


Open

Opened 1 week ago by  Patrick Storz

g_stat - possbile memory corruption causing SEGFALT

Investigating [a regression / crashing issue](#) in Inkscape when built with MSYS2's mingw-w64 I came up with the following minimal testcase which reproduces the segmentation fault:

```
#include <iostream>
#include <vector>

#include <glib/gstdio.h>

int main()
{
    std::vector<std::string> filesFound;

    GStatBuf st;
    g_stat("C:\\", &st);
}
```

(I compiled it with `g++ test.cpp pkg-config --cflags --libs glib-2.0 -O1 -o test.exe`)

What I found so far:

- The segfault seems to occur when deleting "filesFound" and unless there's a bug in gcc (which I can't rule out at this point) this might indicate that there's some sort of memory corruption while calling `g_stat`.
- The segfault occurs with glib 2.56.0 and above but not with glib 2.54.3 (and probably earlier versions).
- The segfault occurs with gcc 8.2.0 and gcc 7.3.0 (the only other recent update besides glib I can think of).
- The segfault occurs in 64-bit builds bot *not* in 32-bit builds.
- The segfault occurs when compiled with `-O1` and below but not with `-O2` and above.

One change in glib that might be relevant is [53bd6a35](#) by [@ruslanizhb](#). Maybe there's some discrepancy in sizes of the stat struct which is now exposed due to using 64-bit types in 64-bit builds (which was not done before AFAIK)?


2 Related Merge Requests

-  I226 [gstdio: use _stat64 for GStatBuf on 64bit mingw. Fixes #1476](#)

Open
-  I228 [meson: define _FILE_OFFSET_BITS=64 for MinGW. See #1476](#)

Open


When these merge requests are accepted, this issue will be closed automatically.

 Christoph Reiter [@creiter](#) · 1 week ago

Developer

I suspect this is https://bugzilla.gnome.org/show_bug.cgi?id=728663


From a quick test with 2.56, sizeof(GStatBuf) is different at glib compile time vs program compile time, which would explain the stack corruption.

 LRN [@ruslanizhb](#) · 1 week ago

Developer

`sizeof(GStatBuf)` shouldn't change. I deliberately used `struct _stat` for it, which ensures that it always uses 32-bit fields. If it does change, then something is wrong.


I've been meaning to debug this, but performance investigations ate a lot of time.

 LRN [@ruslanizhb](#) · 1 week ago

Developer

Okay, the `gstdio.h` header has an exception for `_WIN64`, which means that on x86_64 it does use normal `struct stat`. But when compiling for `x86_64` MinGW-w64 continues to define `off_t` to be 32-bit, and `st_size` has type `off_t`, meaning that `st_size` continues to be 32-bit.

I've asked on MinGW-w64 ML about this. Meanwhile a quick solution on your part is to compile with `-D_FILE_OFFSET_BITS=64` - that is, use [LFS](#).


 Simon McVittie [@smcv](#) added

1. Crash

gstdio

win32

 labels · 1 week ago


 Christoph Reiter [@creiter](#) · 1 week ago

Developer

I suspect this is https://bugzilla.gnome.org/show_bug.cgi?id=728663

Oh, I assumed that patch wasn't in 2.56, but it is, so ignore that. The reason I couldn't reproduce on master is likely meson vs autotools then.


@LRN could it be that the autotools build passes `_FILE_OFFSET_BITS=64` by default? Problem then is if we switch msys2 to meson things start to no longer match again..

 Christoph Reiter [@creiter](#) · 1 week ago

Developer


[@ruslanizhb](#) what if we change the "struct stat" to "struct _stat64" for mingw+64bit? From what I see that would match the old default autotools size, makes meson+autotools match, and fixes the reported crash here if backported.


Edited by Christoph Reiter 1 week ago


 LRN [@ruslanizhb](#) · 1 week ago

Developer

That could work.


 Christoph Reiter [@creiter](#) mentioned in commit [631893f7](#) · 1 week ago

 Christoph Reiter [@creiter](#) mentioned in merge request [I226](#) · 1 week ago

 Christoph Reiter [@creiter](#) · 1 week ago

Developer


I've opened [I226](#) (might be good for msvc as well, but it would be an ABI break there I guess?)

