KMIP 32-bit Binary Alignment Proposal

To: OASIS KMIP Technical Committee
From: Matt Ball, Sun Microsystems, Inc.
Date: April 29, 2009
Version: 3~~2~~
Purpose: To propose a change to the binary encoding such that each part is aligned to a 4-byte boundary

Revision History

- Version 1, 2009-04-29: Initial version
- Version 2, 2009-04-29: Added clarifying text to the end of 9.1.1.3 concerning the padded length of Strings and Structures.
- Version 3, 2009-04-30: Changed order of Item Type list to place 'Structure' at top and to move all items down.

Introduction

The binary encoding as defined in the 0.98 version of the KMIP draft does not maintain alignment to 4-byte boundaries within the message structure. This causes problems on hard-aligned processors, such as the ARM, that are not able to easily access memory on addresses that are not aligned to 4 bytes. When unaligned memory contents are requested, either the compiler has to add extra instructions to perform two aligned memory accesses and reassemble the data, or the processor has to take a 'trap' (i.e., an interrupt generated on unaligned memory accesses) to correctly assemble the memory contents. Either of these options results in reduced performance.

This proposal suggests ways to improve the performance on hard-aligned processors by aligning all data structures to 4-byte boundaries.

Summary of Proposed Changes

This proposal includes the following changes to the KMIP 0.98 draft submission to the OASIS KMIP TC:

- Change the alignment of the KMIP binary encoding such that each part is aligned to a 4-byte boundary. This is done by:
  - Change the size of the Item Type field from one to four bytes.
  - Require that all Item Value fields be padded with zero, one, two, or three bytes of the value 00 such that the length of the Value field is a multiple of 4 bytes. The Item Length still contains the correct, unpadded length of the Item.
- Change the length of the Item Value for Binary types to be 4 bytes (instead of 1 byte), with hex values 00000000 for false and 00000001 for true.
- Change the format of the Big Integer Item Type to require that the Item Value be padded with sign-extended bytes on the left (i.e., most significant bytes) such that the total length is a multiple of 4 bytes.

Proposed Changes

The following text shows proposed changes to the KMIP 0.98 draft as accepted by the KMIP TC on April 24, 2009, with underline to indicate additions and ~~strikethrough~~ to indicate deletions.

9 Message Encoding

To support different transport protocols and different client capabilities, a number of message-encoding mechanisms are supported.

9.1 TTLV Encoding

In order to minimize the resource impact on potentially low-function clients, one encoding mechanism to be used for protocol messages is a simplified TTLV (Tag, Type, Length, Value) scheme.

The scheme is designed to minimize the CPU cycle and memory requirements of clients that must encode or decode protocol messages. Minimizing bandwidth over the transport mechanism is considered to be of lesser importance.

9.1.1 TTLV Encoding Fields

Every Data object encoded by the TTLV scheme consists of 4 items, in order:

9.1.1.1 Item Tag

An Item Tag is a 32-bit binary unsigned integer, transmitted big endian, which contains a number that designates the specific Protocol Field or Object that the TTLV object represents. To ease debugging, and to ensure that malformed messages are detected more easily, all tags contain the value 42 in hex as the high order (first) byte. Tags defined by this specification contain hex 00 in the second byte. Extensions, which are permitted, but not defined in this specification, contain the value 01 hex in the second byte. A list of defined Item Tags is in Section **Error! Reference source not found.**~~9.1.3.1~~ .

9.1.1.2 Item Type

An Item Type is a 4-byte (32-bit) field containing a coded value which indicates the data type of the data object. The allowed values are:

Data Type	**Coded Value in Hex**
Structure	00000001
Integer	0000000~~1~~2
Long Integer	0000000~~2~~3
Big Integer	0000000~~3~~4
Enumeration	0000000~~4~~5
Boolean	0000000~~5~~6
Text String	0000000~~6~~7
Octet String	0000000~~7~~8
Date-Time	0000000~~8~~9
Interval	0000000~~9~~A
~~Structure~~	~~80~~

9.1.1.3 Item Length

An Item Length is a 32-bit binary integer, transmitted big-endian, containing the number of bytes in the Item Value.

Allowed values are:

Data Type	**Length**
Integer	4
Long Integer	8
Big Integer	Varies, multiple of 4
Enumeration	4
Boolean	~~1~~ 4
Text String	Varies
Octet String	Varies
Date-Time	8
Interval	4
Structure	Varies, multiple of 4

If the Item Type is a ~~Big Integer,~~ Text String or Octet String, then the Item Length must be the number of unpadded bytes in the string. Strings must not be null-terminated, but must be padded such that the transmitted length of the Item Value is a multiple of 4 bytes (see 9.1.1.4). If the Item Type is a structure, then the Item Length is the total length of all of the sub-items contained in the structure, including any padding. If the Item Type is a Big Integer, then the Item Length must be the number bytes in the string after padding with leading bytes to make the length a multiple of 4 bytes.

