Key Management Interoperability Protocol JSON Profile Version 1.0

Working Draft 01

30 October 2012

Technical Committee:
OASIS Key Management Interoperability Protocol (KMIP) TC

Chairs:
Robert Griffin (robert.griffin@rsa.com), EMC Corporation
Subhash Sankuratripati (Subhash.Sankuratripati@netapp.com), NetApp

Editors:
Tim Hudson (tjh@cryptsoft.com), Cryptsoft

Related work:
This specification is related to:

- *Key Management Interoperability Protocol Profiles Version 1.0*. 01 October 2010. OASIS Standard. http://docs.oasis-open.org/kmip/profiles/v1.0/os/kmip-profiles-1.0-os.html.
- *Key Management Interoperability Protocol Specification Version 1.1*. Latest version. http://docs.oasis-open.org/kmip/spec/v1.1/kmip-spec-v1.1.html
- *Key Management Interoperability Protocol Use Cases Version 1.1*. Latest version. http://docs.oasis-open.org/kmip/usecases/v1.1/kmip-usecases-v1.1.html
- *Key Management Interoperability Protocol Usage Guide Version 1.1*. Latest version. http://docs.oasis-open.org/kmip/ug/v1.1/kmip-ug-v1.1.html

Abstract:
Describes a JSON encoding alternative to TTLV encoding.

Status:
This Working Draft (WD) has been produced by one or more TC Members; it has not yet been voted on by the TC or approved as a Committee Draft (Committee Specification Draft or a Committee Note Draft). The OASIS document Approval Process begins officially with a TC vote to approve a WD as a Committee Draft. A TC may approve a Working Draft, revise it, and re-approve it any number of times as a Committee Draft.

Table of Contents

1 Introduction

For normative definition of the elements of KMIP see the KMIP Specification ([KMIP-SPEC-1_0 and KMIP-SPEC-1_1]) and the KMIP Profiles ([KMIP-PROF]).

Illustrative guidance for the implementation of KMIP clients and servers is provided in the KMIP Usage Guide ([KMIP-UG]) and KMIP Use Cases ([KMIP_UC]).

This profile defines the necessary encoding rules for the transport of KMIP messages encoded in JSON ([RFC4627]) rather than TTLV.

1.1 Terminology

The key words “MUST”, “MUST NOT”, “REQUIRED”, “SHALL”, “SHALL NOT”, “SHOULD”, “SHOULD NOT”, “RECOMMENDED”, “MAY”, and “OPTIONAL” in this document are to be interpreted as described in [RFC2119].

1.2 Normative References

[RFC2119] S. Bradner, *Key words for use in RFCs to Indicate Requirement Levels*, http://www.ietf.org/rfc/rfc2119.txt, IETF RFC 2119, March 1997.

[RFC4627] D. Crockford, *The application/json Media Type for JavaScript Object Notation (JSON)*, http://www.ietf.org/rfc/rfc4627.txt, IETF RFC 4627, July 2006.

[KMIP-SPEC-1_0] OASIS Standard, Key Management Interoperability Protocol Specification Version 1.0, October 2010, http://docs.oasis-open.org/kmip/spec/v1.0/os/kmip-spec-1.0-os.doc

[KMIP-SPEC-1_1] *Key Management Interoperability Protocol Specification Version 1.1.* http://docs.oasis-open.org/kmip/spec/v1.1/csd01/kmip-spec-v1.1-csd01.doc Committee Specification Draft 01.1 December 2011.

[KMIP-PROF] *Key Management Interoperability Protocol Usage Guide Version 1.1*. 01 December 2011. OASIS Standard. http://docs.oasis-open.org/kmip/profiles/v1.1/cd01/kmip-profiles-1.1-cd-01.doc

1.3 Non-Normative References

[KMIP-UG] *Key Management Interoperability Protocol Usage Guide Version 1.1.* http://docs.oasis-open.org/kmip/ug/v1.1/kmip-ug-v1.1-cnd01.doc Committee Note Draft, 1 December 2011,

[KMIP-TC] *Key Management Interoperability Protocol Test Cases Version 1.1.* http://docs.oasis-open.org/kmip/usecases/v1.1/kmip-usecases-v1.1-cnd01.doc, Committee Note Draft, 1 December 2011.

