

KMIP Additional Message Encodings Version 1.0

OASIS Standard

19 May 2015

Specification URIs

This version:

<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/os/kmip-addtl-msg-enc-v1.0-os.doc>
(Authoritative)
<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/os/kmip-addtl-msg-enc-v1.0-os.html>
<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/os/kmip-addtl-msg-enc-v1.0-os.pdf>

Previous version:

<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/csprd01/kmip-addtl-msg-enc-v1.0-csprd01.doc> (Authoritative)
<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/csprd01/kmip-addtl-msg-enc-v1.0-csprd01.html>
<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/csprd01/kmip-addtl-msg-enc-v1.0-csprd01.pdf>

Latest version:

<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/kmip-addtl-msg-enc-v1.0.doc>
(Authoritative)
<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/kmip-addtl-msg-enc-v1.0.html>
<http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/kmip-addtl-msg-enc-v1.0.pdf>

Technical Committee:

OASIS Key Management Interoperability Protocol (KMIP) TC

Chairs:

Saikat Saha (saikat.saha@oracle.com), Oracle
Tony Cox (tjc@cryptsoft.com), Cryptsoft Pty Ltd.

Editor:

Tim Hudson (tjh@cryptsoft.com), Cryptsoft Pty Ltd.

Related work:

This specification is related to:

- *Key Management Interoperability Protocol Profiles Version 1.0*. Edited by Robert Griffin and Subhash Sankuratipati. Latest version: <http://docs.oasis-open.org/kmip/profiles/v1.0/kmip-profiles-1.0.html>.
- *Key Management Interoperability Protocol Profiles Version 1.1*. Edited by Robert Griffin and Subhash Sankuratipati. Latest version: <http://docs.oasis-open.org/kmip/profiles/v1.1/kmip-profiles-v1.1.html>.
- *Key Management Interoperability Protocol Profiles Version 1.2*. Edited by Tim Hudson and Robert Lockhart. Latest version: <http://docs.oasis-open.org/kmip/profiles/v1.2/kmip-profiles-v1.2.html>.

- *Key Management Interoperability Protocol Specification Version 1.1*. Edited by Robert Haas and Indra Fitzgerald. Latest version: <http://docs.oasis-open.org/kmip/spec/v1.1/kmip-spec-v1.1.html>.
- *Key Management Interoperability Protocol Specification Version 1.2*. Edited by Kiran Thota and Kelley Burgin. Latest version: <http://docs.oasis-open.org/kmip/spec/v1.2/kmip-spec-v1.2.html>.
- *Key Management Interoperability Protocol Test Cases Version 1.2*. Edited by Tim Hudson and Faisal Faruqi. Latest version: <http://docs.oasis-open.org/kmip/testcases/v1.2/kmip-testcases-v1.2.html>.
- *Key Management Interoperability Protocol Usage Guide Version 1.2*. Edited by Indra Fitzgerald and Judith Furlong. Latest version: <http://docs.oasis-open.org/kmip/ug/v1.2/kmip-ug-v1.2.html>.

Abstract:

Describes additional (optional) message encodings as an alternative to the (mandatory) raw TTLV (Tag, Type, Length, Value) encoding including HTTPS, JSON and XML.

Status:

This document was last revised or approved by the membership of OASIS on the above date. The level of approval is also listed above. Check the “Latest version” location noted above for possible later revisions of this document. Any other numbered Versions and other technical work produced by the Technical Committee (TC) are listed at https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=kmip#technical.

Technical Committee members should send comments on this specification to the Technical Committee’s email list. Others should send comments to the Technical Committee by using the “Send A Comment” button on the Technical Committee’s web page at <https://www.oasis-open.org/committees/kmip/>.

For information on whether any patents have been disclosed that may be essential to implementing this specification, and any offers of patent licensing terms, please refer to the Intellectual Property Rights section of the Technical Committee web page (<https://www.oasis-open.org/committees/kmip/ipr.php>).

Citation format:

When referencing this specification the following citation format should be used:

[kmip-addtl-msg-enc-v1.0]

KMIP Additional Message Encodings Version 1.0. Edited by Tim Hudson. 19 May 2015. OASIS Standard. <http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/os/kmip-addtl-msg-enc-v1.0-os.html>. Latest version: <http://docs.oasis-open.org/kmip/kmip-addtl-msg-enc/v1.0/kmip-addtl-msg-enc-v1.0.html>.

Notices

Copyright © OASIS Open 2015. All Rights Reserved.

All capitalized terms in the following text have the meanings assigned to them in the OASIS Intellectual Property Rights Policy (the "OASIS IPR Policy"). The full [Policy](#) may be found at the OASIS website.

This document and translations of it may be copied and furnished to others, and derivative works that comment on or otherwise explain it or assist in its implementation may be prepared, copied, published, and distributed, in whole or in part, without restriction of any kind, provided that the above copyright notice and this section are included on all such copies and derivative works. However, this document itself may not be modified in any way, including by removing the copyright notice or references to OASIS, except as needed for the purpose of developing any document or deliverable produced by an OASIS Technical Committee (in which case the rules applicable to copyrights, as set forth in the OASIS IPR Policy, must be followed) or as required to translate it into languages other than English.

The limited permissions granted above are perpetual and will not be revoked by OASIS or its successors or assigns.

This document and the information contained herein is provided on an "AS IS" basis and OASIS DISCLAIMS ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY OWNERSHIP RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

OASIS requests that any OASIS Party or any other party that believes it has patent claims that would necessarily be infringed by implementations of this OASIS Committee Specification or OASIS Standard, to notify OASIS TC Administrator and provide an indication of its willingness to grant patent licenses to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification.

OASIS invites any party to contact the OASIS TC Administrator if it is aware of a claim of ownership of any patent claims that would necessarily be infringed by implementations of this specification by a patent holder that is not willing to provide a license to such patent claims in a manner consistent with the IPR Mode of the OASIS Technical Committee that produced this specification. OASIS may include such claims on its website, but disclaims any obligation to do so.

OASIS takes no position regarding the validity or scope of any intellectual property or other rights that might be claimed to pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; neither does it represent that it has made any effort to identify any such rights. Information on OASIS' procedures with respect to rights in any document or deliverable produced by an OASIS Technical Committee can be found on the OASIS website. Copies of claims of rights made available for publication and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this OASIS Committee Specification or OASIS Standard, can be obtained from the OASIS TC Administrator. OASIS makes no representation that any information or list of intellectual property rights will at any time be complete, or that any claims in such list are, in fact, Essential Claims.

The name "OASIS" is a trademark of [OASIS](#), the owner and developer of this specification, and should be used only to refer to the organization and its official outputs. OASIS welcomes reference to, and implementation and use of, specifications, while reserving the right to enforce its marks against misleading uses. Please see <https://www.oasis-open.org/policies-guidelines/trademark> for above guidance.

Table of Contents

1	Introduction	6
1.1	Terminology	6
1.2	Normative References	6
1.3	Non-Normative References	7
2	HTTPS Profile.....	8
2.1	Authentication Suite.....	8
2.2	KMIP Port Number.....	8
2.3	Request URI	8
2.4	HTTP Encoding - Client	8
2.5	HTTP Encoding - Server.....	8
3	HTTPS Profile Test Cases	10
3.1	Mandatory HTTPS Profile Test Cases KMIP v1.0.....	10
3.1.1	MSGENC-HTTPS-M-1-10 - Query, Maximum Response Size	10
3.2	Mandatory HTTPS Profile Test Cases KMIP v1.1	14
3.2.1	MSGENC-HTTPS-M-1-11 - Query, Maximum Response Size	14
3.3	Mandatory HTTPS Profile Test Cases KMIP v1.2.....	18
3.3.1	MSGENC-HTTPS-M-1-12 - Query, Maximum Response Size	18
4	JSON Profile.....	23
4.1	JSON Encoding	23
4.1.1	Hex representations	23
4.1.2	Tags.....	23
4.1.3	Normalizing Names	23
4.1.4	Type.....	24
4.1.5	Value	24
4.1.6	JSON Object.....	25
5	JSON Profile Test Cases	27
5.1	Mandatory JSON Profile Test Cases KMIP v1.0.....	27
5.1.1	MSGENC-JSON-M-1-10 - Query, Maximum Response Size	27
5.2	Mandatory JSON Profile Test Cases KMIP v1.1	31
5.2.1	MSGENC-JSON-M-1-11 - Query, Maximum Response Size	31
5.3	Mandatory JSON Profile Test Cases KMIP v1.2.....	35
5.3.1	MSGENC-JSON-M-1-12 - Query, Maximum Response Size	35
6	XML Profile	40
6.1	XML Encoding	40
6.1.1	Hex representations	40
6.1.2	Tags.....	40
6.1.3	Normalizing Names	40
6.1.4	Type.....	41
6.1.5	Value	41
6.1.6	XML Element Encoding.....	42
7	XML Profile Test Cases.....	44
7.1	Mandatory XML Profile Test Cases KMIP v1.0	44
7.1.1	MSGENC-XML-M-1-10 - Query, Maximum Response Size	44

7.2 Mandatory XML Profile Test Cases KMIP v1.1	46
7.2.1 MSGENC-XML-M-1-11 - Query, Maximum Response Size	46
7.3 Mandatory XML Profile Test Cases KMIP v1.2	49
7.3.1 MSGENC-XML-M-1-12 - Query, Maximum Response Size	49
8 Conformance	53
8.1 HTTPS Profile	53
8.1.1 HTTPS Client KMIP v1.0 Profile Conformance	53
8.1.2 HTTPS Client KMIP v1.1 Profile Conformance	53
8.1.3 HTTPS Client KMIP v1.2 Profile Conformance	53
8.1.4 HTTPS Server KMIP v1.0 Profile Conformance	53
8.1.5 HTTPS Server KMIP v1.1 Profile Conformance	53
8.1.6 HTTPS Server KMIP v1.2 Profile Conformance	54
8.2 JSON Profile	54
8.2.1 JSON Client KMIP v1.0 Profile Conformance	54
8.2.2 JSON Client KMIP v1.1 Profile Conformance	54
8.2.3 JSON Client KMIP v1.2 Profile Conformance	54
8.2.4 JSON Server KMIP v1.0 Profile Conformance	54
8.2.5 JSON Server KMIP v1.1 Profile Conformance	55
8.2.6 JSON Server KMIP v1.2 Profile Conformance	55
8.3 XML Profile	55
8.3.1 XML Client KMIP v1.0 Profile Conformance	55
8.3.2 XML Client KMIP v1.1 Profile Conformance	55
8.3.3 XML Client KMIP v1.2 Profile Conformance	56
8.3.4 XML Server KMIP v1.0 Profile Conformance	56
8.3.5 XML Server KMIP v1.1 Profile Conformance	56
8.3.6 XML Server KMIP v1.2 Profile Conformance	56
8.4 Permitted Test Case Variations	56
8.4.1 Variable Items	57
8.4.2 Variable behavior	58
Appendix A. Acknowledgments	59
Appendix B. KMIP Specification Cross Reference	62
Appendix C. Revision History	67

1 Introduction

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

This profile defines the necessary encoding rules for the transport of KMIP TTLV messages encoded in:

- [Hypertext Transfer Protocol](#) [RFC2616] over [TLS](#) as specified in [HTTP over TLS](#) [RFC2818]
- [JavaScript Object Notification](#) [RFC4627]
- [Extensible Markup Language](#) [XML]

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.
- [RFC2616] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, L. Masinter, P. Leach, T. Berners-Lee, *Hypertext Transfer Protocol -- HTTP/1.1*, <http://www.ietf.org/rfc/rfc2616.txt>, IETF RFC 2616, June 1999.
- [RFC2818] E. Rescorla, *HTTP over TLS*, IETF RFC 2818, May 2000, <http://www.ietf.org/rfc/rfc2818.txt>
- [RFC7159] Bray, T., Ed., *The JavaScript Object Notation (JSON) Data Interchange Format*, RFC 7159, March 2014. <http://www.ietf.org/rfc/rfc7159.txt>
- [XML] Bray, Tim, et.al. eds, *Extensible Markup Language (XML) 1.0 (Fifth Edition)*, W3C Recommendation 26 November 2008, available at <http://www.w3.org/TR/2008/REC-xml-20081126/>
- [KMIP-SPEC] One or more of [KMIP-SPEC-1_0], [KMIP-SPEC-1_1], [KMIP-SPEC-1_2]
- [KMIP-SPEC-1_0] *Key Management Interoperability Protocol Specification Version 1.0* <http://docs.oasis-open.org/kmip/spec/v1.0/os/kmip-spec-1.0-os.doc>
OASIS Standard, October 2010.
- [KMIP-SPEC-1_1] *Key Management Interoperability Protocol Specification Version 1.1*. <http://docs.oasis-open.org/kmip/spec/v1.1/os/kmip-spec-v1.1-os.doc>
OASIS Standard. 24 January 2013.
- [KMIP-SPEC-1_2] *Key Management Interoperability Protocol Specification Version 1.2*. Edited by Kiran Thota and Kelley Burgin. Latest version: <http://docs.oasis-open.org/kmip/spec/v1.2/kmip-spec-v1.2.doc>.
- [KMIP-PROF] One or more of [KMIP-PROF-1_0], [KMIP-PROF-1_1], [KMIP-PROF-1_2]
- [KMIP-PROF-1_0] *Key Management Interoperability Protocol Profiles Version 1.0*. <http://docs.oasis-open.org/kmip/profiles/v1.0/os/kmip-profiles-1.0-os.doc>
OASIS Standard. 1 October 2010.
- [KMIP-PROF-1_1] *Key Management Interoperability Protocol Profiles Version 1.1*. <http://docs.oasis-open.org/kmip/profiles/v1.1/os/kmip-profiles-v1.1-os.doc>
OASIS Standard 01. 24 January 2013.
- [KMIP-PROF-1_2] *Key Management Interoperability Protocol Profiles Version 1.2*. Edited by Tim Hudson and Robert Lockhart. Latest version: <http://docs.oasis-open.org/kmip/profiles/v1.2/kmip-profiles-v1.2.doc>.