9.1.1.4 Item Value

The item value is a sequence of bytes containing the value of the data item, depending on the type:

- Integers are encoded as 4-byte long (32 bit) binary signed numbers in 2's complement notation, transmitted big-endian.
- Long Integers are encoded as 8-byte long (64 bit) binary signed numbers in 2's complement notation, transmitted big-endian.
- Big Integers are encoded as a sequence of 8-bit bytes, in 2's complement notation, transmitted big-endian. If the length is not a multiple of 4 bytes (32 bits), then the Big Integer shall be padded with enough leading sign-extended bytes to make the length a multiple of 4 bytes (32 bits).
- Enumerations are encoded as 4-byte long (32 bit) binary unsigned numbers transmitted big-endian.
- Booleans are encoded as a 4-byte~~, which~~ value that must either contain the ~~binary~~ hexadecimal value 00000000, indicating the boolean value *False*, or the ~~binary~~ hexadecimal value 00000001, transmitted big-endian, indicating the boolean value *True*. Values other than 00000000 or 00000001 are to be interpreted as *True*, but are deprecated.
- Text Strings are sequences of bytes encoding character values according to the UTF-8 encoding standard. There must be no null-termination at the end of such strings.
- Octet Strings are sequences of bytes containing individual unspecified 8 bit binary values.
- Date-Time values are encoded as 8-byte long (64 bit) binary signed numbers, transmitted big-endian. They are POSIX Time values (described in IEEE Standard 1003.1) extended to a 64 bit value to eliminate the "Year 2038 problem". The value is expressed as the number of seconds from a time epoch, which is 00:00:00 GMT January 1st, 1970. This value has a resolution of 1 second. All Date-Time values are expressed as UTC values.

- Intervals are encoded as 4-byte long (32 bit) binary unsigned numbers, transmitted big-endian. They have a resolution of 1 second.
- Structure Values are encoded as the concatenated encodings of the elements of the structure. All structures defined in this specification must have all of their fields encoded in the order in which they appear in their respective structure descriptions.

If the Item Length contains a value that is not a multiple of 4, then the contents of the Item Value shall be padded with one, two, or three trailing '00' bytes such that the transmitted length is a multiple of 4 bytes (32 bits).

9.1.2 Examples

These examples are assumed to be encoding a Protocol Object whose tag is ~~42~~000020. The examples are shown as a sequence of bytes in hexadecimal notation:

- An Integer containing the decimal value 8:

  42 00 00 20 | 00 00 00 01 | 00 00 00 04 | 00 00 00 08

- A Long Integer containing the decimal value 123456789000000000:

  42 00 00 20 | 00 00 00 02 | 00 00 00 08 | 01 B6 9B 4B A5 74 92 00

- A Big Integer containing the decimal value 1234567890000000000000000000:

  42 00 00 20 | 00 00 00 03 | 00 00 00 0C | 03 FD 35 EB 6B C2 DF 46 18 08 00 00

- An Enumeration with value 255:

  42 00 00 20 | 00 00 00 04 | 00 00 00 04 | 00 00 00 FF

- A Boolean with the value *True*:

  42 00 00 20 | 00 00 00 05 | 00 00 00 0~~4~~1 | 00 00 00 01

- A Text String:

  42 00 00 20 | 00 00 00 06 | 00 00 00 0B | 48 65 6C 6C 6F 20 57 6F 72 6C 64 00

- An Octet String:

  42 00 00 20 | 00 00 00 07 | 00 00 00 03 | 01 02 03 00

- A Date-Time, containing the value for Friday, March 14, 2008, 11:56:40 GMT:

  42 00 00 20 | 00 00 00 08 | 00 00 00 08 | 00 00 00 00 47 DA 67 F8

- An Interval, containing the value for 10 days:

  42 00 00 20 | 00 00 00 09 | 00 00 00 04 | 00 0D 2F 00

- A Structure containing an Enumeration, value 254, followed by an Integer, value 255, having tags 42000004 and 42000005 respectively:

  42 00 00 20 | 00 00 00 80 | 00 00 00 20~~1A~~ | 42 00 00 04 | 00 00 00 04 | 00 00 00 04 | 00 00 00 FE | 42 00 00 05 | 00 00 00 01 | 00 00 00 04 | 00 00 00 FF