2 JSON Profile

The JSON profile is simply the use of KMIP replacing the TTLV message encoding with a JSON message encoding.

2.1 JSON Encoding

2.1.1 Hex representations

Hex representations of numbers must always begin with '0x' and must not include any spaces. They may use either upper or lower case 'a'-'f'. The hex representation must include all leading zeros or sign extension bits when representing a value of a fixed width such as Tags (3 bytes), Integer (32-bit signed big-endian), Long Integer (64-bit signed big-endian) and Big Integer (big-endian multiple of 8 bytes). The Integer values for -1, 0, 1 are represented as `"0xffffffff"`, `"0x00000000"`, `"0x00000001"`. Hex representation for Byte Strings are similar to numbers, but do not include the '0x' prefix, and can be of any length.

2.1.2 Tags

Tags are a String that may contain either:

- The 3-byte tag hex value prefixed with '0x'
- The normalised text of a Tag as specified in the KMIP Specification

Other text values may be used such as published names of Extension tags, or names of new tags added in future KMIP versions. Producers may however choose to use hex values for these tags to ensure they are understood by all consumers.

2.1.3 Normalizing Names

KMIP text values of Tags, Types and Enumerations SHALL be normalized to create a 'CamelCase' format that would be suitable to be used as a variable name in C/Java or an JSON name.

The basic approach to converting from KMIP text to CamelCase is to separate the text into individual word tokens (rules 1-4), capitalize the first letter of each word (rule 5) and then join with spaces removed (rule 6). The tokenizing splits on whitespace and on dashes where the token following is a valid word. The tokenizing also removes round brackets and shifts decimals from the front to the back of the first word in each string. The following rules SHALL be applied to create the normalized CamelCase form:

1. Replace round brackets '(', ')' with spaces
2. If a non-word char (not alpha, digit or underscore) is followed by a letter (either upper or lower case) then a lower case letter, replace the non-word char with space
3. Replace remaining non-word chars (except whitespace) with underscore.
4. If the first word begins with a digit, move all digits at start of first word to end of first word
5. Capitalize the first letter of each word
6. Concatenate all words with spaces removed

```
# 1. Replace brackets with space
noBrackets = re.sub('[()]', ' ', enumName)
# 2. replace \W with space if followed by letter, lower
nonWordToSpace = re.sub('\W([A-Za-z][a-z])', r' \1', noBrackets)
# 3. non-word to underscore
words = [re.sub('\W', '_', s) for s in nonWordToSpace.split()]
# 4. move numbers to end of first word
words[0] = re.sub('^(\d+)(.*)', r'\2\1', words[0])
# 5. captialize first letter of each word
words = [re.sub('^.', s[0].upper(), s) for s in words]
# 6. concatenate
enumNameCamel = ''.join(words)
```

Example python name normalization code

```
# 1. Replace brackets with space
$enumName=~s/[\(\)]/ /g;
# 2. replace \W with space if followed by letter, lower
$enumName=~s/\W([A-Za-z][a-z])/ \1/g;
# 3. non-word to underscore
@words=split(/ /,$enumName);
for($i=0;$i<=$#words;$i++) {  $words[$i]=~s/\W/_/g; }
# 4. move numbers to end of first word
$words[0] =~ s/^(\d+)(.*)/\2\1/;
# 5. captialize first letter of each word
for($i=0;$i<=$#words;$i++) {
  substr($words[$i],0,1)=~tr/a-z/A-Z/;
}
# 6. concatenate
$enumNameCamel  = join('',@words);
```

Example perl name normalization code

2.1.4 Type

Type must be a String containing one of the normalized CamelCase values as defined in the KMIP specification.

