Due to the precision of timestamp that Psych dumps, dumped time loaded by Psych is not always identical to the original time.

```ruby
require 'psych'
time = Time.at(Time.now.sec)
p time == (Psych.load Psych.dump time) #=> true

time = Time.now
p time == (Psych.load Psych.dump time) #=> false (in my environment)

time = Time.at(Time.now.sec, Rational(1, 3))
p time == (Psych.load Psych.dump time) #=> false
```

According to the YAML timestamp specification [[http://yaml.org/type/timestamp.html]], the standard format cannot represent time in rational. Thus, to fully express Ruby's Time object, we have to use a format other than the standard YAML timestamp.

```
#1 - 04/04/2010 10:02 AM - tenderlovemaking (Aaron Patterson)
- Priority changed from Normal to 3

hmmm. Is this necessary? You can dump your time object as a rational if you need that kind of accuracy:

```
$ irb
irb(main):001:0> require 'psych'
=> true
irb(main):002:0> time = Time.at(Time.now.sec, Rational(1, 3))
=> 1969-12-31 16:00:28 -0800
irb(main):003:0> yaml = Psych.dump(time.to_r)
=> "--- !ruby/object:Rational
denominator: 3000000
numerator: 84000001"
irb(main):004:0> time == Time.at(Psych.load(yaml))
=> true
irb(main):005:0>
```

#2 - 04/05/2010 10:24 AM - wishdev (John Higgins)

I don't believe any change is required here. Psych isn't trying to dump Ruby objects for reloading into Ruby - it's trying to dump Ruby objects so they can be loaded back by ANY language that has a YAML parser. As with most language-independent specs - YAML unfortunately doesn't have as much precision as Ruby does in this area - but for YAML it's more important to be language-independent than full precision over and above the spec.

If one wishes to dump an object and reload it with full fidelity - isn't that the job of Marshall?

As a side note - JSON exhibits the exact same behavior here - again it's trying to be language-independent rather than full precision.

#3 - 04/05/2010 01:50 PM - naruse (Yui NARUSE)
2010/4/5 Tanaka Akira akr@fsij.org:

2010/4/5 Aaron Patterson aaron@tenderlovemaking.com:

How should this problem be fixed? If a time object when dumped as YAML
does not use the YAML time format, that would be surprising to me. Is
my assumption wrong? I'd like to leave it the way it works now.

If the fractional seconds are representable in the decimal system,
it should be representable in the YAML time format.

For example, Ruby obtains the current time using clock_gettime if available.
clock_gettime returns struct timespec which has nano-second resolution.
It needs 9 digits to preserve the information.
But psych dumps only 6 digits.
This is why Psych.load(Psych.dump(Time.now)) doesn't round trip.

FreeBSD has bintime which is 2**(-64) second resolution.
It is also representable in the decimal system.
(It may need 64 digits, though.)
Ruby generates such time in Socket::AncillaryData#timestamp.

YAML's timestamp (ISO 8601)'s fraction can have non-limited digits.
http://yaml.org/type/timestamp.html

--
NARUSE, Yui
naruse@airemix.jp
=end

#4 - 04/21/2010 07:52 PM - mame (Yusuke Endoh)
=begin
Hi,

Is this really a bug?

Syck has emitted 6-digits for fraction of Time.
Psych is compatible to syck.
So I think that this is just a feature request.

Aaron, you are the maintainer. Please make decision:

1) this is a bug; you'll fix this soon (before 1.9.2)
2) this is a feature request, and
2-1) accepted; you'll implement this at 1.9.3 or later
2-2) rejected; you won't change it

BTW, I'm neutral for this suggestion.
I like human-readable yaml, while I can understand what akr said.

--
Yusuke Endoh mame@tsg.ne.jp
=end

#5 - 04/22/2010 07:34 AM - tenderlovemaking (Aaron Patterson)
- Target version set to 2.0.0

=begin
I choose 2-1. I agree this is a feature request. I think I can fix it, but I don't want to promise for 1.9.2.
=end

#6 - 07/03/2010 09:17 AM - Anonymous
- Status changed from Open to Closed
- % Done changed from 0 to 100

=begin
This issue was solved with changeset r28531.
=end
Tomo, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.
=end