


# null pointer error in get\_basename() which leads to crash when paste a zip format file in ubuntu 22.04

## Affected version

- Nightly flatpak: Can't test it because i can't download Nightly version.
- Other:1:42.2  [SwitchyOmega.zip](#)

## Steps to reproduce

1. install ubuntu 22.04 in vmware pro 16.2.4 build-20089737.
2. install flatpak environment, download nautilus 1:42.2.
3. use Builder build it, and run it with Valgrind, the nautilus window will open.
4. outside vmware, copy the attachment file.
5. in vmware, in the opened nautilus window, paste it.
6. nautilus crashes then.

## Current behavior

it just crashed.

## Expected behavior

the zip file is pasted successfully.

## Additional information

1. the Valgrind information:

- ==2== Memcheck, a memory error detector
- ==2== Copyright (C) 2002-2022, and GNU GPL'd, by Julian Seward et al.
- ==2== Using Valgrind-3.19.0 and LibVEX; rerun with -h for copyright info
- ==2== Command: nautilus
- ==2==
- ==2== Thread 10 pool-org.gnome.:
- ==2== Invalid read of size 1
- ==2== at 0x48F9670: g\_utf8\_validate (in /usr/lib/x86\_64-linux-gnu/libglib-2.0.so.0.7302.0)
- ==2== by 0x16EAE8: get\_basename (nautilus-file-operations.c:1024)
- ==2== by 0x173EE2: scan\_file (nautilus-file-operations.c:3634)
- ==2== by 0x1741BD: scan\_sources (nautilus-file-operations.c:3715)
- ==2== by 0x178AA4: nautilus\_file\_operations\_copy (nautilus-file-operations.c:6064)
- ==2== by 0x554E095: ??? (in /usr/lib/x86\_64-linux-gnu/libgio-2.0.so.0.7302.0)
- ==2== by 0x48EE461: ??? (in /usr/lib/x86\_64-linux-gnu/libglib-2.0.so.0.7302.0)
- ==2== by 0x48ED9C8: ??? (in /usr/lib/x86\_64-linux-gnu/libglib-2.0.so.0.7302.0)
- ==2== by 0x5A711D9: ??? (in /usr/lib/x86\_64-linux-gnu/libc.so.6)
- ==2== by 0x5AF9D83: clone (in /usr/lib/x86\_64-linux-gnu/libc.so.6)
- ==2== Address 0x0 is not stack'd, malloc'd or (recently) free'd
- ==2==
- ==2==
- ==2== Process terminating with default action of signal 11 (SIGSEGV)
- ==2== Access not within mapped region at address 0x0
- ==2== at 0x48F9670: g\_utf8\_validate (in /usr/lib/x86\_64-linux-gnu/libglib-2.0.so.0.7302.0)
- ==2== by 0x16EAE8: get\_basename (nautilus-file-operations.c:1024)
- ==2== by 0x173EE2: scan\_file (nautilus-file-operations.c:3634)
- ==2== by 0x1741BD: scan\_sources (nautilus-file-operations.c:3715)
- ==2== by 0x178AA4: nautilus\_file\_operations\_copy (nautilus-file-operations.c:6064)
- ==2== by 0x554E095: ??? (in /usr/lib/x86\_64-linux-gnu/libgio-2.0.so.0.7302.0)
- ==2== by 0x48EE461: ??? (in /usr/lib/x86\_64-linux-gnu/libglib-2.0.so.0.7302.0)
- ==2== by 0x48ED9C8: ??? (in /usr/lib/x86\_64-linux-gnu/libglib-2.0.so.0.7302.0)
- ==2== by 0x5A711D9: ??? (in /usr/lib/x86\_64-linux-gnu/libc.so.6)

- ==2== by 0x5AF9D83: clone (in /usr/lib/x86\_64-linux-gnu/libc.so.6)
- ==2== If you believe this happened as a result of a stack
- ==2== overflow in your program's main thread (unlikely but
- ==2== possible), you can try to increase the size of the
- ==2== main thread stack using the --main-stacksize= flag.
- ==2== The main thread stack size used in this run was 8388608.
- ==2==
- ==2== HEAP SUMMARY:
- ==2== in use at exit: 11,518,984 bytes in 109,964 blocks
- ==2== total heap usage: 968,922 allocs, 858,958 frees, 127,795,237 bytes allocated
- ==2==
- ==2== LEAK SUMMARY:
- ==2== definitely lost: 21,413 bytes in 16 blocks
- ==2== indirectly lost: 269,366 bytes in 1,220 blocks
- ==2== possibly lost: 103,840 bytes in 2,298 blocks
- ==2== still reachable: 9,166,389 bytes in 93,765 blocks
- ==2== of which reachable via heuristic:
- ==2== newarray : 4,264 bytes in 1 blocks
- ==2== suppressed: 0 bytes in 0 blocks
- ==2== Rerun with --leak-check=full to see details of leaked memory
- ==2==
- ==2== For lists of detected and suppressed errors, rerun with: -s
- ==2== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)

#### 1. backtrace information

- the crash happened in `g_utf8_validate()` function, despite this function is provide by `libglib2.0`, but I think the crash is caused by the function `get_basename()`, because it passed a null pointer "basename" to `g_utf8_validate()`.