- Structure
- Integer
- LongInteger
- BigInteger
- Enumeration
- Boolean
- TextString
- ByteString
- DateTime
- Interval

If type is not included, the default type of Structure SHALL be used.

2.1.5 Value

The specification of a value is represented differently for each TTLV type.

2.1.6 JSON Object

For JSON encoding, each TTLV is represented as a JSON Object with properties 'tag', optional 'name', 'type' and 'value'.

```
{"tag": "ActivationDate", "type":"DateTime", "value":"2001-01-01T10:00:00+10:00"}
{"tag": "0x54FFFF", "name":"SomeExtension", "type":"Integer", "value":"0x00000001"}
```

The 'type' property / attribute SHALL have a default value of 'Structure' and may be omitted for Structures.

2.1.6.1 Tags

Tags are a String that may contain either:

- The 3-byte tag hex value prefixed with '0x'
- The normalised text of a Tag as specified in the KMIP Specification

Other text values may be used such as published names of Extension tags, or names of new tags added in future KMIP versions. Producers may however choose to use hex values for these tags to ensure they are understood by all consumers.

```
{"tag": "0x420001", "type":"DateTime", "value":"2001-01-01T10:00:00+10:00"}
{"tag": "ActivationDate", "type":"DateTime", "value":"2001-01-01T10:00:00+10:00"}
{"tag": "IVCounterNonce", "type":"ByteString", "value":"a1b2c3d4"}
{"tag": "PrivateKeyTemplateAttribute", "type":"Structure", "value":[]}
{"tag": "0x545352", "type":"TextString", "value":"This is an extension"}
{"tag": "WELL_KNOWN_EXTENSION", "type":"TextString", "value":"This is an extension"}
```

2.1.6.2 Structure

For JSON, value is an Array containing sub-items, or may be null.

```
{"tag": "ProtocolVersion", "type":"Structure", "value":[
  {"tag": "ProtocolVersionMajor", "type":"Integer", "value":1},
  {"tag": "ProtocolVersionMajor", "type":"Integer", "value":0}
]}
{"tag": "ProtocolVersion", "value":[
  {"tag": "ProtocolVersionMajor", "type":"Integer", "value":1},
  {"tag": "ProtocolVersionMajor", "type":"Integer", "value":0}
]}
```

The 'type' property / attribute is optional for a Structure.

2.1.6.3 Integer

For JSON, value is either a Number or a hex string.

```
{"tag": "BatchCount", "type":"Integer", "value":10}
{"tag": "BatchCount", "type":"Integer", "value":"0x0000000A"}
```

2.1.6.4 Integer - Special case for Masks

(Cryptographic Usage Mask, Storage Status Mask):

Integer mask values can also be encoded as a String containing mask components. JSON uses '|' as the separator. Components may be either the text of the enumeration value as defined in the KMIP Specification or a 32-bit unsigned big-endian hex string.

```
{"tag": "CryptographicUsageMask", "type":"Integer", "value": "0x0000100c"}
{"tag": "CryptographicUsageMask", "type":"Integer", "value": "Encrypt|Decrypt|CertificateSign"}
{"tag": "CryptographicUsageMask", "type":"Integer", "value":
"CertificateSign|0x00000004|0x0000008"}
{"tag": "CryptographicUsageMask", "type":"Integer", "value": "CertificateSign|0x0000000c"}
```

2.1.6.5 Long Integer

For JSON, value is either a Number or a hex string. Note that JS Numbers are 64-bit floating point and can only represent 53-bits of precision, so any values >= 2^52 must be represented as hex strings.

```
{"tag": "0x540001", "type":"LongInteger", "value":"0xfffffffffffffffe"}
{"tag": "0x540001", "type":"LongInteger", "value":-2}
{"tag": "UsageLimitsCount", "type":"LongInteger", "value":"0x1000000000000000"}
```

Note that this value (2^60) is too large to be represented as a Number in JSON.

