ruby blocks due to unavoidable getrandom without GRND_NONBLOCK

06/09/2018 05:05 AM - kevinoid (Kevin Locke)

Status: Closed
Priority: Normal
Assignee:
Target version: ruby -v: 2.5.1p57
Backport: 2.3: REQUIRED, 2.4: REQUIRED, 2.5: REQUIRED

Description
Following the instructions in the Rubygems FAQ I added the following to ~/.bashrc:

```bash
if which ruby >/dev/null && which gem >/dev/null; then
  PATH="$(ruby -e 'puts Gem.user_dir')/bin:$PATH"
fi
```

Unfortunately, this causes login to block for several seconds to minutes (depending on available entropy sources) because ruby -e 'puts Gem.user_dir' makes two getrandom syscalls without the GRND_NONBLOCK flag. On Linux v4.17-rc2 and later, this causes ruby, and therefore the login process, to block until the RNG is fully initialized.

Arguably Rubygems could provide/recommend a way to get the user GEM dir without invoking Ruby. That would solve the specific login problem reported above. However, since even ruby -v makes two getrandom syscalls, I suspect this may cause difficult to diagnose hangs in startup or login scripts for many users and that it is a desirable use case to support, which is why I am reporting it here.

Some relevant history: r51182 added getrandom, r51374 added GRND_NONBLOCK to address Bug #11395 (similar to this issue), r52808 removed GRND_NONBLOCK in some cases for SecureRandom.

Thanks for considering,
Kevin

Associated revisions
Revision 8389f8c3 - 06/10/2018 06:33 AM - nobu (Nobuyoshi Nakada)
random.c: fix need_secure flags

- random.c (fill_random_seed): do not need to be secure, to get rid of blocking at the start-up time.
  [ruby-core:87462] [Bug #14837]
- random.c (random_raw_seed): expected to be a cryptographically secure, as documented.

Revision 63624 - 06/10/2018 06:33 AM - nobu (Nobuyoshi Nakada)
random.c: fix need_secure flags

- random.c (fill_random_seed): do not need to be secure, to get rid of blocking at the start-up time.
  [ruby-core:87462] [Bug #14837]
- random.c (random_raw_seed): expected to be a cryptographically secure, as documented.

Revision 63624 - 06/10/2018 06:33 AM - nobu (Nobuyoshi Nakada)
random.c: fix need_secure flags

- random.c (fill_random_seed): do not need to be secure, to get
random.c (random_raw_seed): expected to be a cryptographically secure, as documented.

History

#1 - 06/09/2018 09:05 AM - shevegen (Robert A. Heiler)

Makes sense.

As for Gem.user_dir:

Arguably Rubygems could provide/recommend a way to get the user GEM dir without invoking Ruby.

The other way may be to have ruby directly support it since gems are also bundled with ruby (and bundler already is or soon-to-be is).

For example perhaps RbConfig.user_dir or something similar. I am not necessarily suggesting this (and in any way, it should then go into a new issue request), but I think various assumptions by rubygems were also made when rubygems was more of a separate add-on.

#2 - 06/10/2018 06:32 AM - nobu (Nobuyoshi Nakada)

- Backport changed from 2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN to 2.3: REQUIRED, 2.4: REQUIRED, 2.5: REQUIRED

#3 - 06/10/2018 06:34 AM - nobu (Nobuyoshi Nakada)

- Status changed from Open to Closed

Applied in changeset trunk|r63624.

random.c: fix need_secure flags

- random.c (fill_random_seed): do not need to be secure, to get rid of blocking at the start-up time.
  [ruby-core:87462] [Bug #14837]

- random.c (random_raw_seed): expected to be a cryptographically secure, as documented.

#4 - 06/11/2018 01:17 AM - shyouhei (Shyouhei Urabe)

Let me leave a weak concern that I do not fully understand the impact of this changeset.

I recommend some reviews by cryptographic experts about it.

#5 - 06/11/2018 05:02 PM - kevinoid (Kevin Locke)

Thanks for the quick response and fix! Sorry I didn't see the changes sooner. (I didn't get an email notification, will investigate.)

If SecureRandom requires cryptographically random numbers, removing GRND_NONBLOCK will cause security issues when there is insufficient entropy, since it will get numbers which are insufficiently random. Is kosaki (Motohiro KOSAKI) still active? Perhaps he could weigh in since this effectively reverts his change to add GRND_NONBLOCK? I'll email him.

#6 - 06/11/2018 05:53 PM - kevinoid (Kevin Locke)

After re-reading the diff more closely I realized I had misunderstood. As I now understand, r63624 has the effect of adding GRND_NONBLOCK for Random.new_seed and the internal seeds and removing it for Random.urandom, which is probably what was originally intended in r52808. Seems like a good fix to me. The only potential issue I can see is that the internal seed is used to protect against hash table algorithmic complexity attacks (see r17465) so it might be possible to attack programs started before the system gains sufficient entropy to be unpredictable.

Thanks again!