Ruby trunk - Bug #4374

[ext/openssl] ASN1.decode wrong for infinite length values

02/07/2011 02:52 AM - MartinBosslet (Martin Bosslet)

Status:	Closed		
Priority:	Normal		
Assignee:	MartinBosslet (Martin Bosslet)		
Target version:	1.9.3		
ruby -v:	ruby 1.9.2p136 (2010-12-25 revision 30365) [i686-linux]	**Backport:**	2.3: UNKNOWN, 2.4: UNKNOWN, 2.5: UNKNOWN

Description

=begin
Hi all,

ASN.1 decoding behaves incorrectly for DER encodings with infinite length values. Two examples:

```
require 'openssl'
require 'pp'

eoc = OpenSSL::ASN1::EndOfContent.new
int = OpenSSL::ASN1::Integer.new (1)

inner = OpenSSL::ASN1::Sequence.new([int, eoc])
inner.infinite_length = true

outer = OpenSSL::ASN1::Sequence.new([inner, eoc])
outer.infinite_length = true

asn1 = OpenSSL::ASN1.decode(outer.to_der)

pp asn1
```

=> #,
#,
#]>]>

The end of content DER for the outer Sequence is incorrectly stored with the values
of the inner sequence. Although after encoding the resulting DER will be correct, the
structure should rather look like this:

#,
#]>,
#]>

Another example:

```
require 'openssl'
require 'pp'

eoc = OpenSSL::ASN1::EndOfContent.new
oct = OpenSSL::ASN1::OctetString.new ("\x01")

inner = OpenSSL::ASN1::Constructive.new([oct, eoc], OpenSSL::ASN1::OCTET_STRING)
inner.infinite_length = true

outer = OpenSSL::ASN1::Constructive.new([inner, eoc], OpenSSL::ASN1::OCTET_STRING)
outer.infinite_length = true

asn1 = OpenSSL::ASN1.decode(outer.to_der)

pp asn1
```

=> ,
#,
#]>]>]>]>

Here it's worse, because when calling asn1.to_der it will even result in an error:

test.rb:17:in to_der': invalid constructed encoding (OpenSSL::ASN1::ASN1Error)
from test.rb:17:ineach'
from test.rb:17:in to_der'
from test.rb:17:in'

The problem are the defaults for tagging and tag_class in ossl_asn1_initialize that are not intuitive and are defaults for tagged DER values instead of "normal" values.

The correct structure for the above would look like this:

#,
#]>,
#]>

The attached patch fixes the problems and has also "more natural" defaults for ossl_asn1_initialize.

Regards,
Martin
=end

Associated revisions

Revision e7d04f4b - 05/22/2011 09:01 PM - emboss

- ext/openssl/ossl_asn1.c: Fix decoding of infinite length values. Simplified ossl_asn1_decode0 by splitting it into three separate functions. Add tests. [Ruby 1.9 - Bug #4374][ruby-core:35123]

git-svn-id: svn+ssh://ci.ruby-lang.org/ruby/trunk@31700 b2dd03c8-39d4-4d8f-98ff-823fe69b080e

Revision 31700 - 05/22/2011 09:01 PM - emboss

- ext/openssl/ossl_asn1.c: Fix decoding of infinite length values. Simplified ossl_asn1_decode0 by splitting it into three separate functions. Add tests. [Ruby 1.9 - Bug #4374][ruby-core:35123]

Revision 31700 - 05/22/2011 09:01 PM - emboss

- ext/openssl/ossl_asn1.c: Fix decoding of infinite length values. Simplified ossl_asn1_decode0 by splitting it into three separate functions. Add tests. [Ruby 1.9 - Bug #4374][ruby-core:35123]

Revision 31700 - 05/22/2011 09:01 PM - emboss

- ext/openssl/ossl_asn1.c: Fix decoding of infinite length values. Simplified ossl_asn1_decode0 by splitting it into three separate functions. Add tests. [Ruby 1.9 - Bug #4374][ruby-core:35123]

Revision 31700 - 05/22/2011 09:01 PM - emboss

- ext/openssl/ossl_asn1.c: Fix decoding of infinite length values. Simplified ossl_asn1_decode0 by splitting it into three separate functions. Add tests. [Ruby 1.9 - Bug #4374][ruby-core:35123]

Revision 31700 - 05/22/2011 09:01 PM - emboss

- ext/openssl/ossl_asn1.c: Fix decoding of infinite length values. Simplified ossl_asn1_decode0 by splitting it into three separate functions. Add tests. [Ruby 1.9 - Bug #4374][ruby-core:35123]

History

#1 - 02/07/2011 04:46 AM - MartinBosslet (Martin Bosslet)

=begin
I verified that with the current code it's not possible to construct a Constructive

instance without tagging if the tag_class is passed as an explicit parameter.
Tagging will be set to :EXPLICIT if it was passed as nil. That's why

if (infinite && !(tag == V_ASN1_SEQUENCE || tag == V_ASN1_SET)){
asn1data = rb_funcall(cASN1Constructive,
rb_intern("new"),
4,
value,
INT2NUM(tag),
Qnil,
ID2SYM(tag_class));

in ossl_asn1_decode0 will always produce a Constructive with :EXPLICIT tagging, and reencoding a parsed ASN.1 value will fail or produce incorrect output.
At first I thought changing the defaults in ossl_asn1_initialize would be more intuitive but not essential to the fix - but now I think it has become mandatory.
=end

#2 - 05/12/2011 08:11 AM - MartinBosslet (Martin Bosslet)

- Assignee set to MartinBosslet (Martin Bosslet)

#3 - 05/23/2011 06:01 AM - Anonymous

- Status changed from Open to Closed

- % Done changed from 0 to 100

This issue was solved with changeset r31700.
Martin, thank you for reporting this issue.
Your contribution to Ruby is greatly appreciated.
May Ruby be with you.

- ext/openssl/ossl_asn1.c: Fix decoding of infinite length values. Simplified ossl_asn1_decode0 by splitting it into three separate functions. Add tests. [Ruby 1.9 - Bug #4374]

#4 - 05/23/2011 10:32 AM - MartinBosslet (Martin Bosslet)

Hi,

I fixed this bug and tried to simplify ossl_asn1_decode0 and improve its performance.

Here is what I did:

- removed the "once" parameter
- added sanity checks that verify that all bytes are actually read when parsing is finished
- tried to reduce complexity by splitting ossl_asn1_decode0 into three separate methods to make it easier to spot the code that handles constructed values and the code that handles primitive values
- ossl_asn1_decode now returns an Array just in the case of parsing constructed values, otherwise it returns the plain value to reduce overall allocated objects
- changed the behavior of ossl_asn1_initialize slightly to allow the construction of infinite length primitive values in the form of a Constructive
- replaced rb_intern with static symbols to improve performance
- allocate and initialize ASN1Data (and sub-classes) instances in C to improve performance and save additional VM roundtrips
- initialize those instances additionally with tag and tag_class to avoid hash look-ups each time in ossl_asn1_initialize
- added some tests for edge cases (I'll add more soon)

The performance gain is noticeable - in my tests (certificates, CMS signatures) the current implementation was roughly twice as fast as that of 1.9.2p180.

I would be happy about comments and improvements.

Regards,
Martin

Files

fix_asn1.diff	3.62 KB	02/07/2011	MartinBosslet (Martin Bosslet)