Ruby master - Bug #5676

miniruby linking error: undefined reference to ___stack_chk_guard

11/27/2011 12:48 PM - duerst (Martin Dürst)

<table>
<thead>
<tr>
<th>Status:</th>
<th>Closed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Priority:</td>
<td>Normal</td>
</tr>
<tr>
<td>Assignee:</td>
<td>kosaki (Motohiro KOSAKI)</td>
</tr>
<tr>
<td>Target version:</td>
<td>2.0.0</td>
</tr>
<tr>
<td>Backport:</td>
<td></td>
</tr>
</tbody>
</table>

**Description**

I get what I think is a linking error when linking miniruby. Below is a (shortened) copy of the output I get. This is on a clean checkout of trunk (using Ruby 1.8.7 as baseruby).

```
duerst@jougashima /cygdrive/c/Data/ruby-public
$ make
CC = gcc
LD = ld
LDSHARED = gcc -shared -s
CFLAGS = -O3 -g -Wall -Wextra -Werror=pointer-arith -Werror=write-strings -Werror=declaration-after-statement
-Xerror=implicit-function-declaration
XCFLAGS = -include ruby/config.h -include ruby/missing.h -D_FORTIFY_SOURCE=2 -fstack-protector -DRUBY_EXPORT
CPPFLAGS =   -I. -I.ext/include/i386-cygwin -I./include -I.
DLDFLAGS = -Wl,--enable-auto-image-base,--enable-auto-import -Wl,--out-implib=RbRuby191.dll.a cygruby191.def -Xlinker
--no-undefined
SOLIBS = cygruby191.res.o -lpthread -lrt -ldl -lcrypt
linking miniruby.exe
```

```
dmyencoding.o: In function set_encoding_const':
/cygdrive/c/Data/ruby-public/encoding.c:1473: undefined reference to _stack_chk_guard'
/cygdrive/c/Data/ruby-public/encoding.c:1520: undefined reference to `stack_chk_guard'
/cygdrive/c/Data/ruby-public/encoding.c:1520: undefined reference to `stack_chk_fail'
bignum.o: In function rb_str_to_inum':
/cygdrive/c/Data/ruby-public/bignum.c:763: undefined reference to _stack_chk_guard'
/cygdrive/c/Data/ruby-public/bignum.c:790: undefined reference to `stack_chk_guard'
/cygdrive/c/Data/ruby-public/bignum.c:790: undefined reference to `stack_chk_fail'
dir.o: In function dir_read':
/cygdrive/c/Data/ruby-public/dir.c:569: undefined reference to _stack_chk_guard'
/cygdrive/c/Data/ruby-public/dir.c:586: undefined reference to `stack_chk_guard'
/cygdrive/c/Data/ruby-public/dir.c:586: undefined reference to __stack_chk_fail'
```

```
vm_dump.o: In function control_frame_dump':
/cygdrive/c/Data/ruby-public/vm_dump.c:27: undefined reference to _stack_chk_guard'
/cygdrive/c/Data/ruby-public/vm_dump.c:148: undefined reference to `stack_chk_guard'
cont.o: In function cont_restore_0':
/cygdrive/c/Data/ruby-public/cont.c:733: undefined reference to `stack_chk_guard'
 unicode.o: In function onigenc_unicode_property_name_to_ctype':
/cygdrive/c/Data/ruby-public/enc/unicode.c:2087: undefined reference to `stack_chk_guard'
/cygdrive/c/Data/ruby-public/enc/unicode.c:2114: undefined reference to `stack_chk_guard'
/cygdrive/c/Data/ruby-public/enc/unicode.c:2114: undefined reference to __stack_chk_fail'
collect2: ld returned 1 exit status
make: *** [miniruby.exe] Error 1
```

duerst@jougashima /cygdrive/c/Data/ruby-public
$
I get what I think is a linking error when linking miniruby. Below is a (shortened) copy of the output I get. This is on a clean checkout of trunk (using Ruby 1.8.7 as baseruby).

duerst@jougashima /cygdrive/c/Data/ruby-public
$ make
Â Â Â Â CC

It works fine.