45 **1.3 Non-Normative References**

46 **[XML-SCHEMA]** Paul V. Biron, Ashok Malhotra, XML Schema Part 2: Datatypes Second Edition,
47 W3C Recommendation 26 November 2008, available at
48 <http://www.w3.org/TR/2004/REC-xmlschema-2-20041028/>

49

50

51 2 HTTPS Profile

52 The Hypertext Transfer Protocol over Transport Layer Security (HTTPS) is simply the use of HTTP over
53 TLS in the same manner that HTTP is used over TCP.

54 KMIP over HTTPS is simply the use of KMIP messages over HTTPS in the same manner that KMIP is
55 used over TLS.

56 2.1 Authentication Suite

57 Implementations conformant to this profile SHALL support one or more of the Authentication Suites
58 defined within section 3 of [KMIP-PROF].

59 2.2 KMIP Port Number

60 KMIP servers conformant to this profile MAY use TCP port number 5696, as assigned by IANA, to receive
61 and send KMIP messages provided that both HTTP and non-HTTP encoded messages are supported.

62 KMIP clients SHALL enable end user configuration of the TCP port number used, as a KMIP server may
63 specify a different TCP port number.

64 2.3 Request URI

65 KMIP servers conformant to this profile SHOULD support the value */kmip* as the target URI.

66 KMIP clients SHALL enable end user configuration of the target URI used as a KMIP server may specify
67 a different target URI.

68 2.4 HTTP Encoding - Client

69 KMIP client implementations conformant to this profile:

- 70 1. SHALL support HTTP/1.0 and/or HTTP/1.1 over TLS conformant to [RFC2818]
- 71 2. SHALL use the POST request method
- 72 3. SHALL specify a Content-Type of "application/octet-stream" if the message encoding is TTLV
- 73 4. SHALL specify a Content-Type of "text/xml" if the message encoding is XML
- 74 5. SHALL specify a Content-Type of "application/json" if the message encoding is JSON
- 75 6. SHALL specify a Content-Length
- 76 7. SHALL specify a Cache-Control of "no-cache"
- 77 8. SHALL send KMIP TTLV message in binary format as the body of the HTTP request

78

79 KMIP clients that support responding to server to client operations SHALL behave as a HTTPS server.

80 2.5 HTTP Encoding - Server

81 KMIP server implementations conformant to this profile:

- 82 1. SHALL support HTTP/1.0 and HTTP/1.1 over TLS conformant to [RFC2818]
- 83 2. SHALL return HTTP response code 200 if a KMIP response is available
- 84 3. SHALL specify a Content-Type of "application/octet-stream" if the message encoding is TTLV
- 85 4. SHALL specify a Content-Type of "text/xml" if the message encoding is XML
- 86 5. SHALL specify a Content-Type of "application/json" if the message encoding is JSON
- 87 6. SHALL specify a Content-Length

- 88 7. SHALL specify a Cache-Control of “no-cache”
89 8. SHALL send KMIP TTLV message in binary format as the body of the HTTP request
90
91 KMIP servers that support server to client operations SHALL behave as a HTTPS client.
92

93 3 HTTPS Profile Test Cases

94 The test cases define a number of request-response pairs for KMIP operations. Each test case is
95 provided in the XML format specified in section 6 intended to be both human-readable and usable by
96 automated tools. The time sequence (starting from 0) for each request-response pair is noted and line
97 numbers are provided for ease of cross-reference for a given test sequence.

98 Each test case has a unique label (the section name) which includes indication of mandatory (-M-) or
99 optional (-O-) status and the protocol version major and minor numbers as part of the identifier.

100 The test cases may depend on a specific configuration of a KMIP client and server being configured in a
101 manner consistent with the test case assumptions.

102 Where possible the flow of unique identifiers between tests, the date-time values, and other dynamic
103 items are indicated using symbolic identifiers – in actual request and response messages these dynamic
104 values will be filled in with valid values.

105 Note: the values for the returned items and the custom attributes are illustrative. Actual values from a real
106 client system may vary as specified in section 8.4

107 3.1 Mandatory HTTPS Profile Test Cases KMIP v1.0

108 3.1.1 MSGENC-HTTPS-M-1-10 - Query, Maximum Response Size

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

113 The specific list of operations and object types returned in the response MAY vary.

```
0001 # TIME 0
0002 <RequestMessage>
0003   <RequestHeader>
0004     <ProtocolVersion>
0005       <ProtocolVersionMajor type="Integer" value="1"/>
0006       <ProtocolVersionMinor type="Integer" value="0"/>
0007     </ProtocolVersion>
0008     <MaximumResponseSize type="Integer" value="256"/>
0009     <BatchCount type="Integer" value="1"/>
0010   </RequestHeader>
0011   <BatchItem>
0012     <Operation type="Enumeration" value="Query"/>
0013     <RequestPayload>
0014       <QueryFunction type="Enumeration" value="QueryOperations"/>
0015       <QueryFunction type="Enumeration" value="QueryObjects"/>
0016     </RequestPayload>
0017   </BatchItem>
</RequestMessage>

42007801000000904200770100000048420069010000002042006a02000000040000000100000000
42006b0200000004000000000000000042005002000000040000001000000000042000d0200000004
000000010000000042000f010000003842005c050000000400000018000000004200790100000020
4200740500000004000000010000000042007405000000040000000200000000

00000000: 50 4f 53 54 20 2f 6b 6d-69 70 20 48 54 54 50 2f POST /kmip HTTP/
00000010: 31 2e 30 0d 0a 50 72 61-67 6d 61 3a 20 6e 6f 2d 1.0..Pragma: no-
00000020: 63 61 63 68 65 0d 0a 43-61 63 68 65 2d 43 6f 6e cache..Cache-Con
00000030: 74 72 6f 6c 3a 20 6e 6f-2d 63 61 63 68 65 0d 0a trol: no-cache..
00000040: 43 6f 6e 6e 65 63 74 69-6f 6e 3a 20 6b 65 65 70 Connection: keep
00000050: 2d 61 6c 69 76 65 0d 0a-43 6f 6e 74 65 6e 74 2d -alive..Content-
```

	<pre> 00000060: 54 79 70 65 3a 20 61 70-70 6c 69 63 61 74 69 6f Type: applicatio 00000070: 6e 2f 6f 63 74 65 74 2d-73 74 72 65 61 6d 0d 0a n/octet-stream.. 00000080: 43 6f 6e 74 65 6e 74 2d-4c 65 6e 67 74 68 3a 20 Content-Length: 00000090: 31 35 32 20 20 20 20 20-20 20 0d 0a 0d 0a 42 00 152 B. 000000a0: 15 32 78 01 00 00 00 90-42 00 77 01 00 00 00 48 .2x.....B.w....H 000000b0: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j..... 000000c0: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04 B.k..... 000000d0: 00 00 00 00 00 00 00 00-42 00 50 02 00 00 00 04 B.P..... 000000e0: 00 00 01 00 00 00 00 00-42 00 0d 02 00 00 00 04 B..... 000000f0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 38 B.....8 00000100: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 B.\..... 00000110: 42 00 79 01 00 00 00 20-42 00 74 05 00 00 00 04 B.y.... B.t.... 00000120: 00 00 00 01 00 00 00 00-42 00 74 05 00 00 00 04 B.t..... 00000130: 00 00 00 02 00 00 00 00- </pre>
<pre> 0018 <ResponseMessage> 0019 <ResponseHeader> 0020 <ProtocolVersion> 0021 <ProtocolVersionMajor type="Integer" value="1"/> 0022 <ProtocolVersionMinor type="Integer" value="0"/> 0023 </ProtocolVersion> 0024 <TimeStamp type="DateTime" value="2013-06-26T09:09:17+00:00"/> 0025 <BatchCount type="Integer" value="1"/> 0026 </ResponseHeader> <BatchItem> 0027 <Operation type="Enumeration" value="Query"/> 0028 <ResultStatus type="Enumeration" value="OperationFailed"/> 0029 <ResultReason type="Enumeration" value="ResponseTooLarge"/> 0030 <ResultMessage type="TextString" value="TOO_LARGE"/> 0031 </BatchItem> 0032 </ResponseMessage> </pre> <pre> 42007b01000000a042007a0100000048420069010000002042006a02000000040000000100000000 42006b0200000004000000000000000042009209000000080000000051caafbd42000d0200000004 000000010000000042000f010000004842005c0500000004000000180000000042007f0500000004 000000010000000042007e0500000004000000020000000042007d0700000009544f4f5f4c415247 4500000000000000 </pre> <pre> 00000000: 48 54 54 50 2f 31 2e 31-20 32 30 30 20 4f 4b 0d HTTP/1.1 200 OK. 00000010: 0a 43 6f 6e 74 65 6e 74-2d 54 79 70 65 3a 20 61 .Content-Type: a 00000020: 70 70 6c 69 63 61 74 69-6f 6e 2f 6f 63 74 65 74 pplication/octet 00000030: 2d 73 74 72 65 61 6d 0d-0a 43 6f 6e 74 65 6e 74 -stream.Content 00000040: 2d 4c 65 6e 67 74 68 3a-20 31 36 38 0d 0a 0d 0a -Length: 168.... 00000050: 42 00 7b 01 00 00 00 a0-42 00 7a 01 00 00 00 48 B.{.... B.z....H 00000060: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j..... 00000070: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04 B.k..... 00000080: 00 00 00 00 00 00 00 00-42 00 92 09 00 00 00 08 B..... 00000090: 00 00 00 00 51 ca af bd-42 00 0d 02 00 00 00 04 QJ/=B..... 000000a0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 48 B.....H 000000b0: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 B.\..... 000000c0: 42 00 7f 05 00 00 00 04-00 00 01 00 00 00 00 B..... 000000d0: 42 00 7e 05 00 00 00 04-00 00 02 00 00 00 00 B.~..... 000000e0: 42 00 7d 07 00 00 00 09-54 4f 4f 5f 4c 41 52 47 B.}....TOO_LARG 000000f0: 45 00 00 00 00 00 00 00- E..... </pre>	<pre> # TIME 1 0032 <RequestMessage> 0033 <RequestHeader> 0034 <ProtocolVersion> 0035 <ProtocolVersionMajor type="Integer" value="1"/> 0036 <ProtocolVersionMinor type="Integer" value="0"/> 0037 </ProtocolVersion> 0038 <MaximumResponseSize type="Integer" value="2048"/> 0039 <BatchCount type="Integer" value="1"/> 0040 </RequestHeader> 0041 <BatchItem> </pre>