2.1.6.6 Big Integer

For JSON, value is either a Number or a hex string. Note that Big Integers must be sign extended to contain a multiple of 8 bytes, and as per LongInteger, JS numbers only support a limited range of values.

```
{"tag": "X", "type":"BigInteger", "value":0}
{"tag": "X", "type":"BigInteger", "value":"0x0000000000000000"}
```

2.1.6.7 Enumeration

For JSON, value may contain:

- Number representing the enumeration 32-bit unsigned big-endian value
- Hex string representation of 32-bit unsigned big-endian value
- CamelCase enum text as defined in KMIP 9.1.3.2.x

```
{"tag": "0x420057", "type":"Enumeration", "value":2}
{"tag": "ObjectType", "type":"Enumeration", "value":"0x00000002"}
{"tag": "ObjectType", "type":"Enumeration", "value":"SymmetricKey"}
```

2.1.6.8 Boolean

For JSON, value must be either a hex string, or a JSON Boolean ‘true’ or ‘false’.

```
{"tag": "BatchOrderOption", "type":"Boolean", "value":true}
{"tag": "BatchOrderOption", "type":"Boolean", "value":"0x0000000000000001"}
```

2.1.6.9 Text String

For JSON, value must be a String

```
{"tag": "AttributeName", "type":"TextString", "value":"Cryptographic Algorithm"}
```

2.1.6.10 Byte String

For JSON, value must be a hex string. Note Byte Strings do not include the ‘0x’ prefix, and do not have any leading bytes.

```
{"tag": "MACSignature", "type":"ByteString", "value":"C50F77"}
```

2.1.6.11 Date-Time

For JSON, value must be either a hex string, or an ISO8601 DateTime as used in XSD using format:

```
'-'? yyyy '-' mm '-' dd 'T' hh ':' mm ':' ss ('.' s+)? ((('+' | '-') hh ':' mm) | 'Z')?
```

Fractional seconds are not used in KMIP and should not generally be shown. If they are used, they should be ignored (truncated).

```
{"tag": "ArchiveDate", "type":"DateTime", "value":"0x000000003a505520"}
{"tag": "ArchiveDate", "type":"DateTime", "value":"2001-01-01T10:00:00+10:00"}
```

2.1.6.12 Interval

For JSON, value is either a Number or a hex string. Note that intervals are 32-bit unsigned big-endian values.

```
{"tag": "Offset", "type":"Interval", "value":27}
{"tag": "Offset", "type":"Interval", "value":"0x0000001b"}
```

2.2 JSON Capable KMIP Client or Server

KMIP client and server implementations conformant to this profile SHALL:

1. Support JSON message encoding for request and response messages as specified in Section 2.1 of this profile.
2. Support mapping of all TTLV tags and enumerations specified within each version of the KMIP Specification that is supported ([KMIP-SPEC-1_0 and KMIP-SPEC-1_1])
3. Support user defined extensions containing additional tags and enumerations not specified within the KMIP Specification ([KMIP-SPEC-1_0 and KMIP-SPEC-1_1])

3 JSON Profile Test Cases

This section contains a test case that demonstrates the JSON profile encoding using test case 12.1 from [KMIP-TC] using protocol version 1.0 which exercises the Query operation and the Maximum Response Size header field.

3.1 Query, Maximum Response Size

3.1.1 Test Case: Query, Maximum Response Size

Perform a Query operation, querying the Operations and Objects supported by the server, with a restriction on the Maximum Response Size set in the request header. Since the resulting Query response is too big, an error is returned. Increase the Maximum Response Size, resubmit the Query request, and get a successful response.