The issue occurs on mingw too, and is solved by the patch. But miniruby.exe still can't execute.
miniruby.exe: error while loading shared libraries: libssp-0.dll: cannot open shared object file: No such file or directory

I think -fstack-protector is too problematic and not worth on mingw.

As I'm new to FreeBSD so this may simply be my misconfiguration, but has anyone successfully built trunk with the gcc-4.6.2 package?

BTW, why isn't TRY_LD_FLAGS also used in https://github.com/ruby/ruby/blob/trunk/configure.in#L506-508 (I tried it but it doesn't solve the failure)
The symbol in question differs, so sounds like a runtime library mismatch. Does it fail with the default package?

If it occurs only with the non-default, and also simplified code without ruby, it should be reported to the gcc team.

FYI, I could build ruby on FreeBSD 9.0 amd64 with gcc 4.6 in ports.

Thanks for checking. Trunk builds fine using the old default gcc 4.2.1 and configure adds -fstack-protector to Makefile.

I think you’re right on runtime library mismatch, but shouldn’t it fail when miniruby is used later in the build process, not at miniruby link time? The following sequence definitely looks like a library mismatch...

```
ldconfig -rv | grep gcc
```

```

33:-lgcc_s.1 => /lib/libgcc_s.so.1
...
475:-lssp.0 => /usr/local/lib/gcc46/libssp.so.0
```

```
ldconfig -rv | grep ssp
```

```
35:-lssp.0 => /lib/libssp.so.0
475:-lssp.0 => /usr/local/lib/gcc46/libssp.so.0
```

```
diff -q /lib/libssp.so.0 /usr/local/lib/gcc46/libssp.so.0
```

```
Files /lib/libssp.so.0 and /usr/local/lib/gcc46/libssp.so.0 differ
```

```
stat -f '%z' /lib/libssp.so.0 /usr/local/lib/gcc46/libssp.so.0
```

```
6828
26880
```

```
objdump -tT /usr/local/lib/gcc46/libssp.so.0 | grep __stack_chk_fail
```

```
00000ba0 l text 0000019 __stack_chk_fail_local
00000b40 g text 0000028 __stack_chk_fail
00000b40 g D text 0000028 LIBSSP_1.0 __stack_chk_fail
```

```
objdump -tT /lib/libssp.so.0 | grep __stack_chk_fail
```

```
00000c00 g D text 00000027 LIBSSP_1.0 __stack_chk_fail
```

But shouldn’t gcc46’s default search dirs save the link like it appears to be happening when configure (using --with-gcc=gcc46) executes RUBY_TRY_CFLAGS(-fstack-protector, ...)?

```
gcc46 -print-search-dirs
install: /usr/local/lib/gcc46/gcc/i386-portbd-freebsd9.0/4.6.2/
```

I've overlooked something important or a FreeBSD post-install step after running sudo pkg_add -r gcc-4.6.2. Back to the RTFM-a-thon unless you
guys spot my mistake.

#9 - 02/20/2012 12:43 PM - nobu (Nobuyoshi Nakada)
- Status changed from Feedback to Closed

Jon Forums wrote:

I think you're right on runtime library mismatch, but shouldn't it fail when miniruby is used later in the build process, not at miniruby link time?

The mismatch between the compiler and the runtime library. It can't be deferred.

475: -lssp.0 => /usr/local/lib/gcc46/libssp.so.0

Then -fstack-protector should let gcc link that library.

Try "gcc46 -dumpspecs | grep -A1 "^\*link_ssp:", and if %fstack-protector: does not exist or no options is given after the colon, the spec is wrong. And you will see same error with the following simple code and -fstack-protector option, I guess:

```c
#include <stdlib.h>
#include <stdio.h>
int main() { printf("%p\n", alloca(102400)); return 0; }
```

If it's the case, this is primarily an issue of FreeBSD port.

But shouldn't gcc46's default search dirs save the link like it appears to be happening when configure (using --with-gcc=gcc46) executes RUBY_TRY_CFLAGS(-fstack-protector, ...)?

It should be done in the spec file, as mentioned above.