0042	<Operation type="Enumeration" value="Query"/>
0043	<RequestPayload>
0044	<QueryFunction type="Enumeration" value="QueryOperations"/>
0045	<QueryFunction type="Enumeration" value="QueryObjects"/>
0046	</RequestPayload>
0047	</BatchItem>
0048	</RequestMessage>
	<pre> 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b020000000400000000000000004200500200000004000008000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000 00000000: 50 4f 53 54 20 2f 6b 6d-69 70 20 48 54 54 50 2f POST /kmip HTTP/ 00000010: 31 2e 30 0d 0a 50 72 61-67 6d 61 3a 20 6e 6f 2d 1.0..Pragma: no- 00000020: 63 61 63 68 65 0d 0a 43-61 63 68 65 2d 43 6f 6e cache..Cache-Con 00000030: 74 72 6f 6c 3a 20 6e 6f-2d 63 61 63 68 65 0d 0a trol: no-cache.. 00000040: 43 6f 6e 6e 65 63 74 69-6f 6e 3a 20 6b 65 65 70 Connection: keep 00000050: 2d 61 6c 69 76 65 0d 0a-43 6f 6e 74 65 6e 74 2d -alive..Content- 00000060: 54 79 70 65 3a 20 61 70-70 6c 69 63 61 74 69 6f Type: applicatio 00000070: 6e 2f 6f 63 74 65 74 2d-73 74 72 65 61 6d 0d 0a n/octet-stream.. 00000080: 43 6f 6e 74 65 6e 74 2d-4c 65 6e 67 74 68 3a 20 Content-Length: 00000090: 31 35 32 20 20 20 20 20-20 20 0d 0a 0d 0a 42 00 152 B. 000000a0: 15 32 78 01 00 00 00 90-42 00 77 01 00 00 00 48 .2x.....B.w....H 000000b0: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j..... 000000c0: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04 B.k..... 000000d0: 00 00 00 00 00 00 00 00-42 00 50 02 00 00 00 04 B.P..... 000000e0: 00 00 08 00 00 00 00 00-42 00 0d 02 00 00 00 04 B..... 000000f0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 38 B.....8 00000100: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 B.\..... 00000110: 42 00 79 01 00 00 00 20-42 00 74 05 00 00 00 04 B.y.... B.t..... 00000120: 00 00 00 01 00 00 00 00-42 00 74 05 00 00 00 04 B.t..... 00000130: 00 00 00 02 00 00 00 00- </pre>
0049	<ResponseMessage>
0050	<ResponseHeader>
0051	<ProtocolVersion>
0052	<ProtocolVersionMajor type="Integer" value="1"/>
0053	<ProtocolVersionMinor type="Integer" value="0"/>
0054	</ProtocolVersion>
0055	<TimeStamp type="DateTime" value="2013-06-26T09:09:17+00:00"/>
0056	<BatchCount type="Integer" value="1"/>
0057	</ResponseHeader>
0058	<BatchItem>
0059	<Operation type="Enumeration" value="Query"/>
0060	<ResultStatus type="Enumeration" value="Success"/>
0061	<ResponsePayload>
0062	<Operation type="Enumeration" value="Query"/>
0063	<Operation type="Enumeration" value="Locate"/>
0064	<Operation type="Enumeration" value="Destroy"/>
0065	<Operation type="Enumeration" value="Get"/>
0066	<Operation type="Enumeration" value="Create"/>
0067	<Operation type="Enumeration" value="Register"/>
0068	<Operation type="Enumeration" value="GetAttributes"/>
0069	<Operation type="Enumeration" value="GetAttributeList"/>
0070	<Operation type="Enumeration" value="AddAttribute"/>
0071	<Operation type="Enumeration" value="ModifyAttribute"/>
0072	<Operation type="Enumeration" value="DeleteAttribute"/>
0073	<Operation type="Enumeration" value="Activate"/>
0074	<Operation type="Enumeration" value="Revoke"/>
0075	<Operation type="Enumeration" value="Poll"/>
0076	<Operation type="Enumeration" value="Cancel"/>
0077	<Operation type="Enumeration" value="Check"/>

```

0078 <Operation type="Enumeration" value="GetUsageAllocation"/>
0079 <Operation type="Enumeration" value="CreateKeyPair"/>
0080 <Operation type="Enumeration" value="ReKey"/>
0081 <Operation type="Enumeration" value="Archive"/>
0082 <Operation type="Enumeration" value="Recover"/>
0083 <Operation type="Enumeration" value="ObtainLease"/>
0084 <Operation type="Enumeration" value="Certify"/>
0085 <Operation type="Enumeration" value="ReCertify"/>
0086 <Operation type="Enumeration" value="Notify"/>
0087 <Operation type="Enumeration" value="Put"/>
0088 <ObjectType type="Enumeration" value="Certificate"/>
0089 <ObjectType type="Enumeration" value="SymmetricKey"/>
0090 <ObjectType type="Enumeration" value="SecretData"/>
0091 <ObjectType type="Enumeration" value="PublicKey"/>
0092 <ObjectType type="Enumeration" value="PrivateKey"/>
0093 <ObjectType type="Enumeration" value="Template"/>
0094 <ObjectType type="Enumeration" value="OpaqueObject"/>
0095 <ObjectType type="Enumeration" value="SplitKey"/>
0096 </ResponsePayload>
0097 </BatchItem>
0098 </ResponseMessage>

42007b01000002a042007a0100000048420069010000002042006a02000000040000000100000000
42006b0200000004000000000000004200920900000008000000051caafbd4200d020000004
000000010000000042000f010000024842005c0500000004000000180000000042007f0500000004
00000000000000042007c010000022042005c0500000004000000180000000042005c0500000004
000000080000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c050000000400000001000000042005c05000000040000000300000000420000c0500000004
0000000b0000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
000000120000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
000000110000000042005c0500000004000000020000000042005c0500000004000000007000000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c0500000004000000060000000042005c05000000040000000700000000
42005c05000000040000001b0000000042005c05000000040000001c000000004200570500000004
00000010000000004200570500000004000000020000000042005705000000040000000700000000
42005705000000040000000300000000420057050000000400000004000000004200570500000004
00000006000000004200570500000004000000080000000042005705000000040000000500000000

00000000: 48 54 54 50 2f 31 2e 31-20 32 30 30 20 4f 4b 0d HTTP/1.1 200 OK.
00000010: 0a 43 6f 6e 74 65 6e 74-2d 54 79 70 65 3a 20 61 .Content-Type: a
00000020: 70 70 6c 69 63 61 74 69-6f 6e 2f 6f 63 74 65 74 pplication/octet
00000030: 2d 73 74 72 65 61 6d 0d-0a 43 6f 6e 74 65 6e 74 -stream..Content
00000040: 2d 4c 65 6e 67 74 68 3a-20 36 38 30 0d 0a 0d 0a -Length: 680....
00000050: 42 00 7b 01 00 00 02 a0-42 00 7a 01 00 00 00 48 B.{.... B.z....H
00000060: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j.....
00000070: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04 .....B.k.....
00000080: 00 00 00 00 00 00 00 00-42 00 92 09 00 00 00 08 .....B.....
00000090: 00 00 00 00 51 ca af bd-42 00 0d 02 00 00 00 04 ....QJ/=B.....
000000a0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 02 48 .....B.....H
000000b0: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 B.\.....
000000c0: 42 00 7f 05 00 00 00 04-00 00 00 00 00 00 00 B.....
000000d0: 42 00 7c 01 00 00 02 20-42 00 5c 05 00 00 00 04 B.|.... B.\....
000000e0: 00 00 00 18 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
000000f0: 00 00 00 08 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000100: 00 00 00 14 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000110: 00 00 00 0a 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000120: 00 00 00 01 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000130: 00 00 00 03 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000140: 00 00 00 0b 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000150: 00 00 00 0c 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000160: 00 00 00 0d 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000170: 00 00 00 0e 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000180: 00 00 00 0f 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
00000190: 00 00 00 12 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....
000001a0: 00 00 00 13 00 00 00 00-42 00 5c 05 00 00 00 04 .....B.\....

```

000001b0:	00 00 00 1a 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001c0:	00 00 00 19 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001d0:	00 00 00 09 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001e0:	00 00 00 11 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001f0:	00 00 00 02 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000200:	00 00 00 04 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000210:	00 00 00 15 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000220:	00 00 00 16 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000230:	00 00 00 10 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000240:	00 00 00 06 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000250:	00 00 00 07 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000260:	00 00 00 1b 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000270:	00 00 00 1c 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
00000280:	00 00 00 01 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
00000290:	00 00 00 02 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002a0:	00 00 00 07 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002b0:	00 00 00 03 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002c0:	00 00 00 04 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002d0:	00 00 00 06 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002e0:	00 00 00 08 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002f0:	00 00 00 05 00 00 00 00-

114

115 3.2 Mandatory HTTPS Profile Test Cases KMIP v1.1

116 3.2.1 MSGENC-HTTPS-M-1-11 - Query, Maximum Response Size

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

121 The specific list of operations and object types returned in the response MAY vary.

```

# TIME 0
0001 <RequestMessage>
0002   <RequestHeader>
0003     <ProtocolVersion>
0004       <ProtocolVersionMajor type="Integer" value="1"/>
0005       <ProtocolVersionMinor type="Integer" value="1"/>
0006     </ProtocolVersion>
0007     <MaximumResponseSize type="Integer" value="256"/>
0008     <BatchCount type="Integer" value="1"/>
0009   </RequestHeader>
0010   <BatchItem>
0011     <Operation type="Enumeration" value="Query"/>
0012     <RequestPayload>
0013       <QueryFunction type="Enumeration" value="QueryOperations"/>
0014       <QueryFunction type="Enumeration" value="QueryObjects"/>
0015     </RequestPayload>
0016   </BatchItem>
0017 </RequestMessage>

42007801000000904200770100000048420069010000002042006a02000000040000000100000000
42006b02000000040000000100000000420050020000000400000100000000042000d0200000004
000000010000000042000f010000003842005c050000000400000018000000004200790100000020
4200740500000004000000010000000042007405000000040000000200000000

00000000: 50 4f 53 54 20 2f 6b 6d-69 70 20 48 54 54 50 2f  POST /kmp HTTP/
00000010: 31 2e 30 0d 0a 50 72 61-67 6d 61 3a 20 6e 6f 2d  1.0..Pragma: no-
00000020: 63 61 63 68 65 0d 0a 43-61 63 68 65 2d 43 6f 6e  cache..Cache-Con
00000030: 74 72 6f 6c 3a 20 6e 6f-2d 63 61 63 68 65 0d 0a  trol: no-cache..
00000040: 43 6f 6e 6e 65 63 74 69-6f 6e 3a 20 6b 65 65 70  Connection: keep
00000050: 2d 61 6c 69 76 65 0d 0a-43 6f 6e 74 65 6e 74 2d  -alive..Content-
```

	<pre> 00000060: 54 79 70 65 3a 20 61 70-70 6c 69 63 61 74 69 6f Type: applicatio 00000070: 6e 2f 6f 63 74 65 74 2d-73 74 72 65 61 6d 0d 0a n/octet-stream.. 00000080: 43 6f 6e 74 65 6e 74 2d-4c 65 6e 67 74 68 3a 20 Content-Length: 00000090: 31 35 32 20 20 20 20 20-20 20 0d 0a 0d 0a 42 00 152 B. 000000a0: 15 32 78 01 00 00 00 90-42 00 77 01 00 00 00 48 .2x.....B.w....H 000000b0: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j..... 000000c0: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04 B.k..... 000000d0: 00 00 00 01 00 00 00 00-42 00 50 02 00 00 00 04 B.P..... 000000e0: 00 00 01 00 00 00 00 00-42 00 0d 02 00 00 00 04 B..... 000000f0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 38 B.....8 00000100: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 B.\..... 00000110: 42 00 79 01 00 00 00 20-42 00 74 05 00 00 00 04 B.y.... B.t.... 00000120: 00 00 00 01 00 00 00 00-42 00 74 05 00 00 00 04 B.t..... 00000130: 00 00 00 02 00 00 00 00- </pre>
<pre> 0018 0019 0020 0021 0022 0023 0024 0025 0026 0027 0028 0029 0030 0031 0032 </pre>	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="1"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2014-06-10T08:03:34+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="OperationFailed"/> <ResultReason type="Enumeration" value="ResponseTooLarge"/> <ResultMessage type="TextString" value="TOO_LARGE"/> </BatchItem> </ResponseMessage> 42007b01000000a042007a0100000048420069010000002042006a020000000040000000100000000 42006b0200000004000000010000000042009209000000080000000005396bc244200d0200000004 000000010000000042000f010000004842005c0500000004000000180000000042007f0500000004 000000010000000042007e0500000004000000020000000042007d0700000009544f4f5f4c415247 4500000000000000 00000000: 48 54 54 50 2f 31 2e 31-20 32 30 30 20 4f 4b 0d HTTP/1.1 200 OK. 00000010: 0a 43 6f 6e 74 65 6e 74-2d 54 79 70 65 3a 20 61 .Content-Type: a 00000020: 70 70 6c 69 63 61 74 69-6f 6e 2f 6f 63 74 65 74 pplication/octet 00000030: 2d 73 74 72 65 61 6d 0d-0a 43 6f 6e 74 65 6e 74 -stream.Content 00000040: 2d 4c 65 6e 67 74 68 3a-20 31 36 38 0d 0a 0d 0a -Length: 168.... 00000050: 42 00 7b 01 00 00 00 a0-42 00 7a 01 00 00 00 48 B.{.... B.z....H 00000060: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j..... 00000070: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04 B.k..... 00000080: 00 00 00 01 00 00 00 00-42 00 92 09 00 00 00 08 B..... 00000090: 00 00 00 00 53 96 bc 24-42 00 0d 02 00 00 00 04 S.<\$B..... 000000a0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 48 B.....H 000000b0: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 B.\..... 000000c0: 42 00 7f 05 00 00 00 04-00 00 01 00 00 00 00 B..... 000000d0: 42 00 7e 05 00 00 00 04-00 00 02 00 00 00 00 B.~..... 000000e0: 42 00 7d 07 00 00 00 09-54 4f 4f 5f 4c 41 52 47 B.}....TOO_LARG 000000f0: 45 00 00 00 00 00 00 00- E..... </pre>
<pre> 0033 0034 0035 0036 0037 0038 0039 0040 0041 0042 </pre>	<pre> # TIME 1 <RequestMessage> <RequestHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="1"/> </ProtocolVersion> <MaximumResponseSize type="Integer" value="2048"/> <BatchCount type="Integer" value="1"/> </RequestHeader> <BatchItem> </pre>

