Ruby master - Feature #2565

adding hooks for better tracing

01/06/2010 11:15 PM - yugui (Yuki Sonoda)

Status:	Closed
Priority:	Normal
Assignee:	tenderlovemaking (Aaron Patterson)
Target version:	2.0.0

Description

=begin
Hi,

I made a commit that embeded dtrace probes into Ruby so that you can
profile a Ruby application at runtime. (r26235)

Adding probes had been approved by a Ruby developer's meeting,
however, the commit was little larger than what other committers
expected. I got some objection for the commit. [ruby-dev:39954]
In the end, I decided to temporarily revert the commit. (r26243)

I discussed how we should support dynamic runtime tracing, with ko1,
mame, naruse, unak and shyouhei. The problems of the commit were:

- the probes duplicated with the event_hook framework (rb_add_event_hook, Kernel#set_trace_func)
- Design of the probes were not verified enough.
  - more trial and error are necessary, to make it clear what is necessary to trace a Ruby application.

I accepted ko1's suggestion:

- reverting the commit
- adding some hooks for rb_add_event_hook().
- implementing probes for dynamic runtime tracing on the event_hook framework.
  - these probes can be implemented as a gem
  - I will aget a chance for trial and error.
  - The probes possibly will be merged into Ruby itself after enough designed and getting enough use cases.

Here is a patch to add the hooks I and ko1 talked about. (attached)
And here is an extension library that provides prove points to dtrace,
on top of the hooks. (http://github.com/yugui/vm_probes)

What do you think? Can I commit the patch I attached?

Thank you,
-- Yuki Sonoda (Yugui)

Attachment: adding-hooks.patch
=end

Related issues:

Related to Ruby master - Feature #1279: Add DTrace Probes	**Closed**	**03/13/2009**
Related to Ruby master - Feature #7022: add event hook for garbage collection	**Closed**	**09/14/2012**

Associated revisions

Revision f82d652f - 11/13/2012 06:13 AM - naruse (Yui NARUSE)

Fix dtrace commit r37631, it is [Feature #2565]

- configure.in: disable dtrace because it doesn't work on FreeBSD.
- common.mk (clean-local): rm probes.h.
- common.mk (parse.o): depend $(PROBES_H_INCLUDES).

- common.mk (.d.h): moved from Makefile.in and use BASERUBY.
- tool/gen_dummy_probes.rb: reimplemented with ruby because sed is not available on Windows Microsoft VC++ environment.

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@37636 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 37636 - 11/13/2012 06:13 AM - naruse (Yui NARUSE)

Fix dtrace commit r37631, it is [Feature #2565]

- configure.in: disable dtrace because it doesn't work on FreeBSD.
- common.mk (clean-local): rm probes.h.
- common.mk (parse.o): depend $(PROBES_H_INCLUDES).
- common.mk (.d.h): moved from Makefile.in and use BASERUBY.
- tool/gen_dummy_probes.rb: reimplemented with ruby because sed is not available on Windows Microsoft VC++ environment.

Revision 37636 - 11/13/2012 06:13 AM - naruse (Yui NARUSE)

Fix dtrace commit r37631, it is [Feature #2565]

- configure.in: disable dtrace because it doesn't work on FreeBSD.
- common.mk (clean-local): rm probes.h.
- common.mk (parse.o): depend $(PROBES_H_INCLUDES).
- common.mk (.d.h): moved from Makefile.in and use BASERUBY.
- tool/gen_dummy_probes.rb: reimplemented with ruby because sed is not available on Windows Microsoft VC++ environment.

Revision 37636 - 11/13/2012 06:13 AM - naruse (Yui NARUSE)

Fix dtrace commit r37631, it is [Feature #2565]

- configure.in: disable dtrace because it doesn't work on FreeBSD.
- common.mk (clean-local): rm probes.h.
- common.mk (parse.o): depend $(PROBES_H_INCLUDES).
- common.mk (.d.h): moved from Makefile.in and use BASERUBY.
- tool/gen_dummy_probes.rb: reimplemented with ruby because sed is not available on Windows Microsoft VC++ environment.

Revision 37636 - 11/13/2012 06:13 AM - naruse (Yui NARUSE)

Fix dtrace commit r37631, it is [Feature #2565]

- configure.in: disable dtrace because it doesn't work on FreeBSD.
- common.mk (clean-local): rm probes.h.
- common.mk (parse.o): depend $(PROBES_H_INCLUDES).
- common.mk (.d.h): moved from Makefile.in and use BASERUBY.
- tool/gen_dummy_probes.rb: reimplemented with ruby because sed is not available on Windows Microsoft VC++ environment.

Revision 37636 - 11/13/2012 06:13 AM - naruse (Yui NARUSE)

Fix dtrace commit r37631, it is [Feature #2565]

- configure.in: disable dtrace because it doesn't work on FreeBSD.
- common.mk (clean-local): rm probes.h.
- common.mk (parse.o): depend $(PROBES_H_INCLUDES).
- common.mk (.d.h): moved from Makefile.in and use BASERUBY.
- tool/gen_dummy_probes.rb: reimplemented with ruby because sed is not available on Windows Microsoft VC++ environment.