Time	Request/Response messages

0	Query (operations, objects)
	In (header): maximumResponseSize='256'
	In: queryFunctions={ '00000001', '00000002' }
	
	Tag: REQUEST_MESSAGE (0x420078), Type: STRUCTURE (0x01), Data:
	  Tag: REQUEST_HEADER (0x420077), Type: STRUCTURE (0x01), Data:
	  Tag: PROTOCOL_VERSION (0x420069), Type: STRUCTURE (0x01), Data:
	  Tag: PROTOCOL_VERSION_MAJOR (0x42006a), Type: INTEGER (0x02), Data: 0x00000001
	  Tag: PROTOCOL_VERSION_MINOR (0x42006b), Type: INTEGER (0x02), Data: 0x00000000
	  Tag: MAXIMUM_RESPONSE_SIZE (0x420050), Type: INTEGER (0x02), Data: 0x00000100
	  Tag: BATCH_COUNT (0x42000d), Type: INTEGER (0x02), Data: 0x00000001
	  Tag: BATCH_ITEM (0x42000f), Type: STRUCTURE (0x01), Data:
	  Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000018 (QUERY)
	  Tag: REQUEST_PAYLOAD (0x420079), Type: STRUCTURE (0x01), Data:
	  Tag: QUERY_FUNCTION (0x420074), Type: ENUMERATION (0x05), Data: 0x00000001 (QUERY_OPERATIONS)
	  Tag: QUERY_FUNCTION (0x420074), Type: ENUMERATION (0x05), Data: 0x00000002 (QUERY_OBJECTS)
	
	42007801000000904200770100000048420069010000002042006a020000000400000001000000004200 6b020000000400000000000000004200500200000004000001000000000042000d020000000400000001000000042000f0100000038420 05c05000000040000001800000000420079010000002042007405000000040000001000000004200740500000004000000010000000042 00740500000004000000020000000042007405000000040000000200000000
	
	{"tag":"RequestMessage", "value":[
	  {"tag":"RequestHeader", "value":[
	  {"tag":"ProtocolVersion", "value":[
	  {"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"},
	  {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000000"}
	  ]},
	  {"tag":"MaximumResponseSize", "type":"Integer", "value":"0x00000100"},
	  {"tag":"BatchCount", "type":"Integer", "value":"0x00000001"}
	  ]},
	  {"tag":"BatchItem", "value":[
	  {"tag":"Operation", "type":"Enumeration", "value":"Query"},
	  {"tag":"RequestPayload", "value":[
	  {"tag":"QueryFunction", "type":"Enumeration", "value":"QueryOperations"},
	  {"tag":"QueryFunction", "type":"Enumeration", "value":"QueryObjects"}
	  ]}
	  ]}
	]}
	
	Out: Operation Failed, Response Too Large

```
Tag: RESPONSE_MESSAGE (0x42007b), Type: STRUCTURE (0x01), Data:
  Tag: RESPONSE_HEADER (0x42007a), Type: STRUCTURE (0x01), Data:
    Tag: PROTOCOL_VERSION (0x420069), Type: STRUCTURE (0x01), Data:
      Tag: PROTOCOL_VERSION_MAJOR (0x42006a), Type: INTEGER (0x02), Data:
0x00000001
      Tag: PROTOCOL_VERSION_MINOR (0x42006b), Type: INTEGER (0x02), Data:
0x00000000
    Tag: TIME_STAMP (0x420092), Type: DATE_TIME (0x09), Data: 0x000000004feafb9f
Wed Jun 27 22:25:03 2012
    Tag: BATCH_COUNT (0x42000d), Type: INTEGER (0x02), Data: 0x00000001
  Tag: BATCH_ITEM (0x42000f), Type: STRUCTURE (0x01), Data:
    Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000018 (QUERY)
    Tag: RESULT_STATUS (0x42007f), Type: ENUMERATION (0x05), Data: 0x00000001
(OPERATION_FAILED)
    Tag: RESULT_REASON (0x42007e), Type: ENUMERATION (0x05), Data: 0x00000002
(RESPONSE_TOO_LARGE)
    Tag: RESULT_MESSAGE (0x42007d), Type: TEXT_STRING (0x07), Data: TOO_LARGE

42007B01000000C842007A0100000048420069010000002042006A020000000400000001000000004200
6B020000000400000001000000004200920900000008000000004F9A556B42000D020000000400000
0001000000042000F0100000070420070500000004000000010000000042007E0500000004000000
020000000042007D07000000434526573706F6E73652073697A653A203634382C204D6178696D756D20
526573706F6E73652053697A6520696E6469636174656420696E207265717565737743A203235360000
000000

{"tag":"ResponseMessage", "value":[
  {"tag":"ResponseHeader", "value":[
    {"tag":"ProtocolVersion", "value":[
      {"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"},
      {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000000"}
    ]},
    {"tag":"TimeStamp", "type":"DateTime", "value":"2010-02-15T19:49:30+10:00"},
    {"tag":"BatchCount", "type":"Integer", "value":"0x00000001"}
  ]},
  {"tag":"BatchItem", "value":[
    {"tag":"ResultStatus", "type":"Enumeration", "value":"OperationFailed"},
    {"tag":"ResultReason", "type":"Enumeration", "value":"ResponseTooLarge"},
    {"tag":"ResultMessage", "type":"TextString", "value":"Response size: 568,
Maximum Response Size indicated in request: 256"}
  ]}
]}
```

