Ruby trunk - Bug #9437

Build of ruby 2.1.0 fails on AIX 6.1

01/22/2014 12:19 AM - pedz (Perry Smith)

Status:	Feedback		
Priority:	Normal		
Assignee:	kanemoto (Yutaka Kanemoto)		
Target version:			
ruby -v:	-	**Backport:**	1.9.3: DONTNEED, 2.0.0: DONTNEED, 2.1: REQUIRED

Description

miniruby will not load and gets an error of:

```
linking miniruby
Could not load program ./miniruby:
        Dependent module libgmp.a(libgmp.so.10) could not be loaded.
Could not load module libgmp.a(libgmp.so.10).
System error: No such file or directory
make: *** [.rbconfig.time] Error 255
```

The dump -H of miniruby shows:

```
dump -H miniruby

miniruby:

                        ***Loader Section***
                     Loader Header Information
VERSION#         #SYMtableENT     #RELOCent        LENidSTR
0x00000001       0x00000811       0x000031ec       0x000000b9

#IMPfilID        OFFidSTR         LENstrTBL        OFFstrTBL
0x00000007       0x000318c8       0x0000942e       0x00031981


                        ***Import File Strings***
INDEX  PATH                            BASE                MEMBER
0      /gsa/ausgsa/projects/r/ruby/prvm/ruby-2.1.0/lib:/usr/lib:/lib

1                                      libpthread.a        shr_comm.o
2                                      libpthread.a        shr_xpg5.o
3                                      libgmp.a            libgmp.so.10
4                                      libcrypt.a          shr.o
5                                      libc.a              shr.o
6                                      librtl.a            shr.o
```

Note the dependency of libgmp. libgmp is not in /usr/lib or /lib but is in another directory that the compiler knows about via the prefix path but Ruby does not. So the link succeeds because ld knows how to find libgmp but the execution does not because of the embedded LIBPATH (element 0 above). This has come up before. I can work around the issue by various ways but I thought I should open an issue.

I'm happy to help out with debug or more data.

History

#1 - 01/22/2014 02:00 AM - nobu (Nobuyoshi Nakada)

- Description updated

- Status changed from Open to Feedback

Where is libgmp located in?

#2 - 01/22/2014 02:01 AM - nobu (Nobuyoshi Nakada)

- Backport changed from 1.9.3: UNKNOWN, 2.0.0: UNKNOWN, 2.1: UNKNOWN to 1.9.3: DONTNEED, 2.0.0: DONTNEED, 2.1: REQUIRED

#3 - 01/22/2014 02:03 AM - nobu (Nobuyoshi Nakada)

- Category set to platform/aix

- Target version set to 2.2.0

#4 - 01/25/2014 12:40 AM - pedz (Perry Smith)

miniruby is built with a command that looks like this:

```
gcc -O3 -fno-fast-math -ggdb3 -Wall -Wextra -Wno-unused-parameter -Wno-parentheses -Wno-long-long -Wno-missing
-field-initializers -Wunused-variable -Wpointer-arith -Wwrite-strings -Wdeclaration-after-statement -Wimplicit
-function-declaration -ansi -std=iso9899:199409  -L.  -Wl,-bE:ruby.imp -Wl,-brtl -Wl,-blibpath:/gsa/ausgsa/pro
jects/r/ruby/prvm/ruby-2.1.0/lib:/usr/lib:/lib  main.o dmydln.o miniinit.o miniprelude.o array.o bignum.o clas
s.o compar.o complex.o dir.o dln_find.o encoding.o enum.o enumerator.o error.o eval.o load.o proc.o file.o gc.
o hash.o inits.o io.o marshal.o math.o node.o numeric.o object.o pack.o parse.o process.o random.o range.o rat
ional.o re.o regcomp.o regenc.o regerror.o regexec.o regparse.o regsyntax.o ruby.o safe.o signal.o sprintf.o s
t.o strftime.o string.o struct.o time.o transcode.o util.o variable.o version.o compile.o debug.o iseq.o vm.o
vm_dump.o vm_backtrace.o vm_trace.o thread.o cont.o ascii.o us_ascii.o unicode.o utf_8.o newline.o flock.o str
lcpy.o strlcat.o setproctitle.o  dmyext.o   -lpthread -lgmp -ldl -lcrypt -lm   -o miniruby
```

If a -v flag is added, we see that gcc add these flags:

```
-L.
-L/gsa/ausgsa-p9/06/ruby/bin/../lib/gcc/powerpc-ibm-aix6.1.0.0/4.5.2
-L/gsa/ausgsa-p9/06/ruby/bin/../lib/gcc
-L/gsa/ausgsa-p9/06/ruby/bin/../lib/gcc/powerpc-ibm-aix6.1.0.0/4.5.2/../../..
```

which is why ld can find libgmp.a. But the compile line tells ld to ignore the -L flags when it creates the final executable and make the libpath be /gsa/ausgsa/projects/r/ruby/prvm/ruby-2.1.0/lib:/usr/lib:/lib Now, the executable depends upon things that it does not know how to find.