Revision 37636 - 11/13/2012 06:13 AM - naruse (Yui NARUSE)

Fix dtrace commit r37631, it is [Feature #2565]

- configure.in: disable dtrace because it doesn't work on FreeBSD.
- common.mk (clean-local): rm probes.h.
- common.mk (parse.o): depend $(PROBES_H_INCLUDES).
- common.mk (.d.h): moved from Makefile.in and use BASERUBY.
- tool/gen_dummy_probes.rb: reimplemented with ruby because sed is not available on Windows Microsoft VC++ environment.

History

#1 - 01/07/2010 08:07 PM - mame (Yusuke Endoh)

=begin
Hi,

2010/1/6 Yugui yugui@yugui.jp:

> What do you think? Can I commit the patch I attached?

All the event types you added are C-level events which set_trace_func can not monitor. But I think that RUBY_EVENT_RESCUE may be Ruby-level event because it is a language-level event like RUBY_EVENT_RAISE.

In addition, the modification of thread_reset_event_flags is just bug fix.
So, in advance, the part should be committed as another commit.

--
Yusuke ENDOH mame@tsg.ne.jp

=end

#2 - 03/19/2010 12:17 AM - mame (Yusuke Endoh)

=begin
Hi Yugui,

2010/1/6 Yugui yugui@yugui.jp:

> I made a commit that embeded dtrace probes into Ruby so that you can profile a Ruby application at runtime. (r26235)
>
> *snip*
>
> Here is a patch to add the hooks I and ko1 talked about. (attached)
> And here is an extension library that provides prove points to dtrace, on top of the hooks. (http://github.com/yugui/vm_probes)
>
> What do you think? ?Can I commit the patch I attached?

Any update here? I'm not against the feature.
We must determine whether 1.9.2 will aim to include the feature or not.

Rocky Bernstein offered two suggestions. [ruby-core:27449]
The first suggestion seems reasonable, but will take some time to discuss the API.
I guess it is better to leave it to 1.9.3, unfortunately.

--
Yusuke ENDOH mame@tsg.ne.jp

=end

#3 - 04/02/2010 08:45 AM - znz (Kazuhiro NISHIYAMA)

- Category set to core

- Status changed from Open to Assigned

- Assignee set to yugui (Yuki Sonoda)

- Target version set to 2.0.0

- Start date set to 01/06/2010

=begin

=end

#4 - 10/27/2010 01:47 AM - porter_md (Mark Porter)

=begin
Any update on this issue? Will we see for 1.9.3?
=end

#5 - 06/30/2011 05:41 AM - tenderlovemaking (Aaron Patterson)

- File probe_hacks.patch added

Hi, I am interested in this functionality too. I'm interested in this because I have an application where object creation has increased. The increased objects are hash, array, and string literals. The existing trace methods will not be triggered on allocation of those objects, and the scripting / filtering / speed of dtrace is useful to me.

I've been patching trunk with the attached patch in order to debug my current issues. I don't think it's as complete as yugui's commit though.

What can I do to get dtrace support in trunk? How can I help?

#6 - 10/01/2011 12:04 PM - porter_md (Mark Porter)

I cant figure out for the life of me why this issue is not getting more traction. We're seeing more and more environments getting dtrace (nodejs now has dtrace probes) and ruby has removed them, seemingly never to return. We've got people willing to help (Aaron Patterson!!), help us help Ruby. Please?

#7 - 10/03/2011 11:14 AM - naruse (Yui NARUSE)

Mark's concern, why Ruby doesn't include DTrace support, is reasonable, I explain why.
It is because tracing is so important that we can't implement it without careful design.
But we know such too careful way annoys you.
So we are considering that 2.0 (not 1.9.3) includes experimental tracing support.

#8 - 02/14/2012 02:54 AM - tenderlovemaking (Aaron Patterson)

- Assignee changed from yugui (Yuki Sonoda) to tenderlovemaking (Aaron Patterson)

Since I really want this feature, I should update this ticket with my progress.

I've ported the Joyant probes to trunk (and added a few). You can view the progress I've made here:

https://github.com/tenderlove/ruby/tree/probes

I use this branch on a daily basis with no noticeable performance penalties, but I have a few tasks remaining before I want to call my work "done".

1. Probe stability declarations

One concern that ko1 had was people relying on DTrace probe APIs. DTrace allows us to declare the stability of any particular probe. This allows us to tell DTrace consumers how much they should rely on that particular API. I like the API of the probes I've added (and the Joyant probes), so I don't want to change them. However, it may be useful to declare them as unstable until we've had multiple ruby releases that contain the probes (and the community is happy with the API).