Frame	Function	Arguments	Location	Binary
0	<code>g_utf8_validate</code>			<code>/usr/lib/x86_64-linux-gnu/libglib-2.0.so.0</code>
1	<code>get_basename</code>		<code>... sh/nautilus-42.2/src/nautilus-file-operations.c:1024</code>	
2	<code>scan_file</code>		<code>... sh/nautilus-42.2/src/nautilus-file-operations.c:3634</code>	
3	<code>scan_sources</code>		<code>... sh/nautilus-42.2/src/nautilus-file-operations.c:3715</code>	
4	<code>nautilus_file_operations_copy</code>		<code>... sh/nautilus-42.2/src/nautilus-file-operations.c:6064</code>	
5	<code>g_task_thread_pool_thread</code>			<code>/usr/lib/x86_64-linux-gnu/libgio-2.0.so.0</code>
6	<code>g_thread_pool_thread_proxy</code>			<code>/usr/lib/x86_64-linux-gnu/libglib-2.0.so.0</code>
7	<code>g_thread_proxy</code>			<code>/usr/lib/x86_64-linux-gnu/libglib-2.0.so.0</code>
8	<code>start_thread</code>			<code>/usr/lib/x86_64-linux-gnu/libc.so.6</code>
9	<code>clone</code>			<code>/usr/lib/x86_64-linux-gnu/libc.so.6</code>

#### 1. where the crash happened

```

1003     name = g_mount_get_name (mount);
1004     g_object_unref (mount);
1005 }
1006 else
1007 {
1008     info = g_file_query_info (file,
1009                             G_FILE_ATTRIBUTE_STANDARD_DISPLAY_NAME,
1010                             0,
1011                             g_cancellable_get_current (),
1012                             NULL);
1013     name = NULL;
1014     if (info)
1015     {
1016         name = g_strdup (g_file_info_get_display_name (info));
1017         g_object_unref (info);
1018     }
1019 }
1020
1021 if (name == NULL)
1022 {
1023     basename = g_file_get_basename (file);
1024     if (g_utf8_validate (basename, -1, NULL))
1025     {
1026         name = basename;
1027     }
1028     else
1029     {
1030         name = g_uri_escape_string (basename, G_URI_RESERVED_CHARS_ALLOWED_IN_PATH, TRUE);
1031         g_free (basename);
1032     }
1033 }
1034
1035 /* Some chars can't be put in the markup we use for the dialogs... */
1036 if (has_invalid_xml_char (name))
1037 {

```

#### 1. "basename" leads to the crash

Variable	Type	Value
Parameters		
file	GFile *	0x555555ff9960
Locals		
info	GFileInfo *	0x0
name	gchar *	0x0
basename	gchar *	0x0
tmp	gchar *	0x2 <error: Cann...
mount	GMount *	0x0
Parameters		
file	GFile *	0x555555ff9960
Locals		
info	GFileInfo *	0x0
name	gchar *	0x0
basename	gchar *	0x0
tmp	gchar *	0x2 <error: Cann...
mount	GMount *	0x0

To upload designs, you'll need to enable LFS and have an admin enable hashed storage. [More information](#)

Tasks 0

No tasks are currently assigned. Use tasks to break down this issue into smaller parts.

Linked items 0

Related merge requests 2

☐ [Fix crash when copying an invalid file](#)

!1001



☐ [Incorrect letter shown for file type name in Japanese and Cyrillic languages](#)

!1002



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

## Activity



[wu chunming](#) @wuchunming · 3 months ago

Author

By the way, the version of glib2.0 which I installed is 2.72.1.



[Peter Eisenmann](#) added [1. Crash](#) [2. Needs Diagnosis](#) labels 3 months ago



[wu chunming](#) @wuchunming · 3 months ago

Author

- I write a poc.py which use D-BUS to prove it.
- you can just reproduce it simply by the following steps:

- open a nautilus file window
- execute python3 poc.py
- the windows crashes.

- the poc.py is there.
- [poc.py](#)
- and I record a video.

0:00 / 0:25

- [poc](#)



**Mark Esler** @eslerm · 1 month ago

To solve this security issue with an available PoC, does `basename` need to be checked if `NULL` before being used in `g_utf8_validate` ?

Something like

```
if (name == NULL)
{
-   basename = g_file_get_basename (file);
-   if (g_utf8_validate (basename, -1, NULL))
+   if (basename = g_file_get_basename (file);)
    {
-       name = basename;
+       if (g_utf8_validate (basename, -1, NULL))
+       {
+           name = basename;
+       }
+       else
+       {
+           name = g_uri_escape_string (basename, G_URI_RESERVED_CHARS_ALLOWED_IN_PATH, TRUE);
+           g_free (basename);
+       }
    }
    else
    {
-       name = g_uri_escape_string (basename, G_URI_RESERVED_CHARS_ALLOWED_IN_PATH, TRUE);
-       g_free (basename);
+       /* handle NULL name and basename */
    }
}
```

Edited by [Mark Esler](#) 1 month ago



**wu chunming** @wuchunming · 1 month ago

Author

Yes, I think this issue can be solved by this way.

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



**Mark Esler** @eslerm · 1 month ago

This null pointer dereference pattern occurs three times in `nautilus-file-operations.c`.

It was introduced during Nautilus 2.20 with commit [469047a2](#)



[Aleksandar Dezelin](#) mentioned in merge request [!1001](#) 1 month ago



[Aleksandar Dezelin](#) mentioned in merge request [!1002](#) 1 month ago



[António Fernandes](#) removed [2\\_Needs Diagnosis](#) label 1 month ago



[António Fernandes](#) added [5\\_Operations](#) label 1 month ago



**Steve Beattie** @smb · 1 week ago

For reference, this issue was assigned [CVE-2022-37290](#).

(I'm just a messenger, not the person who assigned the CVE.)



**Corey Berla** mentioned in commit [dezelin/nautilus@420acec9](#) 10 hours ago

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