The reason (I've been told) for setting libpath is for security but I don't understand how that makes it more secure. A user can still set LIBPATH and get arbitrary libraries used by Ruby (I think).

Perhaps there is a way to ask gcc which paths it is going to add and we could all those as well.

#5 - 01/25/2014 07:40 AM - nobu (Nobuyoshi Nakada)

Perry Smith wrote:

> But the compile line tells ld to ignore the -L flags when it creates the final executable and make the libpath be /gsa/ausgsa/projects/r/ruby/prvm/ruby-2.1.0/lib:/usr/lib:/lib Now, the executable depends upon things that it does not know how to find.

Which option lets ld ignore default paths?

> The reason (I've been told) for setting libpath is for security but I don't understand how that makes it more secure. A user can still set LIBPATH and get arbitrary libraries used by Ruby (I think).

It should be for the case su with inherited environment variables

#6 - 01/27/2014 12:02 AM - pedz (Perry Smith)

This option

```
-Wl,-blibpath:/gsa/ausgsa/projects/r/ruby/prvm/ruby-2.1.0/lib:/usr/lib:/lib
```

sets the libpath inside the executable. "dump -H miniruby" will show

```
/gsa/ausgsa/projects/r/ruby/prvm/ruby-2.1.0/lib:/usr/lib:/lib
```

as the 0th item in the list of dependencies. If -blibpath is not given, then the list will include each path given by the -L option as well as the standard /usr/lib and /lib paths. I think the -L option that it producing the angst is "-L.". I wonder if it might be easiest to somehow remove that option.

I'm also looking for a method to get the list of directories gcc needs. I sent an email to the gcc list but they did not help. But I have an idea I need to work up. I hope to have an update to this bug report in a day or so.

#7 - 01/27/2014 12:46 AM - nobu (Nobuyoshi Nakada)

Perry Smith wrote:

> I think the -L option that it producing the angst is "-L.". I wonder if it might be easiest to somehow remove that option.

Could you confirm it?
I have no AIX environments.

> I'm also looking for a method to get the list of directories gcc needs.

Although I can't remember the exact name, gcc had an option -print-library-names or something.
Try gcc -help.

#8 - 01/27/2014 01:25 AM - pedz (Perry Smith)

As far as which -L option is causing the issue, I don't know. Most open source programs do not set libpath as Ruby is doing. Why Ruby is doing it is somewhere in the history of Ruby's discussions I would assume. su will work without setting libpath.

I tried the gcc help list and they (in effect) said there is nothing like I want. But I can figure out a way to get them if necessary.

Can you do some research and try to see why libpath is being set in the first place? I've always believed it would be better to not set it.

(By the way, when you update this issue, I don't see it in the mailing list. So I may be slow to reply. I've hit the "watch" button so maybe I will get emailed now.)

#9 - 01/27/2014 04:11 AM - kanemoto (Yutaka Kanemoto)

- ruby -v changed from trying to build 2.1.0 to -

Hi,

This is AIX's ld's behavior:

---

from man ld

```
       -LDirectory
            Adds Directory to the list of search directories used for finding libraries designated by the -l
            (lowercase letter L) flag. The list of directories, including the standard library directories, is
 also
            recorded in the output object file loader section for use by the system loader unless you use
            the -blibpath, -bnolibpath, or -bsvr4 option. You can repeat this flag.
```

So, this can not be resolved from gcc. Some open source projects are affected this behavior (i.e. http://archives.neohapsis.com/archives/bugtraq/2003-04/0385.html)

Since we are not able to remove -L. at this point, we need to use -blibpath to avoid including '.' from search path.

```
% gcc -print-search-dirs
install: /opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/4.7.2/
programs: =/opt/freeware/libexec/gcc/powerpc-ibm-aix7.1.0.0/4.7.2/:/opt/freeware/libexec/gcc/powerpc-ibm-aix7.
1.0.0/4.7.2/:/opt/freeware/libexec/gcc/powerpc-ibm-aix7.1.0.0/:/opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/4.
7.2/:/opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/:/opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/4.7.2/../../..
/../powerpc-ibm-aix7.1.0.0/bin/powerpc-ibm-aix7.1.0.0/4.7.2/:/opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/4.7.
2/../../../../powerpc-ibm-aix7.1.0.0/bin/
libraries: =/opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/4.7.2/:/opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/4
.7.2/../../../../powerpc-ibm-aix7.1.0.0/lib/powerpc-ibm-aix7.1.0.0/4.7.2/:/opt/freeware/lib/gcc/powerpc-ibm-ai
x7.1.0.0/4.7.2/../../../../powerpc-ibm-aix7.1.0.0/lib/:/opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/4.7.2/../.
./../powerpc-ibm-aix7.1.0.0/4.7.2/:/opt/freeware/lib/gcc/powerpc-ibm-aix7.1.0.0/4.7.2/../../../:/lib/powerpc-i
bm-aix7.1.0.0/4.7.2/:/lib/:/usr/lib/powerpc-ibm-aix7.1.0.0/4.7.2/:/usr/lib/
```

Should we add all of these directories listed in "libraries:" ?