1. Tests

I've started writing tests for the probes, but there is a challenge. DTrace can only be enabled by a user with elevated permissions, so the tests must be run with sudo. It's easy enough to write tests that will only execute when the user has the correct permissions, but I'm not sure what we should do about CI.

1. Autotools generating probes.h

I don't understand autotools very well, so I need help with this task. At rubyconf, nobu helped me patch Makefile.in to generate the probes.h file, but it doesn't seem to automatically generate on my system. I'm not entirely sure what the problem is.

https://github.com/tenderlove/ruby/commit/3f645ce71e29859256063265cfd03ffd52d38dd8

When I run rm probes.h; make, the probes.h file is not regenerated. I don't understand what is wrong. Any help would be greatly appreciated.

That's all the updates I have for now!

#9 - 04/25/2012 01:44 AM - sax (Eric Saxby)

Could the experimental probes be added in 1.9.3, so that developers would gain experience using them and have a more solid basis to recommend better probe design in 2.0?

#10 - 05/02/2012 08:25 AM - tenderlovemaking (Aaron Patterson)

- File dtrace.diff added

Hi,

I've attached a patch that adds dtrace probes to trunk. If nobody objects, I will apply.

The patch doesn't contain every probe I want, but I think it's in a good place to merge to trunk. I can add more probes later. :)

#11 - 05/02/2012 10:23 AM - ko1 (Koichi Sasada)

(2012/05/02 8:25), tenderlovemaking (Aaron Patterson) wrote:

> I've attached a patch that adds dtrace probes to trunk. If nobody objects, I will apply.
>
> The patch doesn't contain every probe I want, but I think it's in a good place to merge to trunk. I can add more probes later. :)

I don't make any objection.

Let us clear. Is that specification? I mean, on 2.1 and later, "should
keep same dtrace interface support?" If we need keep interface, we need
review carefully.

--
// SASADA Koichi at atdot dot net

#12 - 05/02/2012 10:29 AM - ko1 (Koichi Sasada)

(2012/05/02 8:25), tenderlovemaking (Aaron Patterson) wrote:

> File dtrace.diff added

One more comment.

I think macro names (for example, RUBY_FUNCTION_RETURN_ENABLED()) should
rename to another one (for example, RUBY_PROBLE_FUNCTION_RETURN_ENABLED).

Because, of course, RUBY FUNCTIONs can RETURN (ENABLED) :)

--
// SASADA Koichi at atdot dot net

#13 - 05/02/2012 10:29 AM - ko1 (Koichi Sasada)

(2012/05/02 10:23), SASADA Koichi wrote:

> rename to another one (for example, RUBY_PROBLE_FUNCTION_RETURN_ENABLED).

Sorry, RUBY_DTRACE_FUNCTION_RETURN_ENABLED is better for me. Because it selects platform.

--
// SASADA Koichi at atdot dot net

#14 - 05/02/2012 11:53 AM - Anonymous

- File noname added

On Wed, May 02, 2012 at 10:18:53AM +0900, SASADA Koichi wrote:

> (2012/05/02 8:25), tenderlovemaking (Aaron Patterson) wrote:
>
> > I've attached a patch that adds dtrace probes to trunk. If nobody objects, I will apply.
> >
> > The patch doesn't contain every probe I want, but I think it's in a good place to merge to trunk. I can add more probes later. :)
>
> I don't make any objection.
>
> Let us clear. Is that specification? I mean, on 2.1 and later, "should keep same dtrace interface support?" If we need keep interface, we need review carefully.

DTrace allows us to specify the stability of the probes. I've declared the provider name of "ruby" to be stable. We don't declare any modules or functions, so I've declared them as stable. The probes (e.g. function-entry), as well as the type and number of arguments to the probes are declared as unstable, so users are advised not to depend on them.

You can find the stability declarations here:

https://github.com/tenderlove/ruby/blob/probes/probes.d#L26-30

I think declaring them unstable is the best conservative approach. If we find them to be good over the long term, we can change the stability declaration in later releases of Ruby.

--
Aaron Patterson
http://tenderlovemaking.com/

#15 - 05/02/2012 11:53 AM - Anonymous

- File noname added

On Wed, May 02, 2012 at 10:25:13AM +0900, SASADA Koichi wrote:

> (2012/05/02 10:23), SASADA Koichi wrote:
>
> > rename to another one (for example, RUBY_PROBLE_FUNCTION_RETURN_ENABLED).
>
> Sorry, RUBY_DTRACE_FUNCTION_RETURN_ENABLED is better for me. Because it selects platform.

That's fine! I will rename them. :-)

--
Aaron Patterson
http://tenderlovemaking.com/

#16 - 05/02/2012 01:23 PM - ko1 (Koichi Sasada)

Hi,

(2012/05/02 11:38), Aaron Patterson wrote:

> DTrace allows us to specify the stability of the probes. I've declared the provider name of "ruby" to be stable.

I agree.

> We don't declare any modules or functions, so I've declared them as stable.

I'm not sure about it. You mean that we'll declare "there are no modules or functions" ?

