Subject: Using Valgrind output
Posted by slashupp on Tue, 11 Jun 2019 08:59:39 GMT
View Forum Message <> Reply to Message

WHen I do [Debug:Test in Valgrind] I get output like this:

...
Invalid read of size 4
  0x31B5C9 /home/myhome/upp.out/myapps/GCC.Debug.Debug_Full.Gui.Shared/testmyapp
...

Question 1:
I assume the hex-value 0x31B5C9 is some kind of offset - how do I use this to find
the functions/code-locations for this bug/future crash of my app?

Also valgrind outputs a lot of system-file related issues which is out of my control.

Question 2:
How do I get rid of the system-file output and increase/improve the info about issues in my app
so I can locate & fix them?

thx

Subject: Re: Using Valgrind output
Posted by Novo on Tue, 11 Jun 2019 13:26:48 GMT
View Forum Message <> Reply to Message

1.
If your app is built with debug info you should get output like below:
==28420== Use of uninitialised value of size 8
==28420==   at 0x97E165: Upp::BltAAMapRGBA3(unsigned int*, Upp::RGBA const*, unsigned int const*) (ImageScale.cpp:263)
==28420==   by 0x95783E: Upp::RescaleImage::Get(Upp::RGBA*) (ImageScale.cpp:436)
==28420==   by 0x97E712: Upp::Rescale(Upp::RasterEncoder&, Upp::Size_<int>, Upp::Raster&, Upp::Rect_<int> const&, Upp::Function<bool (int, int)>) (ImageScale.cpp:485)
==28420==   by 0x958237: Upp::Rescale(Upp::Image const&, Upp::Size_<int>, Upp::Rect_<int> const&, Upp::Function<bool (int, int)>) (ImageScale.cpp:497)
==28420==   by 0x987951: Upp::ChImageMaker::Make() const (Cham.cpp:52)
==28420==   by 0x99CA6A: Upp::scImageMaker::Make(Upp::Image&) const (MakeCache.cpp:32)
==28420==   by 0x99467D: Upp::LRUCache<Upp::Image, Upp::String>::Get(Upp::LRUCache<Upp::Image, Upp::String>::Maker const&) (Other.h:546)
==28420==   by 0x981263: Upp::MakeImage__(Upp::ImageMaker const&, bool) (MakeCache.cpp:126)

==28420==   by 0x978EFD: Upp::MakeImage(Upp::ImageMaker const&) (MakeCache.cpp:136)
==28420==   by 0x987A3E: Upp::ChDraw(Upp::Draw&, int, int, int, int, Upp::Image const&, Upp::Rect_<int> const&) (Cham.cpp:66)
==28420==   by 0x987B37: Upp::ChDraw(Upp::Draw&, Upp::Rect_<int> const&, Upp::Image const&, Upp::Rect_<int> const&) (Cham.cpp:73)
==28420==   by 0x989722: Upp::StdChLookFn(Upp::Draw&, Upp::Rect_<int> const&, Upp::Value const&, int) (Cham.cpp:328)
==28420== Uninitialised value was created by a stack allocation
==28420==   at 0x5C27C4: GuiMainFn_() (OpenCorpora.cpp:1981)

hex-value should be ignored.

2. You need to create a file with supressions.
--suppressions=<filename> suppress errors described in <filename>

Option below helps to create them.
--gen-suppressions=no|yes|all   print suppressions for errors? [no]

Subject: Re: Using Valgrind output
Posted by slashupp on Tue, 11 Jun 2019 17:47:46 GMT
View Forum Message <> Reply to Message

thx for responding

Q1:
I assume regarding 'debug info' you are referring to (mainmenu)[Build::Output mode..]
'Debug info level' is set to 'Full'
I select 'Full' for each of my modules in grid below
Recompiled everything and did Test in Valgrind again
The output did not change from what I describe in my original question

What should I do to get the output you describe?

Subject: Re: Using Valgrind output
Posted by Novo on Tue, 11 Jun 2019 17:58:48 GMT
View Forum Message <> Reply to Message

I always run valgrind from command line.
valgrind ./my_app args
valgrind has several tools, not just a memory checker, which is used by default.
valgrind is also my favorite profiler

If your code can be compiled with Clang, I'd recommend you to compile and test it with different

sanitizers.
GCC also supports a couple of sanitizers.

Sanitizers require recompilation, valgrind doesn't.

Subject: Re: Using Valgrind output
Posted by slashupp on Wed, 12 Jun 2019 06:18:49 GMT
View Forum Message <> Reply to Message

Novo,
I finally went so far as to install clang & llvm ..
set clang as compiler in Build methods and
added -fsanitize=leak to Debug link options for testing
compiled my project and try to run in debugger, but get:
Quote:Failed to obtain information about threads. Make sure your application posses debug info.
The
debugger and debugge proceses will be stoped!
I've set all to Full in [Build::Output mode], so what is this 'debug info'
it is complaining about and how do I fix it?

Subject: Re: Using Valgrind output
Posted by Novo on Wed, 12 Jun 2019 13:21:07 GMT
View Forum Message <> Reply to Message

slashupp wrote on Wed, 12 June 2019 02:18Novo,
I finally went so far as to install clang & llvm ..
set clang as compiler in Build methods and
added -fsanitize=leak to Debug link options for testing
compiled my project and try to run in debugger, but get:
Quote:Failed to obtain information about threads. Make sure your application posses debug info.
The
debugger and debugge proceses will be stoped!
I've set all to Full in [Build::Output mode], so what is this 'debug info'
it is complaining about and how do I fix it?

I attached my own build method for memory sanitizer.
I'm using "common options", both compile and link.
.USEMALLOC is required with only some of sanitizers.
AFAIK, -fsanitize=leak is a limited version of the memory sanitizer.

Make sure your application posses debug info.
It is likely that your app for some reason doesn't contain debug info.
It is easy to check:
1) find location of your executable by either

a) looking at the build console output, or
b) Build --> "Open output directory"
2) run "file ./your_app"
you should see the output similar to one below.

$ file ./my_app
./my_app: ELF 64-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=d5ee88481e49f7a1820a7f23711a256a27bbf01f, for GNU/Linux 3.2.0, stripped

The critical part is the word stripped
If you see it, then debug info was stripped out of your executable, or you compiled it without debug info enabled.
you should see not stripped if you want a debugger to work.

Sanitizers print out error messages into cerr, I believe, so you need to run your app from a console to be able to see the output. (or you can change settings in "Debug" --> "Run options")

what is this 'debug info'
'debug info' is a debug info level in a build method (Full/Minimal/None), or the same thing in Output mode
In "Output mode" left part of the dialog is responsible for the Debug configuration, and the right part is responsible for Release ...

Hope this helps.

File Attachments

1) CLANGcpp17msan.bm, downloaded 444 times

Subject: Re: Using Valgrind output
Posted by Novo on Wed, 12 Jun 2019 14:05:06 GMT
View Forum Message <> Reply to Message

Another thing.
TheIDE tends to put executables into different directories when you change something related to build options.
So, always pay attention to the output directory.
I'm always getting into troubles because of this.

Subject: Re: Using Valgrind output
Posted by slashupp on Mon, 17 Jun 2019 07:23:52 GMT
View Forum Message <> Reply to Message

Found the cause of my valgrind problems is a bug in valgrind itself:
see: https://bugs.debian.org/cgi-bin/bugreport.cgi?bug=903581

I downloaded/compiled/and installed the latest from
http://www.valgrind.org/downloads/current.html#current
and now all works as it should.

thx Novo
