Functions names in a relatively large shared library are not shown in cpu profile #625

**Closed**

[**alk**](https://github.com/alk) opened this issue on 23 Aug 2015 · 9 comments

窗体顶端

**Assignees**

@alk [**alk**](https://github.com/gperftools/gperftools/issues/assigned/alk)

窗体底端

窗体顶端

**Labels**

[**Priority-Medium**](https://github.com/gperftools/gperftools/labels/Priority-Medium)[**Status-New**](https://github.com/gperftools/gperftools/labels/Status-New)[**Type-Defect**](https://github.com/gperftools/gperftools/labels/Type-Defect)

窗体底端

窗体顶端

**Projects**

None yet

窗体底端

窗体顶端

**Milestone**

No milestone

窗体底端

**1 participant**

[@alk](https://github.com/alk)

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issue-102614939)

|  |
| --- |
| Originally reported on Google Code with ID 622  What steps will reproduce the problem?  1. Build a dynamic linking library with -g and -fno-pie.  2. Build an executable that is also compiled with -g and -fno-pie, and also linked  with the library with -g.  3. Link to libprofiler. ProfilerStart/ProfilerStop in the executable, then pprof --pdf  <application> <profile> > a.pdf  What is the expected output? What do you see instead?  Expected: All function names are retrieved from the dynamic library and demangled names  are shown in the profile.  Actual: Many function names in the shared library are shown as physical addresses like  "00007f54c09cbbf7"  What version of the product are you using? On what operating system?  Fedora 20, x86\_64.  gcc 4.8.2. libunwind 1.1.3.  Tried two versions of gperftools  1. gperftools/gperftools-devel-2.1.4-fc20.x86\_64 (from the repo)  2. gperftools 2.2, build from source myself.  Please provide any additional information below.  Both the shared library and the executable have debug symbols. Attached nm results  as well as the profile.  When I build the library as a static library and link the executable to it, I can get  the correct function names.  I tried several things, but no avail so far.  I guess it's related to the following issues.  https://code.google.com/p/gperftools/issues/detail?id=562  https://code.google.com/p/gperftools/issues/detail?id=586  So, I tried -fno-pie and the newest gperftools (2.2) from source build.  Nope, the problem is still there.  The symptoms described in Issue 586 seem quite similar to what I get.  This problem happens only with shared library.  I didn't have this issue until my shared library became relatively big (8MB).  I had no problems using gperftools/pprof otherwise.  It seems like gperftools fails to retrieve symbols from shared libraries for some reason.  I vaguely remember there is some option to get gperftools into verbose mode and describe  what symbols/why it couldn't load, is there?  The workaround so far is to build the library as a static library when I want to profile,  but not ideal:-(  Reported by hideaki.kimura on 2014-05-15 18:28:26  - \_Attachment: [exec.nm](https://storage.googleapis.com/google-code-attachments/gperftools/issue-622/comment-0/exec.nm)\_ - \_Attachment: [lib.nm](https://storage.googleapis.com/google-code-attachments/gperftools/issue-622/comment-0/lib.nm)\_ - \_Attachment: [tpcb\_experiment.prof](https://storage.googleapis.com/google-code-attachments/gperftools/issue-622/comment-0/tpcb\_experiment.prof)\_ |

@alk [**alk**](https://github.com/alk) self-assigned this [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#event-389766233)

@alk [**alk**](https://github.com/alk) added [**Type-Defect**](https://github.com/gperftools/gperftools/labels/Type-Defect) [**Priority-Medium**](https://github.com/gperftools/gperftools/labels/Priority-Medium) [**Status-New**](https://github.com/gperftools/gperftools/labels/Status-New) labels [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#event-389766234)

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issuecomment-133827148)

|  |
| --- |
| There's --debug option, but I'm not sure it will give you explanation why it's not working.  BTW, when trying 2.2 are you sure you've used pprof from 2.2 ? I'm asking because 2.2  has some fix for pprof about some addr2line problem.  Reported by alkondratenko on 2014-05-18 18:19:35 |

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issuecomment-133827212)

|  |
| --- |
| Thanks for the reply.  Yes, I used 2.2's pprof. I explicitly specified like  /home/<myname>/local/bin/pprof ... (I compiled/installed gperftools into ~/local)  Also, I confirmed via ldd that I surely linked to 2.2's libprofiler.so.  Where should I specify the --debug option? It's not command line arg for the application,  or is it? Does it assume LD\_PRELOAD?  Reported by hideaki.kimura on 2014-05-18 19:47:10 |

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issuecomment-133827268)

|  |
| --- |
| --debug is pprof option.  Reported by alkondratenko on 2014-05-18 19:47:51 |

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issuecomment-133827320)

|  |
| --- |
| I tried --debug on pprof, but it doesn't say anything about symbols.  I think the issue happens before pprof when gperftools retrieves symbols from the shared  library.  "00007f..." doesn't look like a logical function address, so it doesn't match anything  no matter how pprof tries. I wonder if the fix for base address adjustment (Issue 586)  is working correctly. Is there any debug option for ProfilerStart/ProfilerStop, not  pprof?  Reported by hideaki.kimura on 2014-05-18 22:57:22 |

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issuecomment-133827374)

|  |
| --- |
| libprofiler.so is not mapping addresses to functions. It's just writing addresses which  pprof then turns them into functions  Reported by alkondratenko on 2014-05-18 23:02:22 |

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issuecomment-133827438)

|  |
| --- |
| Oh, so pprof retrieves the function name from shared libraries?  Or, does pprof assumes that symbols of shared libraries are also included in the executable  (which is doable, but a bit tedious)?  Well, I guess it doesn't matter. I didn't get the .so function names resolved even  with:  pprof --pdf <.so file> <profile> > a.pdf  One interesting thing is that I didn't have this issue when the .so was small.  The size of .so has grown and after some point of time I started hitting this issue.  Also, seemingly a single function was profiled with several different addresses (all  00007f... with very close addresses). That sounds like PIE-related issue although I  specify fno-pic everywhere.  Any ideas?  Reported by hideaki.kimura on 2014-05-19 16:35:32 |

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issuecomment-133827487)

|  |
| --- |
| Some new information on this issue.  Seems like this issue is specific to Fedora 20. I don't get this issue on Fedora 19.  That means either something in the newer linux kernel (3.9 -> 3.11) or newer glibc  (2.17 -> 2.18). It doesn't seem like a gcc version issue as it works fine on Fedora  19 with the exact same version of gcc.  I'd appreciate giving it a try on Fedora 20 environment.  Reported by hideaki.kimura on 2014-06-08 02:20:41 |

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 23 Aug 2015](https://github.com/gperftools/gperftools/issues/625#issuecomment-133827544)

|  |
| --- |
| I'll need some large .so library for that. Currently I don't have time to deal with  that issue. It might be something with particular version of binutils. It would be  great if you could investigate this issue yourself.  For example, have you tried older versions of gperftools (2.1 or 2.0) just in case  ?  Reported by alkondratenko on 2014-06-28 20:16:52 |

[](https://github.com/alk)

Contributor

[**alk**](https://github.com/alk)commented [on 22 Feb 2016](https://github.com/gperftools/gperftools/issues/625#issuecomment-186980156)

|  |
| --- |
| Closing this as perl pprof implementation is deprecated in favor of golang version at github.com/google/pprof |

@alk [**alk**](https://github.com/alk) closed this [on 22 Feb 2016](https://github.com/gperftools/gperftools/issues/625#event-558903284)