DCMTK - Bug #975

DCMTK static binaries on Linux are not really static

2021-03-25 10:28 - Marco Eichelberg

Status:	Closed	**Start date:**	2021-03-25
Priority:	Normal	**Due date:**	
Assignee:	Nikolas Goldhammer	**% Done:**	100%
Category:	Library and Apps	**Estimated time:**	0:00 hour
Target version:	3.6.8		
Module:		**Compiler:**	
Operating System:			

Description

The static Linux binaries created by the packaging system are not really static, as glibc will load some shared objects. When running these binaries on a different distributions, errors like the following may occur:

```
img2dcm: dl-call-libc-early-init.c:37: _dl_call_libc_early_init:
Assertion `sym != NULL' failed.
```

Apparently this problem can be solved by linking against special static versions of glibc and libstdc++,
which on some distributions are available as packages glibc-static and libstdc++-static
See https://www.systutorials.com/how-to-statically-link-c-and-c-programs-on-linux-with-gcc/.

The static binaries should be built as truly static binaries that do not load any shared object and thus run on any Linux distribution.

History

#1 - 2021-05-14 12:13 - Marco Eichelberg

According to the article referenced in the original issue description, it is not possible (or at least not desirable due to side-effects) to compile completely static binaries on Linux when using glibc. The problem is that glibc needs to load certain shared objects for example for the DNS related gethostinfo() function which resolves hostnames to IP addresses. In a static binary, this will fail if the version numbers of the shared libraries loaded glibc do not match exactly, i.e. if the host system uses a different glibc version. That means that fully static binaries actually **decrease** portability of the binary when glibc is in use.

There are two possible solutions:

1. Either compile fully static binaries using a different libc implementation such as the MUSL libc used in Alpine Linux. Whether or not these binaries work without problems on Linux distributions internally using glibc (such as Debian, Ubuntu, Redhat, SuSE) needs to be determined
2. Alternatively, we could compile dynamic libraries where only glibc is loaded dynamically and all other libraries are linked statically. This will provide maximum compatibility for glibc based distributions as this only requires a glibc that is "recent enough". Details on that are discussed here: https://insanecoding.blogspot.com/2012/07/creating-portable-linux-binaries.html

For the second solution, we would have to provide a "portable binaries" mode in the CMake build chain that only detects static versions of the support libraries (i.e. only lib<name>.a, not lib<name>.so in the find_package() calls and that also sets the LDFLAGS "-static-libstdc++ -static-libgcc", which will cause the static versions of libgcc and libstdc++ (but not of glibc) to be used. Note that this applies to gcc and clang, not necessarily to other compilers. The resulting binary should only have few dependencies from shared objects, possibly the following ones:

- linux-vdso.so.1
- libc.so.6
- libdl.so.2
- libnsl.so.1
- libpthread.so.0
- libm.so.6
- /lib64/ld-linux-x86-64.so.2

#2 - 2021-11-11 21:29 - Michael Onken

- Target version set to 3.6.7

#3 - 2021-11-12 11:25 - Michael Onken

Maybe also using something like Flatpak, Snap or AppImage could work, if not every binary has to be put into its own "container".

#4 - 2021-11-12 11:25 - Michael Onken

- Assignee set to Nikolas Goldhammer

#5 - 2022-05-03 21:07 - Jörg Riesmeier

- Target version changed from 3.6.7 to 3.6.8

#6 - 2022-06-15 08:48 - Marco Eichelberg

Meanwhile a new build mode DCMTK_PORTABLE_LINUX_BINARIES has been introduced. This will cause binaries to be created where all third-party libraries are linked statically, whereas glibc is linked dynamically. These binaries should be fairly portable and are about the best we can create while using glibc. Truly static binaries can be generated on a Linux distribution using the MUSL libc, such as Alpine Linux, using the DCMTK_LINK_STATIC build mode.

#7 - 2022-10-10 10:31 - Marco Eichelberg

- Status changed from New to Closed

- % Done changed from 0 to 100