> The probes (e.g. function-entry), as well as the type and number of arguments to the probes are declared as unstable, so users are advised not to depend on them.

I agree. Or declare specification as "Ruby 2.0.0" (specific implementation) and make warning that "there are possibilities of changing probes after Ruby 2.0.1 or later".

> You can find the stability declarations here:
>
> https://github.com/tenderlove/ruby/blob/probes/probes.d#L26-30

Thanks.

I have a question: We have lazy sweep which run many short scattered sweep process. Should we measure such a thing?

> I think declaring them unstable is the best conservative approach. If we find them to be good over the long term, we can change the stability declaration in later releases of Ruby.

I agree.

Thanks,
Koichi

--
// SASADA Koichi at atdot dot net

#17 - 05/03/2012 02:12 AM - tenderlovemaking (Aaron Patterson)

I've updated the patch to rename all macros to RUBY_DTRACE_*

#18 - 05/03/2012 02:29 AM - Anonymous

- File noname added

On Wed, May 02, 2012 at 01:01:17PM +0900, SASADA Koichi wrote:

> Hi,
>
> (2012/05/02 11:38), Aaron Patterson wrote:
>
> > DTrace allows us to specify the stability of the probes. I've declared the provider name of "ruby" to be stable.
>
> I agree.
>
> > We don't declare any modules or functions, so I've declared them as stable.
>
> I'm not sure about it. You mean that we'll declare "there are no modules or functions" ?

That's correct. We are not allowed to define modules or functions, so I've declared that our non-existent modules and functions are stable.

> The probes (e.g. function-entry), as well as the type and number of arguments to the probes are declared as unstable, so users are advised not to depend on them.

I agree. Or declare specification as "Ruby 2.0.0" (specific implementation) and make warning that "there are possibilities of changing probes after Ruby 2.0.1 or later".

I believe that is implied since we've said the api is not stable.

> You can find the stability declarations here:
>
> https://github.com/tenderlove/ruby/blob/probes/probes.d#L26-30

Thanks.

I have a question: We have lazy sweep which run many short scattered sweep process. Should we measure such a thing?

I think it's fine to measure. I've added probes in all the places where we gather GC statistics (near GC_PROF_SWEEP_TIMER_START and GC_PROF_SWEEP_TIMER_STOP). The nice thing is that the DTrace system will provide the time, so we don't need to write any C code to calculate timing.

> I think declaring them unstable is the best conservative approach. If we find them to be good over the long term, we can change the stability declaration in later releases of Ruby.

I agree.

Great! If you're happy with my patch, I'll apply. :-)

--
Aaron Patterson
http://tenderlovemaking.com/

#19 - 05/03/2012 08:04 AM - andhapp (Anuj Dutta)

I was just looking at Joyent's ruby dtrace page and saw ruby-probe (Probe that can be fired from ruby code). However, I didn't see it in Aaron's implementation here https://github.com/tenderlove/ruby/blob/probes/probes.d. Is there a reason why that's not included? Please enlighten. Thanks.

#20 - 05/03/2012 09:53 AM - Anonymous

- File noname added

On Thu, May 03, 2012 at 08:04:37AM +0900, andhapp (Anuj Dutta) wrote:

> Issue #2565 has been updated by andhapp (Anuj Dutta).
>
> I was just looking at Joyent's ruby dtrace page and saw ruby-probe (Probe that can be fired from ruby code). However, I didn't see it in Aaron's implementation here https://github.com/tenderlove/ruby/blob/probes/probes.d. Is there a reason why that's not included? Please enlighten. Thanks.

We shouldn't add it today because we don't know the license of the Joyant patches, and someone must be the maintainer. A bigger reason is that this is a feature that can be added via a gem download. Adding interpreter probes cannot be done via gem download.

Maybe we will expose dtrace through ruby in the stdlib someday, but I don't think we should today.

--
Aaron Patterson
http://tenderlovemaking.com/

#21 - 05/05/2012 08:11 AM - andhapp (Anuj Dutta)

Aaron,

Thanks. I had no idea about the licence.

Is your branch stable(ish) to try it out?

#22 - 05/10/2012 07:15 PM - vo.x (Vit Ondruch)

> We shouldn't add it today because we don't know the license of the
> Joyant patches

The license is known: http://svn.joyent.com/opensource/dtrace/ruby/Joyent_copyright_notice.txt

#23 - 05/10/2012 08:11 PM - vo.x (Vit Ondruch)

I don't feel very comfortable to see dummy_probes.h as part of the patch while probes.h is generated by dtrace during build. Either both files should be already in patch, or both files should be created during compilation. I prefer the latter.

#24 - 05/11/2012 11:28 PM - vo.x (Vit Ondruch)

- File 0001-Generate-probes.h-from-probes.d.patch added

Hi,

The attached patch reimplements dummy_probe_gen.rb, which is now capable to generate the probes.h directly from the probes.d.