0043	<Operation type="Enumeration" value="Query"/>
0044	<RequestPayload>
0045	<QueryFunction type="Enumeration" value="QueryOperations"/>
0046	<QueryFunction type="Enumeration" value="QueryObjects"/>
0047	</RequestPayload>
0048	</BatchItem>
0049	</RequestMessage>
	<pre> 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b0200000004000000010000000042005002000000040000008000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000 00000000: 50 4f 53 54 20 2f 6b 6d-69 70 20 48 54 54 50 2f POST /kmip HTTP/ 00000010: 31 2e 30 0d 0a 50 72 61-67 6d 61 3a 20 6e 6f 2d 1.0..Pragma: no- 00000020: 63 61 63 68 65 0d 0a 43-61 63 68 65 2d 43 6f 6e cache..Cache-Con 00000030: 74 72 6f 6c 3a 20 6e 6f-2d 63 61 63 68 65 0d 0a trol: no-cache.. 00000040: 43 6f 6e 6e 65 63 74 69-6f 6e 3a 20 6b 65 65 70 Connection: keep 00000050: 2d 61 6c 69 76 65 0d 0a-43 6f 6e 74 65 6e 74 2d -alive..Content- 00000060: 54 79 70 65 3a 20 61 70-70 6c 69 63 61 74 69 6f Type: applicatio 00000070: 6e 2f 6f 63 74 65 74 2d-73 74 72 65 61 6d 0d 0a n/octet-stream.. 00000080: 43 6f 6e 74 65 6e 74 2d-4c 65 6e 67 74 68 3a 20 Content-Length: 00000090: 31 35 32 20 20 20 20 20-20 20 0d 0a 0d 0a 42 00 152 B. 000000a0: 15 32 78 01 00 00 00 90-42 00 77 01 00 00 00 48 .2x.....B.w....H 000000b0: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j..... 000000c0: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04 B.k..... 000000d0: 00 00 00 01 00 00 00 00-42 00 50 02 00 00 00 04 B.P..... 000000e0: 00 00 08 00 00 00 00 00-42 00 0d 02 00 00 00 04 B..... 000000f0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 38 B.....8 00000100: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 B.\..... 00000110: 42 00 79 01 00 00 00 20-42 00 74 05 00 00 00 04 B.y.... B.t..... 00000120: 00 00 00 01 00 00 00 00-42 00 74 05 00 00 00 04 B.t..... 00000130: 00 00 00 02 00 00 00 00- </pre>
0050	<ResponseMessage>
0051	<ResponseHeader>
0052	<ProtocolVersion>
0053	<ProtocolVersionMajor type="Integer" value="1"/>
0054	<ProtocolVersionMinor type="Integer" value="1"/>
0055	</ProtocolVersion>
0056	<TimeStamp type="DateTime" value="2014-06-10T08:03:34+00:00"/>
0057	<BatchCount type="Integer" value="1"/>
0058	</ResponseHeader>
0059	<BatchItem>
0060	<Operation type="Enumeration" value="Query"/>
0061	<ResultStatus type="Enumeration" value="Success"/>
0062	<ResponsePayload>
0063	<Operation type="Enumeration" value="Query"/>
0064	<Operation type="Enumeration" value="Locate"/>
0065	<Operation type="Enumeration" value="Destroy"/>
0066	<Operation type="Enumeration" value="Get"/>
0067	<Operation type="Enumeration" value="Create"/>
0068	<Operation type="Enumeration" value="Register"/>
0069	<Operation type="Enumeration" value="GetAttributes"/>
0070	<Operation type="Enumeration" value="GetAttributeList"/>
0071	<Operation type="Enumeration" value="AddAttribute"/>
0072	<Operation type="Enumeration" value="ModifyAttribute"/>
0073	<Operation type="Enumeration" value="DeleteAttribute"/>
0074	<Operation type="Enumeration" value="Activate"/>
0075	<Operation type="Enumeration" value="Revoke"/>
0076	<Operation type="Enumeration" value="Poll"/>
0077	<Operation type="Enumeration" value="Cancel"/>
0078	<Operation type="Enumeration" value="Check"/>


```

0079     <Operation type="Enumeration" value="GetUsageAllocation"/>
0080     <Operation type="Enumeration" value="CreateKeyPair"/>
0081     <Operation type="Enumeration" value="ReKey"/>
0082     <Operation type="Enumeration" value="Archive"/>
0083     <Operation type="Enumeration" value="Recover"/>
0084     <Operation type="Enumeration" value="ObtainLease"/>
0085     <Operation type="Enumeration" value="ReKeyKeyPair"/>
0086     <Operation type="Enumeration" value="Certify"/>
0087     <Operation type="Enumeration" value="ReCertify"/>
0088     <Operation type="Enumeration" value="DiscoverVersions"/>
0089     <Operation type="Enumeration" value="Notify"/>
0090     <Operation type="Enumeration" value="Put"/>
0091     <ObjectType type="Enumeration" value="Certificate"/>
0092     <ObjectType type="Enumeration" value="SymmetricKey"/>
0093     <ObjectType type="Enumeration" value="SecretData"/>
0094     <ObjectType type="Enumeration" value="PublicKey"/>
0095     <ObjectType type="Enumeration" value="PrivateKey"/>
0096     <ObjectType type="Enumeration" value="Template"/>
0097     <ObjectType type="Enumeration" value="OpaqueObject"/>
0098     <ObjectType type="Enumeration" value="SplitKey"/>
0099     </ResponsePayload>
0100 </BatchItem>
0101 </ResponseMessage>

42007b01000002c042007a0100000048420069010000002042006a02000000040000000100000000
42006b02000000040000000010000000042009209000000080000000005396bc2442000d0200000004
000000010000000042000f010000026842005c0500000004000000180000000042007f0500000004
000000000000000042007c010000024042005c0500000004000000180000000042005c0500000004
000000080000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c0500000004000000010000000042005c0500000004000000030000000042005c0500000004
0000000b0000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
000000120000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
000000110000000042005c050000000400000020000000042005c05000000040000000400000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c05000000040000001d0000000042005c05000000040000000600000000
42005c0500000004000000070000000042005c05000000040000001e0000000042005c0500000004
0000001b0000000042005c05000000040000001c0000000042005705000000040000000100000000
42005705000000040000000200000000420057050000000400000007000000004200570500000004
00000003000000004200570500000004000000040000000042005705000000040000000600000000
4200570500000004000000080000000042005705000000040000000500000000

00000000: 48 54 54 50 2f 31 2e 31-20 32 30 30 20 4f 4b 0d   HTTP/1.1 200 OK.
00000010: 0a 43 6f 6e 74 65 6e 74-2d 54 79 70 65 3a 20 61   .Content-Type: a
00000020: 70 70 6c 69 63 61 74 69-6f 6e 2f 6f 63 74 65 74   pplication/octet
00000030: 2d 73 74 72 65 61 6d 0d-0a 43 6f 6e 74 65 6e 74   -stream..Content
00000040: 2d 4c 65 6e 67 74 68 3a-20 37 31 32 0d 0a 0d 0a   -Length: 712....
00000050: 42 00 7b 01 00 00 02 c0-42 00 7a 01 00 00 00 48   B.{....@B.z....H
00000060: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04   B.i.... B.j.....
00000070: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04   .....B.k.....
00000080: 00 00 00 01 00 00 00 00-42 00 92 09 00 00 00 08   .....B.....
00000090: 00 00 00 00 53 96 bc 24-42 00 0d 02 00 00 00 04   ....S.<$B.....
000000a0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 02 68   .....B.....h
000000b0: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 00   B.\.....
000000c0: 42 00 7f 05 00 00 00 04-00 00 00 00 00 00 00 00   B.....
000000d0: 42 00 7c 01 00 00 02 40-42 00 5c 05 00 00 00 04   B.|....@B.\....
000000e0: 00 00 00 18 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....
000000f0: 00 00 00 08 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....
00000100: 00 00 00 14 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....
00000110: 00 00 00 0a 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....
00000120: 00 00 00 01 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....
00000130: 00 00 00 03 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....
00000140: 00 00 00 0b 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....
00000150: 00 00 00 0c 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....
00000160: 00 00 00 0d 00 00 00 00-42 00 5c 05 00 00 00 04   .....B.\....

```

00000170:	00 00 00 0e 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000180:	00 00 00 0f 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000190:	00 00 00 12 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001a0:	00 00 00 13 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001b0:	00 00 00 1a 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001c0:	00 00 00 19 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001d0:	00 00 00 09 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001e0:	00 00 00 11 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001f0:	00 00 00 02 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000200:	00 00 00 04 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000210:	00 00 00 15 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000220:	00 00 00 16 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000230:	00 00 00 10 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000240:	00 00 00 1d 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000250:	00 00 00 06 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000260:	00 00 00 07 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000270:	00 00 00 1e 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000280:	00 00 00 1b 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000290:	00 00 00 1c 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002a0:	00 00 00 01 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002b0:	00 00 00 02 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002c0:	00 00 00 07 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002d0:	00 00 00 03 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002e0:	00 00 00 04 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000002f0:	00 00 00 06 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
00000300:	00 00 00 08 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
00000310:	00 00 00 05 00 00 00 00-

122

123 3.3 Mandatory HTTPS Profile Test Cases KMIP v1.2

124 3.3.1 MSGENC-HTTPS-M-1-12 - Query, Maximum Response Size

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

129 The specific list of operations and object types returned in the response MAY vary.

```

0001 # TIME 0
0002 <RequestMessage>
0003   <RequestHeader>
0004     <ProtocolVersion>
0005       <ProtocolVersionMajor type="Integer" value="1"/>
0006       <ProtocolVersionMinor type="Integer" value="2"/>
0007     </ProtocolVersion>
0008     <MaximumResponseSize type="Integer" value="256"/>
0009     <BatchCount type="Integer" value="1"/>
0010   </RequestHeader>
0011   <BatchItem>
0012     <Operation type="Enumeration" value="Query"/>
0013     <RequestPayload>
0014       <QueryFunction type="Enumeration" value="QueryOperations"/>
0015       <QueryFunction type="Enumeration" value="QueryObjects"/>
0016     </RequestPayload>
0017   </BatchItem>
</RequestMessage>

42007801000000904200770100000048420069010000002042006a02000000040000000100000000
42006b02000000040000000200000000420050020000000400000100000000042000d0200000004
000000010000000042000f010000003842005c050000000400000018000000004200790100000020
4200740500000004000000010000000042007405000000040000000200000000

```

	<pre> 00000000: 50 4f 53 54 20 2f 6b 6d-69 70 20 48 54 54 50 2f POST /kmp HTTP/ 00000010: 31 2e 30 0d 0a 50 72 61-67 6d 61 3a 20 6e 6f 2d 1.0..Pragma: no- 00000020: 63 61 63 68 65 0d 0a 43-61 63 68 65 2d 43 6f 6e cache..Cache-Con 00000030: 74 72 6f 6c 3a 20 6e 6f-2d 63 61 63 68 65 0d 0a trol: no-cache.. 00000040: 43 6f 6e 6e 65 63 74 69-6f 6e 3a 20 6b 65 65 70 Connection: keep 00000050: 2d 61 6c 69 76 65 0d 0a-43 6f 6e 74 65 6e 74 2d -alive..Content- 00000060: 54 79 70 65 3a 20 61 70-70 6c 69 63 61 74 69 6f Type: applicatio 00000070: 6e 2f 6f 63 74 65 74 2d-73 74 72 65 61 6d 0d 0a n/octet-stream.. 00000080: 43 6f 6e 74 65 6e 74 2d-4c 65 6e 67 74 68 3a 20 Content-Length: 00000090: 31 35 32 20 20 20 20 20-20 20 0d 0a 0d 0a 42 00 152B. 000000a0: 15 32 78 01 00 00 00 90-42 00 77 01 00 00 00 48 .2x.....B.w....H 000000b0: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j.... 000000c0: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04B.k.... 000000d0: 00 00 00 02 00 00 00 00-42 00 50 02 00 00 00 04B.P.... 000000e0: 00 00 01 00 00 00 00 00-42 00 0d 02 00 00 00 04B..... 000000f0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 38B.....8 00000100: 42 00 5c 05 00 00 00 04-00 00 00 18 00 00 00 00 B.\..... 00000110: 42 00 79 01 00 00 00 20-42 00 74 05 00 00 00 04 B.y.... B.t.... 00000120: 00 00 00 01 00 00 00 00-42 00 74 05 00 00 00 04B.t.... 00000130: 00 00 00 02 00 00 00 00- </pre>
<pre> 0018 0019 0020 0021 0022 0023 0024 0025 0026 0027 0028 0029 0030 0031 0032 </pre>	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="2"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2014-06-10T08:07:28+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="OperationFailed"/> <ResultReason type="Enumeration" value="ResponseTooLarge"/> <ResultMessage type="TextString" value="TOO_LARGE"/> </BatchItem> </ResponseMessage> </pre> <p>42007b01000000a042007a0100000048420069010000002042006a02000000040000000100000000</p> <p>42006b02000000040000000200000000420092090000008000000005396bcc042000d0200000004</p> <p>000000010000000042000f010000004842005c0500000004000000180000000042007f0500000004</p> <p>000000010000000042007e0500000004000000020000000042007d0700000009544f45f4c415247</p> <p>4500000000000000</p> <pre> 00000000: 48 54 54 50 2f 31 2e 31-20 32 30 30 20 4f 4b 0d HTTP/1.1 200 OK. 00000010: 0a 43 6f 6e 74 65 6e 74-2d 54 79 70 65 3a 20 61 .Content-Type: a 00000020: 70 70 6c 69 63 61 74 69-6f 6e 2f 6f 63 74 65 74 pplication/octet 00000030: 2d 73 74 72 65 61 6d 0d-0a 43 6f 6e 74 65 6e 74 -stream..Content 00000040: 2d 4c 65 6e 67 74 68 3a-20 31 36 38 0d 0a 0d 0a -Length: 168.... 00000050: 42 00 7b 01 00 00 00 a0-42 00 7a 01 00 00 00 48 B.{.... B.z....H 00000060: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j.... 00000070: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04B.k.... 00000080: 00 00 00 02 00 00 00 00-42 00 92 09 00 00 00 08B..... 00000090: 00 00 00 00 53 96 bd 50-42 00 0d 02 00 00 00 04S.=PB..... 000000a0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 48B.....H 000000b0: 42 00 5c 05 00 00 00 04-00 00 00 18 00 00 00 00 B.\..... 000000c0: 42 00 7f 05 00 00 00 04-00 00 00 01 00 00 00 00 B..... 000000d0: 42 00 7e 05 00 00 00 04-00 00 00 02 00 00 00 00 B.~..... 000000e0: 42 00 7d 07 00 00 00 09-54 4f 4f 5f 4c 41 52 47 B.}.....TOO_LARG 000000f0: 45 00 00 00 00 00 00 00- E..... </pre>
<pre> 0033 0034 0035 0036 0037 </pre>	<pre> # TIME 1 <RequestMessage> <RequestHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="2"/> </pre>

