4 allocated as `rw_cond5' at dcmtk/ofstd/tests/tthread.cc:227
  the static object of size 4 allocated as `rw_cond6' at dcmtk/ofstd/tests/tthread.cc:228
  the static object of size 4 allocated as `rw_cond7' at dcmtk/ofstd/tests/tthread.cc:229
```

```
  the static object of size 4 allocated as `sem_cond1' at dcmtk/ofstd/tests/tthread.cc:130
  the static object of size 4 allocated as `sem_cond2' at dcmtk/ofstd/tests/tthread.cc:131
  the static object of size 4 allocated as `sem_cond3' at dcmtk/ofstd/tests/tthread.cc:132
  the static object of size 4 allocated as `sem_cond4' at dcmtk/ofstd/tests/tthread.cc:133
  the static object of size 4 allocated as `tsd_cond1' at dcmtk/ofstd/tests/tthread.cc:459
  the static object of size 4 allocated as `tsd_cond2' at dcmtk/ofstd/tests/tthread.cc:460
  the static object of size 4 allocated as `tsd_cond3' at dcmtk/ofstd/tests/tthread.cc:461
  the static object of size 4 allocated as `tsd_cond4' at dcmtk/ofstd/tests/tthread.cc:462
```

The reports are correct, but due to the fact that this code tries to verify whether or not the mutex, semaphore and read/write lock abstractions in DCMTK work correctly. It would be impossible to fix the test program without duplicating the complete code that is actually tested in that unit test. Since only the unit test is affected, we have decided to not fix this for now. We might revisit the issue later when all compilers can be assumed to support C++11, where std::atomic_bool could be used in the test case.