1

Query (operations, objects)
In (header): maximumResponseSize=’2048’
In: queryFunctions={ ‘00000001’, ‘00000002’}

```
Tag: REQUEST_MESSAGE (0x420078), Type: STRUCTURE (0x01), Data:
  Tag: REQUEST_HEADER (0x420077), Type: STRUCTURE (0x01), Data:
    Tag: PROTOCOL_VERSION (0x420069), Type: STRUCTURE (0x01), Data:
      Tag: PROTOCOL_VERSION_MAJOR (0x42006a), Type: INTEGER (0x02), Data:
0x00000001
      Tag: PROTOCOL_VERSION_MINOR (0x42006b), Type: INTEGER (0x02), Data:
0x00000000
    Tag: MAXIMUM_RESPONSE_SIZE (0x420050), Type: INTEGER (0x02), Data: 0x00000800
    Tag: BATCH_COUNT (0x42000d), Type: INTEGER (0x02), Data: 0x00000001
  Tag: BATCH_ITEM (0x42000f), Type: STRUCTURE (0x01), Data:
    Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000018 (QUERY)
    Tag: REQUEST_PAYLOAD (0x420079), Type: STRUCTURE (0x01), Data:
      Tag: QUERY_FUNCTION (0x420074), Type: ENUMERATION (0x05), Data: 0x00000001
(QUERY_OPERATIONS)
      Tag: QUERY_FUNCTION (0x420074), Type: ENUMERATION (0x05), Data: 0x00000002
(QUERY_OBJECTS)

42007801000000904200770100000048420069010000002042006a020000000400000001000000004
42006b020000000400000000000000004200500200000004000008000000000042000d0200000004
000000010000000042000f0100000038420005c050000000400000018000000004200790100000020
42007405000000040000000100000000420074050000000400000002000000000

{"tag":"RequestMessage", "value":[
```

```
  {"tag":"RequestHeader", "value":[
    {"tag":"ProtocolVersion", "value":[
      {"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"},
      {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000000"}
    ]},
    {"tag":"MaximumResponseSize", "type":"Integer", "value":"0x00000800"},
    {"tag":"BatchCount", "type":"Integer", "value":"0x00000001"}
  ]},
  {"tag":"BatchItem", "value":[
    {"tag":"Operation", "type":"Enumeration", "value":"Query"},
    {"tag":"RequestPayload", "value":[
      {"tag":"QueryFunction", "type":"Enumeration", "value":"QueryOperations"},
      {"tag":"QueryFunction", "type":"Enumeration", "value":"QueryObjects"}
    ]}
  ]}
]}
```