--
Yutaka KANEMOTO
http://d.hatena.ne.jp/kinpoco/

#10 - 01/27/2014 06:07 AM - nobu (Nobuyoshi Nakada)

Thank you for the clarification.
Then should we enumerate all directories for other libraries, not only gmp?

#11 - 01/27/2014 06:08 AM - nobu (Nobuyoshi Nakada)

- Assignee set to kanemoto (Yutaka Kanemoto)

#12 - 01/27/2014 01:52 PM - pedz (Perry Smith)

I believe (rather strongly) that the ssh report is incorrect. I recall a test deep inside the loader that does not load libraries under dangerous conditions. I recall seeing this flaw myself but when I tried to take advantage of it, I could not. I finally reviewed the loader source code (which I can do) and figured out why. This is a rather vague memory and I am not sure of the details.

I need to go back and review that and recall the details.

My thought (which I plan to try and implement today or tomorrow) is to have a simple program but when that simple program is linked, tell gcc to use another script, maybe called "pretend-ld" in place of ld. pretend-ld will scan the arguments passed to it and emit a suitable list of directories to be included in libpath. We would use this during the configure stage. In essence, what we need to do (if you want to keep setting libpath) is to dig out the directories that gcc adds somehow. Another approach is to just link a simple program and use dump -H along with a script to dig out the libpath set in the executable.

#13 - 01/27/2014 06:32 PM - pedz (Perry Smith)

I have verified that the article mentioned above is correct (I've reproduced it). I am continuing on with trying to find a solution.

#14 - 01/27/2014 09:00 PM - pedz (Perry Smith)

Here are my thoughts. First, create a script shown below. I call the script get-path

```
#!/usr/bin/sh

echo 'int main() { return 0; }' > temp.c
gcc -o temp temp.c
for i in ` dump -H temp | awk '$1 == "0" {print $2;}' | tr ':' ' ' ` ; do
  if echo "$i" | grep -s '^/' > /dev/null ; then
    echo $i
  fi
done | tr '\n' ':' | sed -e 's%:$%%'
```

Now, in the configuration process, run the script and it produces output like:

```
/gsa/ausgsa-p9/06/ruby/bin/../lib/gcc/powerpc-ibm-aix6.1.0.0/4.5.2:/gsa/ausgsa-p9/06/ruby/bin/../lib/gcc:/gsa/
ausgsa-p9/06/ruby/bin/../lib/gcc/powerpc-ibm-aix6.1.0.0/4.5.2/../../..:/usr/lib:/lib
```

You can prepend any additional paths that you want, then use that where you have the -Wl,-blibpath:... argument. I get lost when trying to figure out Ruby's build process so I'm hoping you can take it from here.

#15 - 01/27/2014 10:51 PM - pedz (Perry Smith)

Yutaka Kanemoto wrote:

> Since we are not able to remove -L. at this point, ...

Why do we need -L. ?

(I'm not arguing that we don't. I just don't understand why we do.)

#16 - 01/30/2014 11:38 PM - pedz (Perry Smith)

This issue is bigger than I thought. While miniruby and perhaps ruby itself are fixed, other shared objects are not fixed. e.g.

```
dump -H ./.ext/powerpc-aix6.1.0.0/enc/euc_jp.so

./.ext/powerpc-aix6.1.0.0/enc/euc_jp.so:

                        ***Loader Section***
                     Loader Header Information
VERSION#         #SYMtableENT     #RELOCent        LENidSTR
0x00000001       0x0000000c       0x0000003c       0x00000104

#IMPfilID        OFFidSTR         LENstrTBL        OFFstrTBL
0x00000002       0x00000410       0x00000163       0x00000514


                        ***Import File Strings***
INDEX  PATH                           BASE                 MEMBER
0      .:/gsa/ausgsa/projects/r/ruby/prvm/ruby-2.1.0/lib:/gsa/ausgsa-p9/06/ruby/bin/../lib/gcc/powerpc-ibm-aix
6.1.0.0/4.5.2:/gsa/ausgsa-p9/06/ruby/bin/../lib/gcc:/gsa/ausgsa-p9/06/ruby/bin/../lib/gcc/powerpc-ibm-aix6.1.0
.0/4.5.2/../../..:/usr/lib:/lib
1                                     libruby.so
```

Of all of the shared objects built, only libruby.so has the corrected libpath. To do this right, not only does miniruby, ruby, and libruby.so need to be built right but all of the shared objects as well as any gem that is built needs to be built right.

I don't understand ruby's build process nor the process of building gem very well. I will continue to explore...

#17 - 01/05/2018 09:00 PM - naruse (Yui NARUSE)

- Target version deleted (2.2.0)