0038	</ProtocolVersion>
0039	<MaximumResponseSize type="Integer" value="2048"/>
0040	<BatchCount type="Integer" value="1"/>
0041	</RequestHeader>
0042	<BatchItem>
0043	<Operation type="Enumeration" value="Query"/>
0044	<RequestPayload>
0045	<QueryFunction type="Enumeration" value="QueryOperations"/>
0046	<QueryFunction type="Enumeration" value="QueryObjects"/>
0047	</RequestPayload>
0048	</BatchItem>
0049	</RequestMessage>
	<pre> 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b02000000040000000200000000420050020000000400000800000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000 00000000: 50 4f 53 54 20 2f 6b 6d-69 70 20 48 54 54 50 2f POST /kmip HTTP/ 00000010: 31 2e 30 0d 0a 50 72 61-67 6d 61 3a 20 6e 6f 2d 1.0..Pragma: no- 00000020: 63 61 63 68 65 0d 0a 43-61 63 68 65 2d 43 6f 6e cache..Cache-Con 00000030: 74 72 6f 6c 3a 20 6e 6f-2d 63 61 63 68 65 0d 0a trol: no-cache.. 00000040: 43 6f 6e 6e 65 63 74 69-6f 6e 3a 20 6b 65 65 70 Connection: keep 00000050: 2d 61 6c 69 76 65 0d 0a-43 6f 6e 74 65 6e 74 2d -alive..Content- 00000060: 54 79 70 65 3a 20 61 70-70 6c 69 63 61 74 69 6f Type: applicatio 00000070: 6e 2f 6f 63 74 65 74 2d-73 74 72 65 61 6d 0d 0a n/octet-stream.. 00000080: 43 6f 6e 74 65 6e 74 2d-4c 65 6e 67 74 68 3a 20 Content-Length: 00000090: 31 35 32 20 20 20 20 20-20 20 0d 0a 0d 0a 42 00 152 B. 000000a0: 15 32 78 01 00 00 00 90-42 00 77 01 00 00 00 48 .2x.....B.w....H 000000b0: 42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04 B.i.... B.j.... 000000c0: 00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04 B.k..... 000000d0: 00 00 00 02 00 00 00 00-42 00 50 02 00 00 00 04 B.P..... 000000e0: 00 00 08 00 00 00 00 00-42 00 0d 02 00 00 00 04 B..... 000000f0: 00 00 00 01 00 00 00 00-42 00 0f 01 00 00 00 38 B.....8 00000100: 42 00 5c 05 00 00 00 04-00 00 18 00 00 00 00 B.\..... 00000110: 42 00 79 01 00 00 00 20-42 00 74 05 00 00 00 04 B.y.... B.t.... 00000120: 00 00 00 01 00 00 00 00-42 00 74 05 00 00 00 04 B.t.... 00000130: 00 00 00 02 00 00 00 00- </pre>
0050	<ResponseMessage>
0051	<ResponseHeader>
0052	<ProtocolVersion>
0053	<ProtocolVersionMajor type="Integer" value="1"/>
0054	<ProtocolVersionMinor type="Integer" value="2"/>
0055	</ProtocolVersion>
0056	<TimeStamp type="DateTime" value="2014-06-10T08:07:28+00:00"/>
0057	<BatchCount type="Integer" value="1"/>
0058	</ResponseHeader>
0059	<BatchItem>
0060	<Operation type="Enumeration" value="Query"/>
0061	<ResultStatus type="Enumeration" value="Success"/>
0062	<ResponsePayload>
0063	<Operation type="Enumeration" value="Query"/>
0064	<Operation type="Enumeration" value="Locate"/>
0065	<Operation type="Enumeration" value="Destroy"/>
0066	<Operation type="Enumeration" value="Get"/>
0067	<Operation type="Enumeration" value="Create"/>
0068	<Operation type="Enumeration" value="Register"/>
0069	<Operation type="Enumeration" value="GetAttributes"/>
0070	<Operation type="Enumeration" value="GetAttributeList"/>
0071	<Operation type="Enumeration" value="AddAttribute"/>
0072	<Operation type="Enumeration" value="ModifyAttribute"/>
0073	<Operation type="Enumeration" value="DeleteAttribute"/>

```
0074 <Operation type="Enumeration" value="Activate"/>
0075 <Operation type="Enumeration" value="Revoke"/>
0076 <Operation type="Enumeration" value="Poll"/>
0077 <Operation type="Enumeration" value="Cancel"/>
0078 <Operation type="Enumeration" value="Check"/>
0079 <Operation type="Enumeration" value="GetUsageAllocation"/>
0080 <Operation type="Enumeration" value="CreateKeyPair"/>
0081 <Operation type="Enumeration" value="ReKey"/>
0082 <Operation type="Enumeration" value="Archive"/>
0083 <Operation type="Enumeration" value="Recover"/>
0084 <Operation type="Enumeration" value="ObtainLease"/>
0085 <Operation type="Enumeration" value="ReKeyKeyPair"/>
0086 <Operation type="Enumeration" value="Certify"/>
0087 <Operation type="Enumeration" value="ReCertify"/>
0088 <Operation type="Enumeration" value="DiscoverVersions"/>
0089 <Operation type="Enumeration" value="Notify"/>
0090 <Operation type="Enumeration" value="Put"/>
0091 <Operation type="Enumeration" value="RNGRetrieve"/>
0092 <Operation type="Enumeration" value="RNGSeed"/>
0093 <Operation type="Enumeration" value="Encrypt"/>
0094 <Operation type="Enumeration" value="Decrypt"/>
0095 <Operation type="Enumeration" value="Sign"/>
0096 <Operation type="Enumeration" value="SignatureVerify"/>
0097 <Operation type="Enumeration" value="MAC"/>
0098 <Operation type="Enumeration" value="MACVerify"/>
0099 <Operation type="Enumeration" value="Hash"/>
0100 <Operation type="Enumeration" value="CreateSplitKey"/>
0101 <Operation type="Enumeration" value="JoinSplitKey"/>
0102 <ObjectType type="Enumeration" value="Certificate"/>
0103 <ObjectType type="Enumeration" value="SymmetricKey"/>
0104 <ObjectType type="Enumeration" value="SecretData"/>
0105 <ObjectType type="Enumeration" value="PublicKey"/>
0106 <ObjectType type="Enumeration" value="PrivateKey"/>
0107 <ObjectType type="Enumeration" value="Template"/>
0108 <ObjectType type="Enumeration" value="OpaqueObject"/>
0109 <ObjectType type="Enumeration" value="SplitKey"/>
0110 <ObjectType type="Enumeration" value="PGPKey"/>
0111 </ResponsePayload>
0112 </BatchItem>
0113 </ResponseMessage>

42007b010000038042007a0100000048420069010000002042006a02000000040000000100000000
42006b02000000040000000020000000042009209000000080000000005396bcc042000d0200000004
0000000100000000042000f010000032842005c0500000004000000180000000042007f0500000004
0000000000000000042007c010000030042005c0500000004000000180000000042005c0500000004
0000000800000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c0500000004000000010000000042005c0500000004000000030000000042005c0500000004
0000000b00000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
0000001200000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
0000001100000000042005c0500000004000000020000000042005c05000000040000000400000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c05000000040000001d0000000042005c05000000040000000600000000
42005c0500000004000000070000000042005c05000000040000001e0000000042005c0500000004
0000001b0000000042005c05000000040000001c0000000042005c05000000040000000250000000
42005c0500000004000000260000000042005c05000000040000001f0000000042005c0500000004
000000200000000042005c0500000004000000210000000042005c05000000040000002200000000
42005c0500000004000000230000000042005c0500000004000000240000000042005c0500000004
000000270000000042005c0500000004000000280000000042005c05000000040000002900000000
42005705000000040000000100000000420057050000000400000002000000004200570500000004
00000007000000004200570500000004000000030000000042005705000000040000000400000000
```

42005705000000040000000600000000420057050000000400000008000000004200570500000004000000050000000042005705000000040000000900000000		
00000000:	48 54 54 50 2f 31 2e 31-20 32 30 30 20 4f 4b 0d	HTTP/1.1 200 OK.
00000010:	0a 43 6f 6e 74 65 6e 74-2d 54 79 70 65 3a 20 61	.Content-Type: a
00000020:	70 70 6c 69 63 61 74 69-6f 6e 2f 6f 63 74 65 74	pplication/octet
00000030:	2d 73 74 72 65 61 6d 0d-0a 43 6f 6e 74 65 6e 74	-stream..Content
00000040:	2d 4c 65 6e 67 74 68 3a-20 39 30 34 0d 0a 0d 0a	-Length: 904....
00000050:	42 00 7b 01 00 00 03 80-42 00 7a 01 00 00 00 48	B.{.....B.z....H
00000060:	42 00 69 01 00 00 00 20-42 00 6a 02 00 00 00 04	B.i..... B.j.....
00000070:	00 00 00 01 00 00 00 00-42 00 6b 02 00 00 00 04B.k.....
00000080:	00 00 00 02 00 00 00 00-42 00 92 09 00 00 00 08B.....
00000090:	00 00 00 00 53 96 bd 50-42 00 0d 02 00 00 00 04S.=PB.....
000000a0:	00 00 00 01 00 00 00 00-42 00 0f 01 00 00 03 28B..... (
000000b0:	42 00 5c 05 00 00 00 04-00 00 00 18 00 00 00 00	B.\.....
000000c0:	42 00 7f 05 00 00 00 04-00 00 00 00 00 00 00 00	B.....
000000d0:	42 00 7c 01 00 00 03 00-42 00 5c 05 00 00 00 04	B.B.\.....
000000e0:	00 00 00 18 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000000f0:	00 00 00 08 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000100:	00 00 00 14 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000110:	00 00 00 0a 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000120:	00 00 00 01 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000130:	00 00 00 03 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000140:	00 00 00 0b 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000150:	00 00 00 0c 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000160:	00 00 00 0d 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000170:	00 00 00 0e 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000180:	00 00 00 0f 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000190:	00 00 00 12 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001a0:	00 00 00 13 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001b0:	00 00 00 1a 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001c0:	00 00 00 19 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001d0:	00 00 00 09 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001e0:	00 00 00 11 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000001f0:	00 00 00 02 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000200:	00 00 00 04 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000210:	00 00 00 15 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000220:	00 00 00 16 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000230:	00 00 00 10 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000240:	00 00 00 1d 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000250:	00 00 00 06 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000260:	00 00 00 07 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000270:	00 00 00 1e 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000280:	00 00 00 1b 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
00000290:	00 00 00 1c 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000002a0:	00 00 00 25 00 00 00 00-42 00 5c 05 00 00 00 04	...%.B.\.....
000002b0:	00 00 00 26 00 00 00 00-42 00 5c 05 00 00 00 04	...&...B.\.....
000002c0:	00 00 00 1f 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000002d0:	00 00 00 20 00 00 00 00-42 00 5c 05 00 00 00 04B.\.....
000002e0:	00 00 00 21 00 00 00 00-42 00 5c 05 00 00 00 04	...!.B.\.....
000002f0:	00 00 00 22 00 00 00 00-42 00 5c 05 00 00 00 04	...".B.\.....
00000300:	00 00 00 23 00 00 00 00-42 00 5c 05 00 00 00 04	...#.B.\.....
00000310:	00 00 00 24 00 00 00 00-42 00 5c 05 00 00 00 04	...\$.B.\.....
00000320:	00 00 00 27 00 00 00 00-42 00 5c 05 00 00 00 04	...'.B.\.....
00000330:	00 00 00 28 00 00 00 00-42 00 5c 05 00 00 00 04	...(.B.\.....
00000340:	00 00 00 29 00 00 00 00-42 00 57 05 00 00 00 04	...).B.W.....
00000350:	00 00 00 01 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
00000360:	00 00 00 02 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
00000370:	00 00 00 07 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
00000380:	00 00 00 03 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
00000390:	00 00 00 04 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000003a0:	00 00 00 06 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000003b0:	00 00 00 08 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000003c0:	00 00 00 05 00 00 00 00-42 00 57 05 00 00 00 04B.W.....
000003d0:	00 00 00 09 00 00 00 00-

131 4 JSON Profile

132 The JSON profile specifies the use of KMIP replacing the TTLV message encoding with a JSON message
133 encoding. The results returned using the JSON encoding SHALL be logically the same as if the message
134 encoding was in TTLV form. All size or length values specified within tag values for KMIP items SHALL be
135 the same in JSON form as if the message encoding were in TTLV form. The implications of this are that
136 items such as MaximumResponseSize are interpreted to refer to a maximum length computed as if it
137 were a TTLV-encoded response, not the length of the JSON-encoded response.

138 4.1 JSON Encoding

139 4.1.1 Hex representations

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

147 4.1.2 Tags

148 Tags are a String that may contain either:

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

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

154 4.1.3 Normalizing Names

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

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

- 162 1. Replace round brackets ('(', ')') with spaces
- 163 2. If a non-word char (not alpha, digit or underscore) is followed by a letter (either upper or lower
164 case) then a lower case letter, replace the non-word char with space
- 165 3. Replace remaining non-word chars (except whitespace) with underscore.
- 166 4. If the first word begins with a digit, move all digits at start of first word to end of first word
- 167 5. Capitalize the first letter of each word
- 168 6. Concatenate all words with spaces removed
- 169

```

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

```

182 *Example python name normalization code*

```

184 # 1. Replace brackets with space
185 $enumName=~s/[\\(\\)]/ /g;
186 # 2. replace \W with space if followed by letter, lower
187 $enumName=~s/\W([A-Za-z][a-z])/ \1/g;
188 # 3. non-word to underscore
189 @words=split(/ /,$enumName);
190 for($i=0;$i<=$#words;$i++) { $words[$i]=~s/\W/_/g; }
191 # 4. move numbers to end of first word
192 $words[0] =~ s/^\d+ (.*)/\2\1/;
193 # 5. captialize first letter of each word
194 for($i=0;$i<=$#words;$i++) {
195     substr($words[$i],0,1)=~tr/a-z/A-Z/;
196 }
197 # 6. concatenate
198 $enumNameCamel = join(' ',@words);
199

```

200 *Example perl name normalization code*

201 4.1.4 Type

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

- 204 • Structure
- 205 • Integer
- 206 • LongInteger
- 207 • BigInteger
- 208 • Enumeration
- 209 • Boolean
- 210 • TextString
- 211 • ByteString
- 212 • DateTime
- 213 • Interval

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

215 4.1.5 Value

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

217 4.1.6 JSON Object

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

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

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

224 4.1.6.1 Tags

225 Tags are a String that may contain either:

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

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

```
231 {"tag": "0x420001", "type": "DateTime", "value": "2001-01-01T10:00:00+10:00"}  
232 {"tag": "ActivationDate", "type": "DateTime", "value": "2001-01-01T10:00:00+10:00"}  
233 {"tag": "IVCounterNonce", "type": "ByteString", "value": "alb2c3d4"}  
234 {"tag": "PrivateKeyTemplateAttribute", "type": "Structure", "value": []}  
235 {"tag": "0x545352", "type": "TextString", "value": "This is an extension"}  
236 {"tag": "WELL_KNOWN_EXTENSION", "type": "TextString", "value": "This is an extension"}
```

237 4.1.6.2 Structure

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

```
239 {"tag": "ProtocolVersion", "type": "Structure", "value": [  
240   {"tag": "ProtocolVersionMajor", "type": "Integer", "value": 1},  
241   {"tag": "ProtocolVersionMajor", "type": "Integer", "value": 0}  
242 ]}  
243 {"tag": "ProtocolVersion", "value": [  
244   {"tag": "ProtocolVersionMajor", "type": "Integer", "value": 1},  
245   {"tag": "ProtocolVersionMajor", "type": "Integer", "value": 0}  
246 ]}
```

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

248 4.1.6.3 Integer

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

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

252 4.1.6.4 Integer - Special case for Masks

253 (Cryptographic Usage Mask, Storage Status Mask):

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

```
257 {"tag": "CryptographicUsageMask", "type": "Integer", "value": "0x0000100c"}  
258 {"tag": "CryptographicUsageMask", "type": "Integer", "value": "Encrypt|Decrypt|CertificateSign"}  
259 {"tag": "CryptographicUsageMask", "type": "Integer", "value":  
260 "CertificateSign|0x00000004|0x00000008"}  
261 {"tag": "CryptographicUsageMask", "type": "Integer", "value": "CertificateSign|0x0000000c"}
```

262 4.1.6.5 Long Integer

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

```
265 {"tag": "0x540001", "type": "LongInteger", "value": "0xfffffffffffffffffe"}  
266 {"tag": "0x540001", "type": "LongInteger", "value": -2}  
267 {"tag": "UsageLimitsCount", "type": "LongInteger", "value": "0x1000000000000000"}
```

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

269 4.1.6.6 Big Integer

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

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

274 4.1.6.7 Enumeration

275 For JSON, value may contain:

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

```
279  
280 {"tag": "0x420057", "type": "Enumeration", "value": 2}  
281 {"tag": "ObjectType", "type": "Enumeration", "value": "0x00000002"}  
282 {"tag": "ObjectType", "type": "Enumeration", "value": "SymmetricKey"}
```

283 4.1.6.8 Boolean

284 For JSON, value must be either a hex string, or a JSON Boolean 'true' or 'false'.

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

287 4.1.6.9 Text String

288 For JSON, value must be a String

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

290 4.1.6.10 Byte String

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

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

294 4.1.6.11 Date-Time

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

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

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

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

301 4.1.6.12 Interval

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

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

306

5 JSON Profile Test Cases

307 The test cases define a number of request-response pairs for KMIP operations. Each test case is
308 provided in the XML format specified in section 6 intended to be both human-readable and usable by
309 automated tools. The time sequence (starting from 0) for each request-response pair is noted and line
310 numbers are provided for ease of cross-reference for a given test sequence.