#25 - 05/14/2012 08:53 AM - Anonymous

- File noname added

On Fri, May 11, 2012 at 11:28:58PM +0900, vo.x (Vit Ondruch) wrote:

> Issue #2565 has been updated by vo.x (Vit Ondruch).
>
> File 0001-Generate-probes.h-from-probes.d.patch added
>
> Hi,
>
> The attached patch reimplements dummy_probe_gen.rb, which is now capable to generate the probes.h directly from the probes.d.

Thanks! I'll apply this to my branch.

--
Aaron Patterson
http://tenderlovemaking.com/

#26 - 05/14/2012 05:54 PM - vo.x (Vit Ondruch)

- File 0001-Generate-probes.h-from-probes.d.patch added

I am attaching updated patch which fixes handling of one-line comments in probes.d.

#27 - 06/30/2012 02:25 PM - yugui (Yuki Sonoda)

- File Dtrace_SystemTap support for Ruby.pptx added

Uploads a presentation slide for the feature triage meeting on 2012-07-21.

#28 - 06/30/2012 02:42 PM - yugui (Yuki Sonoda)

- File deleted (Dtrace_SystemTap support for Ruby.pptx)

#29 - 06/30/2012 02:44 PM - yugui (Yuki Sonoda)

- File Dtrace_SystemTap support for Ruby (3).pptx added

#30 - 07/01/2012 08:28 AM - trans (Thomas Sawyer)

- File 6594.pdf added

- File 6609.pdf added

#31 - 07/01/2012 08:47 AM - trans (Thomas Sawyer)

- File 5922.pdf added

#32 - 07/01/2012 08:53 AM - trans (Thomas Sawyer)

- File 5922.pdf added

Here is better 5922. Please disregard previous.

#33 - 07/01/2012 12:23 PM - trans (Thomas Sawyer)

I was in a rush and mistook this for the issue dedicated to proposal submissions. The last three posts do not pertain to this. Sorry.

#34 - 07/02/2012 03:06 AM - mame (Yusuke Endoh)

Yugui-san, your slide is received. Thank you.
(Using a link is unfair :-)

I wonder if this proposal is not accepted by matz.
What this proposal needs is not matz's approval, but the conclusion of API design. I guess.

Anyway, please make a presentation yourself at the meeting :-)

--
Yusuke Endoh mame@tsg.ne.jp

#35 - 07/02/2012 11:53 PM - Anonymous

- File noname added

On Mon, Jul 02, 2012 at 03:06:59AM +0900, mame (Yusuke Endoh) wrote:

> Issue #2565 has been updated by mame (Yusuke Endoh).
>
> Yugui-san, your slide is received. Thank you.
> (Using a link is unfair :-)
>
> I wonder if this proposal is not accepted by matz.
> What this proposal needs is not matz's approval, but the conclusion of API design. I guess.

I thought we concluded API design in [ruby-core:44793], and
[ruby-core:44813]. Also fixed preprocessor constants in
[ruby-core:44792].

ko1 said he was OK to merge in [ruby-core:44788].

AFAIK, the only open issue (that Vit raised [ruby-core:44980]) is that
we shouldn't check in the dummy probe header file. I agree with Vit, so
I want to update my patch before merge. :-)

--
Aaron Patterson
http://tenderlovemaking.com/

#36 - 07/03/2012 01:31 AM - vo.x (Vit Ondruch)

> AFAIK, the only open issue (that Vit raised [ruby-core:44980]) is that
> we shouldn't check in the dummy probe header file. I agree with Vit, so
> I want to update my patch before merge. :-)

Well, I proposed two options, either have both versions in SCM or keep it out of the SCM. I'm still in favor of the latter, however I realized later that there is one major disadvantage. If we want to avoid the need of Ruby during build from tarball, as it is possible now, then the release manager has to have available either DTrace or SystemTap on his/her system. Not sure if that might be fulfilled :/ If not, then both files should be pre-generated by its maintainer and stored in SCM.

In other words, we should think also about associated work-flow and release management.

#37 - 07/03/2012 06:23 AM - Anonymous

- File noname added

On Tue, Jul 03, 2012 at 01:31:53AM +0900, vo.x (Vit Ondruch) wrote:

> Issue #2565 has been updated by vo.x (Vit Ondruch).
>
> > AFAIK, the only open issue (that Vit raised [ruby-core:44980]) is that
> > we shouldn't check in the dummy probe header file. I agree with Vit, so
> > I want to update my patch before merge. :-)
>
> Well, I proposed two options, either have both versions in SCM or keep it out of the SCM. I'm still in favor of the latter, however I realized later that there is one major disadvantage. If we want to avoid the need of Ruby during build from tarball, as it is possible now, then the release manager has to have available either DTrace or SystemTap on his/her system. Not sure if that might be fulfilled :/ If not, then both files should be pre-generated by its maintainer and stored in SCM.
>
> In other words, we should think also about associated work-flow and release management.