Out: operations, objects, serverInformation

```
Tag: RESPONSE_MESSAGE (0x42007b), Type: STRUCTURE (0x01), Data:
  Tag: RESPONSE_HEADER (0x42007a), Type: STRUCTURE (0x01), Data:
    Tag: PROTOCOL_VERSION (0x420069), Type: STRUCTURE (0x01), Data:
      Tag: PROTOCOL_VERSION_MAJOR (0x42006a), Type: INTEGER (0x02), Data:
0x00000001
      Tag: PROTOCOL_VERSION_MINOR (0x42006b), Type: INTEGER (0x02), Data:
0x00000000
    Tag: TIME_STAMP (0x420092), Type: DATE_TIME (0x09), Data: 0x000000004feafb9f
Wed Jun 27 22:25:03 2012
    Tag: BATCH_COUNT (0x42000d), Type: INTEGER (0x02), Data: 0x00000001
  Tag: BATCH_ITEM (0x42000f), Type: STRUCTURE (0x01), Data:
    Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000018 (QUERY)
    Tag: RESULT_STATUS (0x42007f), Type: ENUMERATION (0x05), Data: 0x00000000
(SUCCESS)
    Tag: RESPONSE_PAYLOAD (0x42007c), Type: STRUCTURE (0x01), Data:
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000018
(QUERY)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000008
(LOCATE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000014
(DESTROY)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x0000000a (GET)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000001
(CREATE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000003
(REGISTER)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x0000000b
(GET_ATTRIBUTES)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x0000000c
(GET_ATTRIBUTE_LIST)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x0000000d
(ADD_ATTRIBUTE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x0000000e
(MODIFY_ATTRIBUTE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x0000000f
(DELETE_ATTRIBUTE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000012
(ACTIVATE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000013
(REVOKE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x0000001a (POLL)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000019
(CANCEL)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000009
(CHECK)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000011
(GET_USAGE_ALLOCATION)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000002
(CREATE_KEY_PAIR)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000004
(RE_KEY)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000015
```

```
(ARCHIVE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000016
(RECOVER)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000010
(OBTAIN_LEASE)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x0000001d
(RE_KEY_KEY_PAIR)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000006
(CERTIFY)
      Tag: OPERATION (0x42005c), Type: ENUMERATION (0x05), Data: 0x00000007
(RE_CERTIFY)
      Tag: OBJECT_TYPE (0x420057), Type: ENUMERATION (0x05), Data: 0x00000001
(CERTIFICATE)
      Tag: OBJECT_TYPE (0x420057), Type: ENUMERATION (0x05), Data: 0x00000002
(SYMMETRIC_KEY)
      Tag: OBJECT_TYPE (0x420057), Type: ENUMERATION (0x05), Data: 0x00000007
(SECRET_DATA)
      Tag: OBJECT_TYPE (0x420057), Type: ENUMERATION (0x05), Data: 0x00000003
(PUBLIC_KEY)
      Tag: OBJECT_TYPE (0x420057), Type: ENUMERATION (0x05), Data: 0x00000004
(PRIVATE_KEY)
      Tag: OBJECT_TYPE (0x420057), Type: ENUMERATION (0x05), Data: 0x00000006
(TEMPLATE)
      Tag: OBJECT_TYPE (0x420057), Type: ENUMERATION (0x05), Data: 0x00000008
(OPAQUE_OBJECT)
      Tag: OBJECT_TYPE (0x420057), Type: ENUMERATION (0x05), Data: 0x00000005
(SPLIT_KEY)

42007b010000029042007a01000000484200690100000020420006a020000000400000001000000000
42006b020000000400000000000000004200920900000008000000004feafb9f42000d0200000004
0000000100000000042000f010000023842005c050000000400000018000000004200 7f0500000004
0000000000000000042007c010000021042005c050000000400000018000000004200 5c0500000004