311 Each test case has a unique label (the section name) which includes indication of mandatory (-M-) or
312 optional (-O-) status and the protocol version major and minor numbers as part of the identifier.

313 The test cases may depend on a specific configuration of a KMIP client and server being configured in a
314 manner consistent with the test case assumptions.

315 Where possible the flow of unique identifiers between tests, the date-time values, and other dynamic
316 items are indicated using symbolic identifiers – in actual request and response messages these dynamic
317 values will be filled in with valid values.

318 Note: the values for the returned items and the custom attributes are illustrative. Actual values from a real
319 client system may vary as specified in section 8.4

320 5.1 Mandatory JSON Profile Test Cases KMIP v1.0

321 5.1.1 MSGENC-JSON-M-1-10 - Query, Maximum Response Size

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

326 The specific list of operations and object types returned in the response MAY vary.

```

# TIME 0
0001 <RequestMessage>
0002   <RequestHeader>
0003     <ProtocolVersion>
0004       <ProtocolVersionMajor type="Integer" value="1"/>
0005       <ProtocolVersionMinor type="Integer" value="0"/>
0006     </ProtocolVersion>
0007     <MaximumResponseSize type="Integer" value="256"/>
0008     <BatchCount type="Integer" value="1"/>
0009   </RequestHeader>
0010   <BatchItem>
0011     <Operation type="Enumeration" value="Query"/>
0012     <RequestPayload>
0013       <QueryFunction type="Enumeration" value="QueryOperations"/>
0014       <QueryFunction type="Enumeration" value="QueryObjects"/>
0015     </RequestPayload>
0016   </BatchItem>
0017 </RequestMessage>

42007801000000904200770100000048420069010000002042006a02000000040000000100000000
42006b020000000400000000000000004200500200000004000001000000000042000d0200000004
000000010000000042000f010000003842005c050000000400000018000000004200790100000020
4200740500000004000000010000000042007405000000040000000200000000

{"tag":"RequestMessage", "value":[
  {"tag":"RequestHeader", "value":[
    {"tag":"ProtocolVersion", "value":[
      {"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"},
      {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000000"}
    ]}
  ]}

```

	<pre>]}, {"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"}]}]}] </pre>
<pre> 0018 0019 0020 0021 0022 0023 0024 0025 0026 0027 0028 0029 0030 0031 0032 </pre>	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="0"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2013-06-26T09:09:17+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="OperationFailed"/> <ResultReason type="Enumeration" value="ResponseTooLarge"/> <ResultMessage type="TextString" value="TOO_LARGE"/> </BatchItem> </ResponseMessage> 42007b01000000a042007a0100000048420069010000002042006a02000000040000000100000000 42006b0200000004000000000000000042009209000000080000000051caafbd42000d0200000004 000000010000000042000f010000004842005c0500000004000000180000000042007f0500000004 000000010000000042007e0500000004000000020000000042007d0700000009544f4f5f4c415247 4500000000000000 {"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":"2013-06-26T09:09:17+00:00"}, {"tag":"BatchCount", "type":"Integer", "value":"0x00000001"}]}, {"tag":"BatchItem", "value":[{"tag":"Operation", "type":"Enumeration", "value":"Query"}, {"tag":"ResultStatus", "type":"Enumeration", "value":"OperationFailed"}, {"tag":"ResultReason", "type":"Enumeration", "value":"ResponseTooLarge"}, {"tag":"ResultMessage", "type":"TextString", "value":"TOO_LARGE"}]}]} </pre>
<pre> 0032 0033 0034 0035 0036 0037 0038 0039 0040 0041 0042 </pre>	<pre> # TIME 1 <RequestMessage> <RequestHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="0"/> </ProtocolVersion> <MaximumResponseSize type="Integer" value="2048"/> <BatchCount type="Integer" value="1"/> </RequestHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> </pre>

0043	<RequestPayload>
0044	<QueryFunction type="Enumeration" value="QueryOperations"/>
0045	<QueryFunction type="Enumeration" value="QueryObjects"/>
0046	</RequestPayload>
0047	</BatchItem>
0048	</RequestMessage>
	<pre> 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b020000000400000000000000000000042005002000000040000008000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000 {"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"}]}]}]] </pre>
0049	<ResponseMessage>
0050	<ResponseHeader>
0051	<ProtocolVersion>
0052	<ProtocolVersionMajor type="Integer" value="1"/>
0053	<ProtocolVersionMinor type="Integer" value="0"/>
0054	</ProtocolVersion>
0055	<TimeStamp type="DateTime" value="2013-06-26T09:09:17+00:00"/>
0056	<BatchCount type="Integer" value="1"/>
0057	</ResponseHeader>
0058	<BatchItem>
0059	<Operation type="Enumeration" value="Query"/>
0060	<ResultStatus type="Enumeration" value="Success"/>
0061	<ResponsePayload>
0062	<Operation type="Enumeration" value="Query"/>
0063	<Operation type="Enumeration" value="Locate"/>
0064	<Operation type="Enumeration" value="Destroy"/>
0065	<Operation type="Enumeration" value="Get"/>
0066	<Operation type="Enumeration" value="Create"/>
0067	<Operation type="Enumeration" value="Register"/>
0068	<Operation type="Enumeration" value="GetAttributes"/>
0069	<Operation type="Enumeration" value="GetAttributeList"/>
0070	<Operation type="Enumeration" value="AddAttribute"/>
0071	<Operation type="Enumeration" value="ModifyAttribute"/>
0072	<Operation type="Enumeration" value="DeleteAttribute"/>
0073	<Operation type="Enumeration" value="Activate"/>
0074	<Operation type="Enumeration" value="Revoke"/>
0075	<Operation type="Enumeration" value="Poll"/>
0076	<Operation type="Enumeration" value="Cancel"/>
0077	<Operation type="Enumeration" value="Check"/>
0078	<Operation type="Enumeration" value="GetUsageAllocation"/>
0079	<Operation type="Enumeration" value="CreateKeyPair"/>

```

0080 <Operation type="Enumeration" value="ReKey"/>
0081 <Operation type="Enumeration" value="Archive"/>
0082 <Operation type="Enumeration" value="Recover"/>
0083 <Operation type="Enumeration" value="ObtainLease"/>
0084 <Operation type="Enumeration" value="Certify"/>
0085 <Operation type="Enumeration" value="ReCertify"/>
0086 <Operation type="Enumeration" value="Notify"/>
0087 <Operation type="Enumeration" value="Put"/>
0088 <ObjectType type="Enumeration" value="Certificate"/>
0089 <ObjectType type="Enumeration" value="SymmetricKey"/>
0090 <ObjectType type="Enumeration" value="SecretData"/>
0091 <ObjectType type="Enumeration" value="PublicKey"/>
0092 <ObjectType type="Enumeration" value="PrivateKey"/>
0093 <ObjectType type="Enumeration" value="Template"/>
0094 <ObjectType type="Enumeration" value="OpaqueObject"/>
0095 <ObjectType type="Enumeration" value="SplitKey"/>
0096 </ResponsePayload>
0097 </BatchItem>
0098 </ResponseMessage>

42007b01000002a042007a0100000048420069010000002042006a02000000040000000100000000
42006b020000000400000000000000042009209000000080000000051caafb42000d0200000004
000000010000000042000f010000024842005c0500000004000000180000000042007f0500000004
000000000000000042007c010000022042005c0500000004000000180000000042005c0500000004
000000080000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c0500000004000000010000000042005c0500000004000000030000000042005c0500000004
0000000b0000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
000000120000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
000000110000000042005c0500000004000000020000000042005c05000000040000000400000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c0500000004000000060000000042005c05000000040000000700000000
42005c05000000040000001b0000000042005c05000000040000001c00000000420057050000004
00000001000000004200570500000004000000020000000042005705000000040000000700000000
4200570500000004000000030000000042005705000000040000000400000000420057050000004
00000006000000004200570500000004000000080000000042005705000000040000000500000000

{"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":"2013-06-26T09:09:17+00:00"},
    {"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":"Query"},
      {"tag":"Operation", "type":"Enumeration", "value":"Locate"},
      {"tag":"Operation", "type":"Enumeration", "value":"Destroy"},
      {"tag":"Operation", "type":"Enumeration", "value":"Get"},
      {"tag":"Operation", "type":"Enumeration", "value":"Create"},
      {"tag":"Operation", "type":"Enumeration", "value":"Register"},
      {"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":"Activate"},
      {"tag":"Operation", "type":"Enumeration", "value":"Revoke"},
      {"tag":"Operation", "type":"Enumeration", "value":"Poll"},
      {"tag":"Operation", "type":"Enumeration", "value":"Cancel"},
      {"tag":"Operation", "type":"Enumeration", "value":"Check"},
    ]}
  ]}
]

```

```

{"tag": "Operation", "type": "Enumeration", "value": "GetUsageAllocation"},
{"tag": "Operation", "type": "Enumeration", "value": "CreateKeyPair"},
{"tag": "Operation", "type": "Enumeration", "value": "ReKey"},
{"tag": "Operation", "type": "Enumeration", "value": "Archive"},
{"tag": "Operation", "type": "Enumeration", "value": "Recover"},
{"tag": "Operation", "type": "Enumeration", "value": "ObtainLease"},
{"tag": "Operation", "type": "Enumeration", "value": "Certify"},
{"tag": "Operation", "type": "Enumeration", "value": "ReCertify"},
{"tag": "Operation", "type": "Enumeration", "value": "Notify"},
{"tag": "Operation", "type": "Enumeration", "value": "Put"},
{"tag": "ObjectType", "type": "Enumeration", "value": "Certificate"},
{"tag": "ObjectType", "type": "Enumeration", "value": "SymmetricKey"},
{"tag": "ObjectType", "type": "Enumeration", "value": "SecretData"},
{"tag": "ObjectType", "type": "Enumeration", "value": "PublicKey"},
{"tag": "ObjectType", "type": "Enumeration", "value": "PrivateKey"},
{"tag": "ObjectType", "type": "Enumeration", "value": "Template"},
{"tag": "ObjectType", "type": "Enumeration", "value": "OpaqueObject"},
{"tag": "ObjectType", "type": "Enumeration", "value": "SplitKey"}
  ]}
  ]}
}]

```

327

328 5.2 Mandatory JSON Profile Test Cases KMIP v1.1

329 5.2.1 MSGENC-JSON-M-1-11 - Query, Maximum Response Size

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

334 The specific list of operations and object types returned in the response MAY vary.

```

# TIME 0
0001 <RequestMessage>
0002   <RequestHeader>
0003     <ProtocolVersion>
0004       <ProtocolVersionMajor type="Integer" value="1"/>
0005       <ProtocolVersionMinor type="Integer" value="1"/>
0006     </ProtocolVersion>
0007     <MaximumResponseSize type="Integer" value="256"/>
0008     <BatchCount type="Integer" value="1"/>
0009   </RequestHeader>
0010   <BatchItem>
0011     <Operation type="Enumeration" value="Query"/>
0012     <RequestPayload>
0013       <QueryFunction type="Enumeration" value="QueryOperations"/>
0014       <QueryFunction type="Enumeration" value="QueryObjects"/>
0015     </RequestPayload>
0016   </BatchItem>
0017 </RequestMessage>

42007801000000904200770100000048420069010000002042006a02000000040000000100000000
42006b02000000040000000100000000420050020000000400000100000000004200d0200000004
000000010000000042000f010000003842005c050000000400000018000000004200790100000020
4200740500000004000000010000000042007405000000040000000200000000

{"tag": "RequestMessage", "value": [
  {"tag": "RequestHeader", "value": [
    {"tag": "ProtocolVersion", "value": [
      {"tag": "ProtocolVersionMajor", "type": "Integer", "value": "0x00000001"},
      {"tag": "ProtocolVersionMinor", "type": "Integer", "value": "0x00000001"}
    ]}
  ]}
]}