I don't think the header file should be generated when the tarball is
being built, but when the user installs Ruby (this is how PostgreSQL
does their dtrace headers).

I'm working on a sed script to convert the .d file to a .h file in the
case that the user doesn't have dtrace on their system. Then the
release manager doesn't need to have Ruby or DTrace on their system when
making the tar.

How does that sound?

--
Aaron Patterson
http://tenderlovemaking.com/

#38 - 07/03/2012 08:23 AM - ko1 (Koichi Sasada)

(2012/07/02 23:32), Aaron Patterson wrote:

> ko1 said he was OK to merge in [ruby-core:44788].

I want to discuss what probes and parameters we need.
(Maybe I said same thing before)

Your patch will be a beginning of this discussion. I hope there is a
summary about it on anywhere, for example, wiki page.

I think implementation is not a matter (we have enough time to do).

(but I can't understand sed script ; I prefer ruby script :)

--
// SASADA Koichi at atdot dot net

#39 - 07/03/2012 01:26 PM - vo.x (Vit Ondruch)

ko1 (Koichi Sasada) wrote:

> (but I can't understand sed script ; I prefer ruby script :)

As I said, there are several conditions which should be fulfilled. If you don't want to

1) store the generated files in SCM
2) generate the scripts during preparation of tarball by release manager
3) want to have Ruby available during build from tarball

than you don't have other option than to use something (in this case sed) what is probably available on your system during the build time. So although I would prefer Ruby, it does not fulfill the 3 conditions above.

#40 - 07/19/2012 03:52 AM - tenderlovemaking (Aaron Patterson)

- File dtrace.patch added

I've attached a new patch that contains a sed script for converting the .d file to a .h file.

When a user tries to install Ruby, if they do not have dtrace, the sed script will convert the .d file to a dummy header file. The dummy header file will be included, and the compiler will remove all dtrace probes.

I have created a wiki page to describe the DTrace probes. It includes the probe names, arguments, stability, and a short description for each probe:

https://bugs.ruby-lang.org/projects/ruby/wiki/DTraceProbes

I hope we can apply this patch soon!

#41 - 07/29/2012 11:53 AM - ko1 (Koichi Sasada)

Sorry for late response.

(2012/07/03 13:26), vo.x (Vit Ondruch) wrote:

> ko1 (Koichi Sasada) wrote:
>
> > (but I can't understand sed script ; I prefer ruby script :)
>
> As I said, there are several conditions which should be fulfilled. If you don't want to
>
> 1) store the generated files in SCM
> 2) generate the scripts during preparation of tarball by release manager
> 3) want to have Ruby available during build from tarball
>
> than you don't have other option than to use something (in this case sed) what is probably available on your system during the build time. So although I would prefer Ruby, it does not fulfill the 3 conditions above.

I don't check what dtrace needs correctly, but is not enough?:

(1) build miniruby with dummy header (manged in SCM)
(2) generate dtrace related files with miniruby
(3) build ruby with (2)

--
// SASADA Koichi at atdot dot net

#42 - 07/29/2012 12:23 PM - ko1 (Koichi Sasada)

Sorry for late response.

(2012/07/19 3:52), tenderlovemaking (Aaron Patterson) wrote:

> https://bugs.ruby-lang.org/projects/ruby/wiki/DTraceProbes

my comments:

> ruby:::function-entry(classname, methodname, filename, lineno);
> ruby:::function-return(classname, methodname, filename, lineno);

set_trace_func spearates "call" and "c-call" (and return).
No need to separate on dtrace?

> ruby:::require-entry(requiredfile, filename, lineno);
> ruby:::require-return(filename);
> ruby:::load-entry(loadedfile, filename, lineno);
> ruby:::load-return(filename);

Do we need require' andload' both needed? (It depends on usecase)
What happen on exception?
(There is a same question on `function-entry')

> ruby:::object-create-start(classname, filename, lineno);
> ruby:::object-create-done(classname, filename, lineno);

(1) How to detect object creation and finish of creation?
(2) I can't accept your patch on insns.def on string and array.

IIRC, you suggest that String creation trace and Array creation trace
and so on. I write examples:

ruby:::string-create(filename, lineno, size)
ruby:::array-create(filename, lineno, size)
ruby:::hash-create(filename, lineno, size)
ruby:::object-create(filename, lineno, classname)

size is string or array size.
classname is a classname of object.

I'm not sure we need string-modified (size) are needed.

ruby:::object-collected(object_id) will be help. To use it correctly,
object-create needs to pass object_id. But it will be complicated.

> ruby:::gc-begin();
> ruby:::gc-end();

How to define GC begin and end?

> ruby:::gc-sweep-begin();
> ruby:::gc-sweep-end();

How to define GC sweep begin and end?

> ruby:::line(filename, lineno);

Your patch depends on the trace' instruction. I plan to removetrace'
instruction on default (if I can implement it). It will conflicts with
your proposed patch.
Or I shouldn't make such optimizations?