 Emmanuele Bassi [@ebassi](#) · 1 week ago

Maintainer

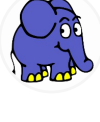
could it be that the autotools build passes `_FILE_OFFSET_BITS=64` by default?

Meson does this by default on any Linux-like C compiler, i.e. GCC, Clang, and ICC, so I'm not entirely sure this applies.

 LRN [@ruslanizhb](#) · 1 week ago


Developer

Most advanced buildsystems do. But *users* of glib could be using anything - plain makefiles or even self-written scripts. We can't expect them to define `_FILE_OFFSET_BITS=64`. And without that their version of `GStatBuf` will not match our version. So - yes, `struct _stat64`, which has 64-bit time and size fields, regardless of LFS being enabled or disabled, is the right thing to use, as long as there aren't any [or many] glib users out there that assumed something else.

 Patrick Storz [@FdeI23](#) · 1 week ago

From the user's point of view the whole idea of `GStatBuf` is to guarantee that it will always have the correct type (regardless of the environment and of what the build system might or might not do).

So whatever that type might be there mustn't be any possibility for the build system to influence it.

 Christoph Reiter [@creiter](#) · 1 week ago

Developer

could it be that the autotools build passes `_FILE_OFFSET_BITS=64` by default? Meson does this by default on any Linux-like C compiler, i.e. GCC, Clang, and ICC, so I'm not entirely sure this applie

It doesn't with mingw (I just tested it). I found <https://github.com/mesonbuild/meson/issues/3049> which is somewhat related.

Ideally it shouldn't matter, as msvc ignores it and we should use the right windows API instead, but I'd enable it anyway in case we somehow depend on it.

 Christoph Reiter [@creiter](#) mentioned in commit [7e6fb333](#) · 1 week ago

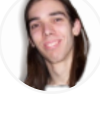
 Christoph Reiter [@creiter](#) mentioned in merge request [I228](#) · 1 week ago

 Simon McVittie [@smcv](#) · 1 week ago

Developer

could it be that the autotools build passes `_FILE_OFFSET_BITS=64` by default?

The Autotools build uses `AC_SYS_LARGEFILE`, which defines `_FILE_OFFSET_BITS=64` (or some obsolete equivalent on rarer platforms) by default, but can be told not to do so with `./configure --disable-largefile`. So this has always been conditional.

 Simon McVittie [@smcv](#) · 1 week ago

Developer

From the user's point of view the whole idea of `GStatBuf` is to guarantee that it will always have the correct type (regardless of the environment and of what the build system might or might not do).

Yes. Specifically, it's whatever type is correct for `g_stat()`.

When a third party uses GLib, there are two relevant build systems: the build system for GLib itself, and the build system for the user code (for example Inkscape or the minimal reproducer at the top of this bug report).

A complication here is that `g_stat` is inlined into `gstdio.h` (as a call to `stat`) on Unix if `G_STDIO_NO_WRAP_ON_UNIX` is undefined, but not on Windows or if `G_STDIO_NO_WRAP_ON_UNIX` is defined.

In the normal Unix case where `g_stat` is just a `#define` for `stat`, the buffer type for both `g_stat` and `GStatBuf` must come from *user code's* build system: it could be the large-file-support version, or the legacy version, depending whether user code has used `AC_SYS_LARGEFILE` or the various non-Autotools equivalents (which it should - all code that might call `stat()` should enable large file support, even if it will never actually open large files, so that it can cope with large inode numbers).

In the Windows case and the weird Unix case where `g_stat` is a real function in `gstdio.c`, the definition of `GStatBuf` must match whatever type was selected by *GLib's* build system, ignoring whatever large-file support is selected or not selected by the build system of user code. That would have to be done by hard-coding it in `glibconfig.h` during GLib's `./configure` or Meson/MSVC equivalent.

One solution to this whole mess is to use the GIO APIs, which use a properly opaque data structure, instead.

Please [register](#) or [sign in](#) to reply