```


0045 0046 0047 0048 0049	<pre> <QueryFunction type="Enumeration" value="QueryOperations"/> <QueryFunction type="Enumeration" value="QueryObjects"/> </RequestPayload> </BatchItem> </RequestMessage> 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b0200000004000000010000000042005002000000040000008000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000 {"tag":"RequestMessage", "value":[{"tag":"RequestHeader", "value":[{"tag":"ProtocolVersion", "value":[{"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"}, {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000001"}]}, {"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"}]}]}]} </pre>
0050 0051 0052 0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0069 0070 0071 0072 0073 0074 0075 0076 0077 0078 0079 0080 0081 0082	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="1"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2014-06-10T08:03:34+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="Success"/> <ResponsePayload> <Operation type="Enumeration" value="Query"/> <Operation type="Enumeration" value="Locate"/> <Operation type="Enumeration" value="Destroy"/> <Operation type="Enumeration" value="Get"/> <Operation type="Enumeration" value="Create"/> <Operation type="Enumeration" value="Register"/> <Operation type="Enumeration" value="GetAttributes"/> <Operation type="Enumeration" value="GetAttributeList"/> <Operation type="Enumeration" value="AddAttribute"/> <Operation type="Enumeration" value="ModifyAttribute"/> <Operation type="Enumeration" value="DeleteAttribute"/> <Operation type="Enumeration" value="Activate"/> <Operation type="Enumeration" value="Revoke"/> <Operation type="Enumeration" value="Poll"/> <Operation type="Enumeration" value="Cancel"/> <Operation type="Enumeration" value="Check"/> <Operation type="Enumeration" value="GetUsageAllocation"/> <Operation type="Enumeration" value="CreateKeyPair"/> <Operation type="Enumeration" value="ReKey"/> <Operation type="Enumeration" value="Archive"/> </pre>

```

0083 <Operation type="Enumeration" value="Recover"/>
0084 <Operation type="Enumeration" value="ObtainLease"/>
0085 <Operation type="Enumeration" value="ReKeyKeyPair"/>
0086 <Operation type="Enumeration" value="Certify"/>
0087 <Operation type="Enumeration" value="ReCertify"/>
0088 <Operation type="Enumeration" value="DiscoverVersions"/>
0089 <Operation type="Enumeration" value="Notify"/>
0090 <Operation type="Enumeration" value="Put"/>
0091 <ObjectType type="Enumeration" value="Certificate"/>
0092 <ObjectType type="Enumeration" value="SymmetricKey"/>
0093 <ObjectType type="Enumeration" value="SecretData"/>
0094 <ObjectType type="Enumeration" value="PublicKey"/>
0095 <ObjectType type="Enumeration" value="PrivateKey"/>
0096 <ObjectType type="Enumeration" value="Template"/>
0097 <ObjectType type="Enumeration" value="OpaqueObject"/>
0098 <ObjectType type="Enumeration" value="SplitKey"/>
0099 </ResponsePayload>
0100 </BatchItem>
0101 </ResponseMessage>

42007b01000002c042007a0100000048420069010000002042006a02000000040000000100000000
42006b020000000400000001000000004200920900000008000000005396bc2442000d0200000004
000000010000000042000f010000026842005c0500000004000000180000000042007f0500000004
000000000000000042007c010000024042005c0500000004000000180000000042005c0500000004
000000800000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c0500000004000000010000000042005c0500000004000000030000000042005c0500000004
000000b0000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
000000120000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
000000110000000042005c0500000004000000020000000042005c05000000040000000400000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c05000000040000001d0000000042005c05000000040000000600000000
42005c0500000004000000070000000042005c05000000040000001e0000000042005c0500000004
0000001b0000000042005c05000000040000001c0000000042005705000000040000000100000000
42005705000000040000000200000000420057050000000400000007000000004200570500000004
0000003000000004200570500000004000000040000000042005705000000040000000600000000
4200570500000004000000080000000042005705000000040000000500000000

{"tag":"ResponseMessage", "value": [
  {"tag":"ResponseHeader", "value": [
    {"tag":"ProtocolVersion", "value": [
      {"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"},
      {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000001"}
    ]},
    {"tag":"TimeStamp", "type":"DateTime", "value":"2014-06-10T08:04:52+00:00"},
    {"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":"Query"},
      {"tag":"Operation", "type":"Enumeration", "value":"Locate"},
      {"tag":"Operation", "type":"Enumeration", "value":"Destroy"},
      {"tag":"Operation", "type":"Enumeration", "value":"Get"},
      {"tag":"Operation", "type":"Enumeration", "value":"Create"},
      {"tag":"Operation", "type":"Enumeration", "value":"Register"},
      {"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":"Activate"},
      {"tag":"Operation", "type":"Enumeration", "value":"Revoke"},
      {"tag":"Operation", "type":"Enumeration", "value":"Poll"},
      {"tag":"Operation", "type":"Enumeration", "value":"Cancel"},
    ]}
  ]}

```

```

{"tag": "Operation", "type": "Enumeration", "value": "Check"},
{"tag": "Operation", "type": "Enumeration", "value": "GetUsageAllocation"},
{"tag": "Operation", "type": "Enumeration", "value": "CreateKeyPair"},
{"tag": "Operation", "type": "Enumeration", "value": "ReKey"},
{"tag": "Operation", "type": "Enumeration", "value": "Archive"},
{"tag": "Operation", "type": "Enumeration", "value": "Recover"},
{"tag": "Operation", "type": "Enumeration", "value": "ObtainLease"},
{"tag": "Operation", "type": "Enumeration", "value": "ReKeyKeyPair"},
{"tag": "Operation", "type": "Enumeration", "value": "Certify"},
{"tag": "Operation", "type": "Enumeration", "value": "ReCertify"},
{"tag": "Operation", "type": "Enumeration", "value": "DiscoverVersions"},
{"tag": "Operation", "type": "Enumeration", "value": "Notify"},
{"tag": "Operation", "type": "Enumeration", "value": "Put"},
{"tag": "ObjectType", "type": "Enumeration", "value": "Certificate"},
{"tag": "ObjectType", "type": "Enumeration", "value": "SymmetricKey"},
{"tag": "ObjectType", "type": "Enumeration", "value": "SecretData"},
{"tag": "ObjectType", "type": "Enumeration", "value": "PublicKey"},
{"tag": "ObjectType", "type": "Enumeration", "value": "PrivateKey"},
{"tag": "ObjectType", "type": "Enumeration", "value": "Template"},
{"tag": "ObjectType", "type": "Enumeration", "value": "OpaqueObject"},
{"tag": "ObjectType", "type": "Enumeration", "value": "SplitKey"}
  ]}
}}
}}
}}

```

335

336 5.3 Mandatory JSON Profile Test Cases KMIP v1.2

337 5.3.1 MSGENC-JSON-M-1-12 - Query, Maximum Response Size

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

342 The specific list of operations and object types returned in the response MAY vary.

```

# TIME 0
0001 <RequestMessage>
0002   <RequestHeader>
0003     <ProtocolVersion>
0004       <ProtocolVersionMajor type="Integer" value="1"/>
0005       <ProtocolVersionMinor type="Integer" value="2"/>
0006     </ProtocolVersion>
0007     <MaximumResponseSize type="Integer" value="256"/>
0008     <BatchCount type="Integer" value="1"/>
0009   </RequestHeader>
0010   <BatchItem>
0011     <Operation type="Enumeration" value="Query"/>
0012     <RequestPayload>
0013       <QueryFunction type="Enumeration" value="QueryOperations"/>
0014       <QueryFunction type="Enumeration" value="QueryObjects"/>
0015     </RequestPayload>
0016   </BatchItem>
0017 </RequestMessage>

42007801000000904200770100000048420069010000002042006a02000000040000000100000000
42006b02000000040000000200000000420050020000000400000100000000042000d0200000004
000000010000000042000f010000003842005c050000000400000018000000004200790100000020
4200740500000004000000010000000042007405000000040000000200000000

{"tag": "RequestMessage", "value": [
  {"tag": "RequestHeader", "value": [
    {"tag": "ProtocolVersion", "value": [

```


<pre> 0043 0044 0045 0046 0047 0048 0049 </pre>	<pre> <Operation type="Enumeration" value="Query"/> <RequestPayload> <QueryFunction type="Enumeration" value="QueryOperations"/> <QueryFunction type="Enumeration" value="QueryObjects"/> </RequestPayload> </BatchItem> </RequestMessage> 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b020000000400000002000000004200500200000004000008000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000 {"tag":"RequestMessage", "value":[{"tag":"RequestHeader", "value":[{"tag":"ProtocolVersion", "value":[{"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"}, {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000002"}]}, {"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"}]}]}]} </pre>
<pre> 0050 0051 0052 0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0069 0070 0071 0072 0073 0074 0075 0076 0077 0078 0079 0080 </pre>	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="2"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2014-06-10T08:07:28+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="Success"/> <ResponsePayload> <Operation type="Enumeration" value="Query"/> <Operation type="Enumeration" value="Locate"/> <Operation type="Enumeration" value="Destroy"/> <Operation type="Enumeration" value="Get"/> <Operation type="Enumeration" value="Create"/> <Operation type="Enumeration" value="Register"/> <Operation type="Enumeration" value="GetAttributes"/> <Operation type="Enumeration" value="GetAttributeList"/> <Operation type="Enumeration" value="AddAttribute"/> <Operation type="Enumeration" value="ModifyAttribute"/> <Operation type="Enumeration" value="DeleteAttribute"/> <Operation type="Enumeration" value="Activate"/> <Operation type="Enumeration" value="Revoke"/> <Operation type="Enumeration" value="Poll"/> <Operation type="Enumeration" value="Cancel"/> <Operation type="Enumeration" value="Check"/> <Operation type="Enumeration" value="GetUsageAllocation"/> <Operation type="Enumeration" value="CreateKeyPair"/> </ResponsePayload> </BatchItem> </ResponseMessage> </pre>

```

0081 <Operation type="Enumeration" value="ReKey"/>
0082 <Operation type="Enumeration" value="Archive"/>
0083 <Operation type="Enumeration" value="Recover"/>
0084 <Operation type="Enumeration" value="ObtainLease"/>
0085 <Operation type="Enumeration" value="ReKeyKeyPair"/>
0086 <Operation type="Enumeration" value="Certify"/>
0087 <Operation type="Enumeration" value="ReCertify"/>
0088 <Operation type="Enumeration" value="DiscoverVersions"/>
0089 <Operation type="Enumeration" value="Notify"/>
0090 <Operation type="Enumeration" value="Put"/>
0091 <Operation type="Enumeration" value="RNGRetrieve"/>
0092 <Operation type="Enumeration" value="RNGSeed"/>
0093 <Operation type="Enumeration" value="Encrypt"/>
0094 <Operation type="Enumeration" value="Decrypt"/>
0095 <Operation type="Enumeration" value="Sign"/>
0096 <Operation type="Enumeration" value="SignatureVerify"/>
0097 <Operation type="Enumeration" value="MAC"/>
0098 <Operation type="Enumeration" value="MACVerify"/>
0099 <Operation type="Enumeration" value="Hash"/>
0100 <Operation type="Enumeration" value="CreateSplitKey"/>
0101 <Operation type="Enumeration" value="JoinSplitKey"/>
0102 <ObjectType type="Enumeration" value="Certificate"/>
0103 <ObjectType type="Enumeration" value="SymmetricKey"/>
0104 <ObjectType type="Enumeration" value="SecretData"/>
0105 <ObjectType type="Enumeration" value="PublicKey"/>
0106 <ObjectType type="Enumeration" value="PrivateKey"/>
0107 <ObjectType type="Enumeration" value="Template"/>
0108 <ObjectType type="Enumeration" value="OpaqueObject"/>
0109 <ObjectType type="Enumeration" value="SplitKey"/>
0110 <ObjectType type="Enumeration" value="PGPKey"/>
0111 </ResponsePayload>
0112 </BatchItem>
0113 </ResponseMessage>

42007b010000038042007a0100000048420069010000002042006a02000000040000000100000000
42006b0200000004000000002000000004200920900000008000000005396bcc042000d0200000004
000000010000000042000f010000032842005c0500000004000000180000000042007f0500000004
000000000000000042007c010000030042005c0500000004000000180000000042005c0500000004
000000080000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c0500000004000000010000000042005c0500000004000000030000000042005c0500000004
0000000b0000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
000000120000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
000000110000000042005c0500000004000000020000000042005c05000000040000000400000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c05000000040000001d0000000042005c05000000040000000600000000
42005c0500000004000000070000000042005c05000000040000001e0000000042005c0500000004
0000001b0000000042005c05000000040000001c0000000042005c05000000040000002500000000
42005c0500000004000000260000000042005c05000000040000001f0000000042005c0500000004
000000200000000042005c0500000004000000210000000042005c05000000040000002200000000
42005c0500000004000000230000000042005c0500000004000000240000000042005c0500000004
000000270000000042005c0500000004000000280000000042005c05000000040000002900000000
42005705000000040000000100000000420057050000000400000002000000004200570500000004
00000007000000004200570500000004000000030000000042005705000000040000000400000000
42005705000000040000000600000000420057050000000400000008000000004200570500000004
000000050000000042005705000000040000000900000000

{"tag":"ResponseMessage", "value":[
  {"tag":"ResponseHeader", "value":[
    {"tag":"ProtocolVersion", "value":[
      {"tag":"ProtocolVersionMajor", "type":"Integer", "value":"0x00000001"},
      {"tag":"ProtocolVersionMinor", "type":"Integer", "value":"0x00000002"}
    ]},
  ]},