--
// SASADA Koichi at atdot dot net

#43 - 07/29/2012 12:23 PM - ko1 (Koichi Sasada)

(2012/07/29 12:02), SASADA Koichi wrote:

> > ruby:::line(filename, lineno);
> > Your patch depends on the trace' instruction. I plan to removetrace'
> > instruction on default (if I can implement it). It will conflicts with
> > your proposed patch.
> > Or I shouldn't make such optimizations?

One approach is add dtrace mode on command line.
If the option is enabled, the process will be *all* dtrace probe ready.
But a bit slow (~10%).
Without this option, the process will be not all dtrace probe ready.
This option can be enabled in running with implementation effort.

I think this approach is against dtrace philosophy.
Is it acceptable?

--
// SASADA Koichi at atdot dot net

#44 - 07/30/2012 04:23 AM - Anonymous

Dne 29.7.2012 4:36, SASADA Koichi napsal(a):

> Sorry for late response.
>
> (2012/07/03 13:26), vo.x (Vit Ondruch) wrote:
>
> > ko1 (Koichi Sasada) wrote:
> >
> > > (but I can't understand sed script ; I prefer ruby script :)

> > As I said, there are several conditions which should be fulfilled. If you don't want to
>
> 1) store the generated files in SCM
> 2) generate the scripts during preparation of tarball by release manager
> 3) want to have Ruby available during build from tarball
>
> than you don't have other option than to use something (in this case sed) what is probably available on your system during the build time.
> So although I would prefer Ruby, it does not fulfill the 3 conditions above.
> I don't check what dtrace needs correctly, but is not enough?:

(1) build miniruby with dummy header (manged in SCM)
(2) generate dtrace related files with miniruby
(3) build ruby with (2)

If somebody can help with this setup, I definitely won't oppose. But I'm
afraid I don't know how to do it by myself :/

#45 - 07/30/2012 05:53 AM - Anonymous

- File noname added

On Sun, Jul 29, 2012 at 11:36:49AM +0900, SASADA Koichi wrote:

> Sorry for late response.
>
> (2012/07/03 13:26), vo.x (Vit Ondruch) wrote:
>
> > ko1 (Koichi Sasada) wrote:
> >
> > > (but I can't understand sed script ; I prefer ruby script :)
> >
> > As I said, there are several conditions which should be fulfilled. If you don't want to
> >
> > 1) store the generated files in SCM
> > 2) generate the scripts during preparation of tarball by release manager
> > 3) want to have Ruby available during build from tarball
> >
> > than you don't have other option than to use something (in this case sed) what is probably available on your system during the build time.
> > So although I would prefer Ruby, it does not fulfill the 3 conditions above.
>
> I don't check what dtrace needs correctly, but is not enough?:
>
> (1) build miniruby with dummy header (manged in SCM)
> (2) generate dtrace related files with miniruby
> (3) build ruby with (2)

We could do this, but it means that every time we change probes.d, we
would have to generate a new dummy header file and check it in. I think
the purpose of the sed script is to eliminate the "generate new dummy
header file and check it in" step.

--
Aaron Patterson
http://tenderlovemaking.com/

#46 - 07/30/2012 05:59 AM - Anonymous

- File noname added

On Sun, Jul 29, 2012 at 12:02:27PM +0900, SASADA Koichi wrote:

> Sorry for late response.
>
> (2012/07/19 3:52), tenderlovemaking (Aaron Patterson) wrote:
>
> > https://bugs.ruby-lang.org/projects/ruby/wiki/DTraceProbes

my comments:

ruby:::function-entry(classname, methodname, filename, lineno);
ruby:::function-return(classname, methodname, filename, lineno);

set_trace_func spearates "call" and "c-call" (and return).
No need to separate on dtrace?

I don't think we need to separate these. Information like that hasn't been useful to me when debugging or profiling my ruby programs.

ruby:::require-entry(requiredfile, filename, lineno);
ruby:::require-return(filename);
ruby:::load-entry(loadedfile, filename, lineno);
ruby:::load-return(filename);

Do we need require' andload' both needed? (It depends on usecase)

I'm not sure we need both. What I *want* is a way to know when Ruby starts and finishes loading a file.

What happen on exception?

The raise probe is fired.

(There is a same question on `function-entry')

ruby:::object-create-start(classname, filename, lineno);
ruby:::object-create-done(classname, filename, lineno);

(1) How to detect object creation and finish of creation?

rb_obj_alloc has probes. It wraps object allocation.

(2) I can't accept your patch on insns.def on string and array.

Why? You want specific string / array / hash probes (like below)?

IIRC, you suggest that String creation trace and Array creation trace and so on. I write examples:

ruby:::string-create(filename, lineno, size)
ruby:::array-create(filename, lineno, size)
ruby:::hash-create(filename, lineno, size)
ruby:::object-create(filename, lineno, classname)

size is string or array size.
classname is a classname of object.

These seem good. I can implement them.

I'm not sure we need string-modified (size) are needed.