0000000800000000042005c050000000400000014000000004200 5c050000004000000a00000000
42005c050000000400000001000000004200 5c050000000400000030000000042005c0500000004
0000000b0000000042005c050000000400000000c000000042005c050000000400000000d00000000
42005c050000000400000000e000000042005c050000000400000000f000000042005c0500000004
0000001200000000042005c050000000400000013000000004200 5c050000000400000001a00000000
42005c050000000400000019000000004200 5c050000000400000009000000004200 5c0500000004
0000001100000000042005c050000000400000002000000004200 5c050000000400000004000000000
42005c050000000400000015000000004200 5c050000000400000016000000004200 5c0500000004
0000001000000000042005c050000000400000001d000000004200 5c050000000400000006000000000
42005c050000000400000007000000004200570500000000400000001000000004200570500000004
0000000200000000420057050000000400000007000000004200570500000000400000003000000000
42005705000000040000000400000000420057050000000400000006000000004200570500000004
0000000800000000420057050000000400000000500000000

{"tag":"ResponseMessage", "value":[
  {"tag":"ResponseHeader", "value":[
    {"tag":"ProtocolVersion", "value":[
      {"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"},
      {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000000"}
    ]},
    {"tag":"TimeStamp", "type":"DateTime", "value":"2010-02-15T19:49:30+10:00"},
    {"tag":"BatchCount", "type":"Integer", "value":"0x00000001"}
  ]},
  {"tag":"BatchItem", "value":[
    {"tag":"Operation", "type":"Enumeration", "value":"Query"},
    {"tag":"ResultStatus", "type":"Enumeration", "value":"Success"},
    {"tag":"ResponsePayload", "value":[
      {"tag":"Operation", "type":"Enumeration", "value":"Create"},
      {"tag":"Operation", "type":"Enumeration", "value":"CreateKeyPair"},
      {"tag":"Operation", "type":"Enumeration", "value":"Register"},
      {"tag":"Operation", "type":"Enumeration", "value":"ReKey"},
      {"tag":"Operation", "type":"Enumeration", "value":"Locate"},
      {"tag":"Operation", "type":"Enumeration", "value":"Check"},
      {"tag":"Operation", "type":"Enumeration", "value":"Get"},
      {"tag":"Operation", "type":"Enumeration", "value":"GetAttributes"},
      {"tag":"Operation", "type":"Enumeration", "value":"GetAttributeList"},
      {"tag":"Operation", "type":"Enumeration", "value":"AddAttribute"},
      {"tag":"Operation", "type":"Enumeration", "value":"ModifyAttribute"},
      {"tag":"Operation", "type":"Enumeration", "value":"DeleteAttribute"},
      {"tag":"Operation", "type":"Enumeration", "value":"ObtainLease"},
```

```
      {"tag":"Operation", "type":"Enumeration", "value":"GetUsageAllocation"},
      {"tag":"Operation", "type":"Enumeration", "value":"Activate"},
      {"tag":"Operation", "type":"Enumeration", "value":"Revoke"},
      {"tag":"Operation", "type":"Enumeration", "value":"Destroy"},
      {"tag":"Operation", "type":"Enumeration", "value":"Archive"},
      {"tag":"Operation", "type":"Enumeration", "value":"Recover"},
      {"tag":"Operation", "type":"Enumeration", "value":"Query"},
      {"tag":"Operation", "type":"Enumeration", "value":"Cancel"},
      {"tag":"Operation", "type":"Enumeration", "value":"Poll"},
      {"tag":"ObjectType", "type":"Enumeration", "value":"Certificate"},
      {"tag":"ObjectType", "type":"Enumeration", "value":"SymmetricKey"},
      {"tag":"ObjectType", "type":"Enumeration", "value":"PublicKey"},
      {"tag":"ObjectType", "type":"Enumeration", "value":"PrivateKey"},
      {"tag":"ObjectType", "type":"Enumeration", "value":"Template"}
    ]}
  ]}
]}
```

Appendix A. Acknowledgments

The following individuals have participated in the creation of this specification and are gratefully acknowledged:

Bob Griffin, EMC.

