

---

Subject: Suppressions file for valgrind

Posted by [Novo](#) on Fri, 30 May 2008 04:04:33 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

ubuntu.710.upp.valgrind.supp - file with suppressions prepared on Ubuntu 7.10 + X11 + GTK.  
Suppressions are checked on TheIDE.

I've added several suppressions for GTK.

### File Attachments

1) [ubuntu.710.upp.valgrind.supp](#), downloaded 443 times

---

---

Subject: Re: Suppressions file for valgrind

Posted by [Novo](#) on Sun, 01 Jun 2008 17:25:17 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Updated file with suppressions.

I've added several suppressions for GTK.

---

---

Subject: Re: Suppressions file for valgrind

Posted by [mdelfede](#) on Mon, 02 Jun 2008 09:45:11 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Thanx! Updated in SVN 277 build.

Ciao

Max

---

---

Subject: Re: Suppressions file for valgrind

Posted by [Novo](#) on Tue, 03 Jun 2008 03:44:12 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

mdelfede wrote on Mon, 02 June 2008 05:45Thanx! Updated in SVN 277 build.

Ciao

Max

IMHO, there is no such thing as one universal suppression file. Each combination of different

versions of X11 and GTK DLLs may have their own list of suppressions.

For some reason all issues in the current suppression file are related to fonts. That make me think that those are actually problems with U++ itself.

---

---

Subject: Re: Suppressions file for valgrind  
Posted by [mdelfede](#) on Wed, 04 Jun 2008 10:15:25 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

Novo wrote on Tue, 03 June 2008 05:44mdelfede wrote on Mon, 02 June 2008 05:45Thanx!  
Updated in SVN 277 build.

Ciao

Max

IMHO, there is no such thing as one universal suppression file. Each combination of different versions of X11 and GTK DLLs may have their own list of suppressions.

Well... you're right, I had to add some suppressions for different libraries on ubuntu 8.04. But, well, suppressing non-existent libraries does no harm

Quote:

For some reason all issues in the current suppression file are related to fonts. That make me think that those are actually problems with U++ itself.

Well, I noticed it too, but there are issues even on some functions that just have XDisplay as parameter.... And I guess that those can't be related to UPP, as XDisplay is needed (and would not work with uninitialized one). So, it must be in X11, IMHO.

Max

---

---

Subject: Re: Suppressions file for valgrind  
Posted by [Novo](#) on Wed, 04 Jun 2008 16:59:11 GMT  
[View Forum Message](#) <> [Reply to Message](#)

---

mdelfede wrote on Wed, 04 June 2008 06:15

Well... you're right, I had to add some suppressions for different libraries on ubuntu 8.04. But, well, suppressing non-existent libraries does no harm

number of suppressions and length of stack trace in each suppression.

BTW, I've found an interesting message in valgrind mailing list:

One of the kinds of errors that Memcheck finds is dangerous uses of uninitialised values, usually when they get used in an if-statement. This is a useful facility, but it is also the single most complained about aspect of Memcheck. The problem is that the point where Memcheck reports the error is often a long way after the heap or stack allocation that created the uninitialised value. And so it can be very difficult to find the root cause of the problem.

Memcheck will now optionally track these origins, and, when reporting an uninitialised value error, it can show the origin too. If the origin is a heap allocation, it shows where the block was allocated. If the origin was a stack allocation, it will tell you the function that did the allocation. A couple of simple examples are shown below.

Of course there's no free lunch: Memcheck's speed is approximately halved, and memory consumption increases by a minimum of 100MB. But in initial testing, on a large C++ codebase, it has proven effective.

You can try out this functionality using the SVN trunk:

```
svn co svn://svn.valgrind.org/valgrind/trunk
cd trunk
./autogen.sh
./configure --prefix=...
```

and then run with `--track-origins=yes`.

This functionality was inspired by the work of Bond, Nethercote, et al, as reported in the paper "Tracking Bad Apples: Reporting the Origin of Null and Undefined Value Errors" (<http://www.valgrind.org/docs/origin-tracking2007.pdf>), but the actual implementation is very different from that described in the paper.

Feedback, comments, bug reports welcome.

J

Simple example of an uninitialised value use originating from

a heap block:

Conditional jump or move depends on uninitialised value(s)

at 0x400ACB: main (origin1-yes.c:64)

Uninitialised value was created by a heap allocation

at 0x4C234BB: malloc (vg\_replace\_malloc.c:207)

by 0x400A9B: main (origin1-yes.c:61)

And one from a stack allocation (a local, or "auto" variable):

Conditional jump or move depends on uninitialised value(s)

at 0x400944: main (origin3-no.c:33)

Uninitialised value was created by a stack allocation

at 0x4008B4: main (origin3-no.c:15)

An example from the opposite end of the size/triviality spectrum:

Use of uninitialised value of size 8

at 0x4F277E7: BitmapReadAccess::SetPixelFor\_24BIT\_TC\_BGR  
(bmpacc2.cxx:195)

by 0x4F1A8CE: Bitmap::ImplConvertUp (bmpacc.hxx:542)

by 0x4F1B9D3: Bitmap::Convert (bitmap3.cxx:365)

Uninitialised value was created by a heap allocation

at 0x4C22DDB: malloc (vg\_replace\_malloc.c:207)

by 0x719B4FA: rtl\_allocateMemory (alloc\_global.c:311)

by 0x470BDA: allocate (operators\_new\_delete.cxx:160)

by 0x470C4A: operator new[](unsigned long)

(operators\_new\_delete.cxx:239)

by 0xEBCAABC: X11SalBitmap::ImplCreateDIB (salbmp.cxx:194)

---

Subject: Re: Suppressions file for valgrind

Posted by [mdelfede](#) on Thu, 05 Jun 2008 13:17:31 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Wow, tracing origin of un-initalized stuffs is a really big improvement ! Is it only on current svn tree ?

Max

---

Subject: Re: Suppressions file for valgrind

Posted by [Novo](#) on Thu, 05 Jun 2008 15:07:29 GMT

[View Forum Message](#) <> [Reply to Message](#)

mdelfede wrote on Thu, 05 June 2008 09:17Wow, tracing origin of un-initalized stuffs is a really big improvement ! Is it only on current svn tree ?

Max

That was announced on Mach 04. Version 3.3.1 of Valgrind was released yesterday ...

Unfortunately, uninitialized value origin tracking is not mentioned in release notes. Probably it is not that important comparing to fixed bugs