I tried doing this once, but it seemed like the patch was too large. I wanted to start small. :)

ruby:::object-collected(object_id) will be help. To use it correctly, object-create needs to pass object_id. But it will be complicated.

ruby:::gc-begin();
ruby:::gc-end();

How to define GC begin and end?

Nari-san already has defined GC begin and end for GC::Profiler. See the GC_PROF_TIMER_START and GC_PROF_TIMER_STOP macros:

https://github.com/ruby/ruby/blob/trunk/gc.c#L180
https://github.com/ruby/ruby/blob/trunk/gc.c#L199

My patch just adds the DTrace probes to his existing macros.

```
ruby:::gc-sweep-begin();
ruby:::gc-sweep-end();
```

> How to define GC sweep begin and end?

Same here. Nari-san has already defined GC_PROF_SWEEP_TIMER_START and GC_PROF_SWEEP_TIMER_STOP for GC::Profiler. My patch just adds probes by his existing macros.

```
ruby:::line(filename, lineno);
```

> Your patch depends on the trace' instruction. I plan to removetrace' instruction on default (if I can implement it). It will conflicts with your proposed patch.
> Or I shouldn't make such optimizations?

I think we should remove this. I added it for backwards compatibility with Joyant's DTrace probes, but I never use this. Besides backwards compatibility, I don't think it's a useful probe.

--
Aaron Patterson
http://tenderlovemaking.com/

#47 - 10/27/2012 05:10 AM - ko1 (Koichi Sasada)

ping. progress? :)

#48 - 10/27/2012 06:01 AM - tenderlovemaking (Aaron Patterson)

- File probes.diff added

Oops, yes. I've updated the probes. I removed line and added some probes around tracing instruction sequences. It's basically the same information that the vm_collect_usage functions collect, but you don't need to recompile ruby to collect that information.

I've attached the updated patch, and I'll work on changing the wiki.

#49 - 11/02/2012 05:07 AM - tenderlovemaking (Aaron Patterson)

- File probes.diff added

I've attached an updated version of the dtrace probes.

@mame-san may I commit this to trunk? ko1 (Koichi Sasada) said I should ask you. :-)

#50 - 11/02/2012 08:20 AM - mame (Yusuke Endoh)

Aaron,

At the developers' meeting, matz entirely leaved this up to Yugui.
I must say "please confirm Yugui"... But I know she is absence a while,
and I believe she would accept it willingly.

So, please go ahead. ASAP. It will be included in preview2.
(Note that it may be reverted if it causes a problem.)

--
Yusuke Endoh mame@tsg.ne.jp

#51 - 11/13/2012 03:13 PM - naruse (Yui NARUSE)

- Status changed from Assigned to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r37636.
Yuki, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

Fix dtrace commit r37631, it is [Feature #2565]

- configure.in: disable dtrace because it doesn't work on FreeBSD.
- common.mk (clean-local): rm probes.h.
- common.mk (parse.o): depend $(PROBES_H_INCLUDES).
- common.mk (.d.h): moved from Makefile.in and use BASERUBY.
- tool/gen_dummy_probes.rb: reimplemented with ruby because sed is not available on Windows Microsoft VC++ environment.

Files

probe_hacks.patch	3.23 KB	06/30/2011	tenderlovemaking (Aaron Patterson)
dtrace.diff	31.8 KB	05/02/2012	tenderlovemaking (Aaron Patterson)
noname	500 Bytes	05/02/2012	Anonymous
noname	500 Bytes	05/02/2012	Anonymous
noname	500 Bytes	05/03/2012	Anonymous
noname	500 Bytes	05/03/2012	Anonymous
0001-Generate-probes.h-from-probes.d.patch	1.57 KB	05/11/2012	vo.x (Vit Ondruch)
noname	500 Bytes	05/14/2012	Anonymous
0001-Generate-probes.h-from-probes.d.patch	1.58 KB	05/14/2012	vo.x (Vit Ondruch)
Dtrace_SystemTap support for Ruby (3).pptx	33.5 KB	06/30/2012	yugui (Yuki Sonoda)
6594.pdf	77.8 KB	07/01/2012	trans (Thomas Sawyer)
6609.pdf	77 KB	07/01/2012	trans (Thomas Sawyer)
5922.pdf	73.3 KB	07/01/2012	trans (Thomas Sawyer)
5922.pdf	73.2 KB	07/01/2012	trans (Thomas Sawyer)
noname	500 Bytes	07/02/2012	Anonymous
noname	500 Bytes	07/03/2012	Anonymous
dtrace.patch	32.4 KB	07/19/2012	tenderlovemaking (Aaron Patterson)
noname	500 Bytes	07/30/2012	Anonymous
noname	500 Bytes	07/30/2012	Anonymous
probes.diff	43.1 KB	10/27/2012	tenderlovemaking (Aaron Patterson)
probes.diff	43.9 KB	11/02/2012	tenderlovemaking (Aaron Patterson)