```

```

{"tag":"TimeStamp", "type":"DateTime", "value":"2014-06-10T08:07:28+00:00"},
{"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":"Query"},
{"tag":"Operation", "type":"Enumeration", "value":"Locate"},
{"tag":"Operation", "type":"Enumeration", "value":"Destroy"},
{"tag":"Operation", "type":"Enumeration", "value":"Get"},
{"tag":"Operation", "type":"Enumeration", "value":"Create"},
{"tag":"Operation", "type":"Enumeration", "value":"Register"},
{"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":"Activate"},
{"tag":"Operation", "type":"Enumeration", "value":"Revoke"},
{"tag":"Operation", "type":"Enumeration", "value":"Poll"},
{"tag":"Operation", "type":"Enumeration", "value":"Cancel"},
{"tag":"Operation", "type":"Enumeration", "value":"Check"},
{"tag":"Operation", "type":"Enumeration", "value":"GetUsageAllocation"},
{"tag":"Operation", "type":"Enumeration", "value":"CreateKeyPair"},
{"tag":"Operation", "type":"Enumeration", "value":"ReKey"},
{"tag":"Operation", "type":"Enumeration", "value":"Archive"},
{"tag":"Operation", "type":"Enumeration", "value":"Recover"},
{"tag":"Operation", "type":"Enumeration", "value":"ObtainLease"},
{"tag":"Operation", "type":"Enumeration", "value":"ReKeyKeyPair"},
{"tag":"Operation", "type":"Enumeration", "value":"Certify"},
{"tag":"Operation", "type":"Enumeration", "value":"ReCertify"},
{"tag":"Operation", "type":"Enumeration", "value":"DiscoverVersions"},
{"tag":"Operation", "type":"Enumeration", "value":"Notify"},
{"tag":"Operation", "type":"Enumeration", "value":"Put"},
{"tag":"Operation", "type":"Enumeration", "value":"RNGRetrieve"},
{"tag":"Operation", "type":"Enumeration", "value":"RNGSeed"},
{"tag":"Operation", "type":"Enumeration", "value":"Encrypt"},
{"tag":"Operation", "type":"Enumeration", "value":"Decrypt"},
{"tag":"Operation", "type":"Enumeration", "value":"Sign"},
{"tag":"Operation", "type":"Enumeration", "value":"SignatureVerify"},
{"tag":"Operation", "type":"Enumeration", "value":"MAC"},
{"tag":"Operation", "type":"Enumeration", "value":"MACVerify"},
{"tag":"Operation", "type":"Enumeration", "value":"Hash"},
{"tag":"Operation", "type":"Enumeration", "value":"CreateSplitKey"},
{"tag":"Operation", "type":"Enumeration", "value":"JoinSplitKey"},
{"tag":"ObjectType", "type":"Enumeration", "value":"Certificate"},
{"tag":"ObjectType", "type":"Enumeration", "value":"SymmetricKey"},
{"tag":"ObjectType", "type":"Enumeration", "value":"SecretData"},
{"tag":"ObjectType", "type":"Enumeration", "value":"PublicKey"},
{"tag":"ObjectType", "type":"Enumeration", "value":"PrivateKey"},
{"tag":"ObjectType", "type":"Enumeration", "value":"Template"},
{"tag":"ObjectType", "type":"Enumeration", "value":"OpaqueObject"},
{"tag":"ObjectType", "type":"Enumeration", "value":"SplitKey"},
{"tag":"ObjectType", "type":"Enumeration", "value":"PGPKey"}
]}
]}
]]}

```

344 6 XML Profile

345 The XML profile specifies the use of KMIP replacing the TTLV message encoding with an XML message
346 encoding. The results returned using the XML encoding SHALL be logically the same as if the message
347 encoding was in TTLV form. All size or length values specified within tag values for KMIP items SHALL be
348 the same in XML form as if the message encoding were in TTLV form. The implications of this are that
349 items such as MaximumResponseSize are interpreted to refer to a maximum length computed as if it
350 were a TTLV-encoded response, not the length of the JSON-encoded response.

351 6.1 XML Encoding

352 6.1.1 Hex representations

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

360 6.1.2 Tags

361 Tags are a String that may contain either:

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

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

367 6.1.3 Normalizing Names

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

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

- 375 7. Replace round brackets ('(', ')') with spaces
- 376 8. If a non-word char (not alpha, digit or underscore) is followed by a letter (either upper or lower
377 case) then a lower case letter, replace the non-word char with space
- 378 9. Replace remaining non-word chars (except whitespace) with underscore.
- 379 10. If the first word begins with a digit, move all digits at start of first word to end of first word
- 380 11. Capitalize the first letter of each word
- 381 12. Concatenate all words with spaces removed
- 382


```

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

```

395 *Example python name normalization code*

396

```

397 # 1. Replace brackets with space
398 $enumName=~s/[\\(\\)]/ /g;
399 # 2. replace \W with space if followed by letter, lower
400 $enumName=~s/\W([A-Za-z][a-z])/ \1/g;
401 # 3. non-word to underscore
402 @words=split(/ /,$enumName);
403 for($i=0;$i<=$#words;$i++) { $words[$i]=~s/\W/_/g; }
404 # 4. move numbers to end of first word
405 $words[0] =~ s/^\d+ (.*)/\2\1/;
406 # 5. captialize first letter of each word
407 for($i=0;$i<=$#words;$i++) {
408     substr($words[$i],0,1)=~tr/a-z/A-Z/;
409 }
410 # 6. concatenate
411 $enumNameCamel = join(' ',@words);
412

```

413 *Example perl name normalization code*

414 6.1.4 Type

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

- 417 • Structure
- 418 • Integer
- 419 • LongInteger
- 420 • BigInteger
- 421 • Enumeration
- 422 • Boolean
- 423 • TextString
- 424 • ByteString
- 425 • DateTime
- 426 • Interval

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

428 6.1.5 Value

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

430 6.1.6 XML Element Encoding

431 For XML, each TTLV is represented as an XML element with attributes. The general form uses a single
432 element named 'TTLV' with 'tag', optional 'name' and 'type' attributes. This form allows any TTLV
433 including extensions to be encoded. For tags defined in the KMIP Specification or other well-known
434 extensions, a more specific form can be used where each tag is encoded as an element with the same
435 name and includes a 'type' attribute. For either form, structure values are encoded as nested xml
436 elements, and non-structure values are encoded using the 'value' attribute.

```
437  
438 <TTLV tag="0x420001" name="ActivationDate" type="DateTime" value="2001-01-01T10:00:00+10:00"/>  
439 <TTLV tag="0x420001" type="DateTime" value="2001-01-01T10:00:00+10:00"/>  
440 <ActivationDate type="DateTime" value="2001-01-01T10:00:00+10:00"/>  
441 <TTLV tag="0x54FFFF" name="SomeExtension" type="DateTime" value="2001-01-01T10:00:00+10:00"/>  
442
```

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

445 If namespaces are required, XML elements SHALL use the following namespace:

```
446 urn:oasis:tc:kmip:xmlns
```

447 6.1.6.1 Tags

448 Tags are a String that may contain either:

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

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

```
454 <ActivationDate xmlns="urn:oasis:tc:kmip:xmlns" type="DateTime" value="2001-01-  
455 01T10:00:00+10:00"/>  
456 <IVCounterNonce type="ByteString" value="alb2c3d4"/>  
457 <PrivateKeyTemplateAttribute type="Structure"/>  
458 <TTLV tag="0x545352" name="SomeExtension" type="TextString" value="This is an extension"/>  
459 <WELL_KNOWN_EXTENSION type="TextString" value="This is an extension"/>
```

460 6.1.6.2 Structure

461 For XML, sub-items are nested elements.

```
462 <ProtocolVersion type="Structure">  
463   <ProtocolVersionMajor type="Integer" value="1"/>  
464   <ProtocolVersionMinor type="Integer" value="0"/>  
465 </ProtocolVersion>  
466 <ProtocolVersion>  
467   <ProtocolVersionMajor type="Integer" value="1"/>  
468   <ProtocolVersionMinor type="Integer" value="0"/>  
469 </ProtocolVersion>
```

470
471 The 'type' property / attribute is optional for a Structure.

472 6.1.6.3 Integer

473 For XML, value is a decimal and uses [XML-SCHEMA] type xsd:int

```
474  
475 <BatchCount type="Integer" value="10"/>
```

476 6.1.6.4 Integer - Special case for Masks

477 (Cryptographic Usage Mask, Storage Status Mask):

478 Integer mask values can also be encoded as a String containing mask components. XML uses an
479 attribute with [XML-SCHEMA] type xsd:list which uses a space separator. Components may be either
480 the text of the enumeration value as defined in [KMIP 9.1.3.3.1](#) / [KMIP 9.1.3.3.2](#), or a 32-bit unsigned big-
481 endian hex string.

```
482 <CryptographicUsageMask type="Integer" value="0x0000100c"/>  
483 <CryptographicUsageMask type="Integer" value="Encrypt Decrypt CertificateSign"/>  
484 <CryptographicUsageMask type="Integer" value="CertificateSign 0x00000004 0x00000008"/>  
485 <CryptographicUsageMask type="Integer" value="CertificateSign 0x0000000c"/>
```

486 6.1.6.5 Long Integer

487 For XML, value uses [XML-SCHEMA] type xsd:long

```
488 <x540001 type="LongInteger" value="-2"/>  
489 <UsageLimitsCount type="LongInteger" value="1152921504606846976"/>
```

490 6.1.6.6 Big Integer

491 For XML, value uses [XML-SCHEMA] type xsd:hexBinary

```
492 <X type="BigInteger" value="0000000000000000"/>
```

493 6.1.6.7 Enumeration

494 For XML, value uses [XML-SCHEMA] type xsd:string and is either a hex string or the CamelCase enum
495 text. If an XSD with xsd:enumeration restriction is used to define valid values (as is the case with the
496 XSD included as an appendix), parsers should also accept any hex string in addition to defined enum
497 values.

```
498 <ObjectType type="Enumeration" value="0x00000002"/>  
499 <ObjectType type="Enumeration" value="SymmetricKey"/>
```

500 6.1.6.8 Boolean

501 For XML, value uses [XML-SCHEMA] type xsd:Boolean

```
502 <BatchOrderOption type="Boolean" value="true"/>
```

503 6.1.6.9 Text String

504 XML uses [XML-SCHEMA] type xsd:string

```
505 <AttributeName type="TextString" value="Cryptographic Algorithm"/>
```

506 6.1.6.10 Byte String

507 XML uses [XML-SCHEMA] type xsd:hexBinary

```
508 <MACSignature type="ByteString" value="C50F77"/>
```

509 6.1.6.11 Date-Time

510 For XML, value uses [XML-SCHEMA] type xsd:dateTime

```
511 <ArchiveDate type="DateTime" value="2001-01-01T10:00:00+10:00"/>
```

512 6.1.6.12 Interval

513 XML uses [XML-SCHEMA] type xsd:unsignedInt

```
514 <Offset type="Interval" value="27"/>
```

515

516

517 7 XML Profile Test Cases

518 The test cases define a number of request-response pairs for KMIP operations. Each test case is
519 provided in the XML format specified in this section intended to be both human-readable and usable by
520 automated tools. The time sequence (starting from 0) for each request-response pair is noted and line
521 numbers are provided for ease of cross-reference for a given test sequence.

522 Each test case has a unique label (the section name) which includes indication of mandatory (-M-) or
523 optional (-O-) status and the protocol version major and minor numbers as part of the identifier.

524 The test cases may depend on a specific configuration of a KMIP client and server being configured in a
525 manner consistent with the test case assumptions.

526 Where possible the flow of unique identifiers between tests, the date-time values, and other dynamic
527 items are indicated using symbolic identifiers – in actual request and response messages these dynamic
528 values will be filled in with valid values.

529 Note: the values for the returned items and the custom attributes are illustrative. Actual values from a real
530 client system may vary as specified in section 8.4

531 7.1 Mandatory XML Profile Test Cases KMIP v1.0

532 7.1.1 MSGENC-XML-M-1-10 - Query, Maximum Response Size

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

537 The specific list of operations and object types returned in the response MAY vary.

0001	<code><RequestMessage></code>
0002	<code> <RequestHeader></code>
0003	<code> <ProtocolVersion></code>
0004	<code> <ProtocolVersionMajor type="Integer" value="1"/></code>
0005	<code> <ProtocolVersionMinor type="Integer" value="0"/></code>
0006	<code> </ProtocolVersion></code>
0007	<code> <MaximumResponseSize type="Integer" value="256"/></code>
0008	<code> <BatchCount type="Integer" value="1"/></code>
0009	<code> </RequestHeader></code>
0010	<code> <BatchItem></code>
0011	<code> <Operation type="Enumeration" value="Query"/></code>
0012	<code> <RequestPayload></code>
0013	<code> <QueryFunction type="Enumeration" value="QueryOperations"/></code>
0014	<code> <QueryFunction type="Enumeration" value="QueryObjects"/></code>
0015	<code> </RequestPayload></code>
0016	<code> </BatchItem></code>
0017	<code></RequestMessage></code>
	 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b0200000004000000000000000042005002000000040000001000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000
0018	<code><ResponseMessage></code>
0019	<code> <ResponseHeader></code>
0020	<code> <ProtocolVersion></code>
0021	<code> <ProtocolVersionMajor type="Integer" value="1"/></code>

0022 0023 0024 0025 0026 0027 0028 0029 0030 0031 0032	<pre> <ProtocolVersionMinor type="Integer" value="0"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2013-06-26T09:09:17+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="OperationFailed"/> <ResultReason type="Enumeration" value="ResponseTooLarge"/> <ResultMessage type="TextString" value="TOO_LARGE"/> </BatchItem> </ResponseMessage> 42007b01000000a042007a0100000048420069010000002042006a02000000040000000100000000 42006b0200000004000000000000000042009209000000080000000051caafb42000d0200000004 000000010000000042000f010000004842005c0500000004000000180000000042007f0500000004 000000010000000042007e0500000004000000020000000042007d0700000009544f4f5f4c415247 4500000000000000 </pre>
0032 0033 0034 0035 0036 0037 0038 0039 0040 0041 0042 0043 0044 0045 0046 0047 0048	<pre> # TIME 1 <RequestMessage> <RequestHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="0"/> </ProtocolVersion> <MaximumResponseSize type="Integer" value="2048"/> <BatchCount type="Integer" value="1"/> </RequestHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <RequestPayload> <QueryFunction type="Enumeration" value="QueryOperations"/> <QueryFunction type="Enumeration" value="QueryObjects"/> </RequestPayload> </BatchItem> </RequestMessage> 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b020000000400000000000000004200500200000004000008000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000 </pre>
0049 0050 0051 0052 0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="0"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2013-06-26T09:09:17+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="Success"/> <ResponsePayload> <Operation type="Enumeration" value="Query"/> <Operation type="Enumeration" value="Locate"/> <Operation type="Enumeration" value="Destroy"/> <Operation type="Enumeration" value="Get"/> <Operation type="Enumeration" value="Create"/> </ResponsePayload> </BatchItem> </ResponseMessage> </pre>

```

0067 <Operation type="Enumeration" value="Register"/>
0068 <Operation type="Enumeration" value="GetAttributes"/>
0069 <Operation type="Enumeration" value="GetAttributeList"/>
0070 <Operation type="Enumeration" value="AddAttribute"/>
0071 <Operation type="Enumeration" value="ModifyAttribute"/>
0072 <Operation type="Enumeration" value="DeleteAttribute"/>
0073 <Operation type="Enumeration" value="Activate"/>
0074 <Operation type="Enumeration" value="Revoke"/>
0075 <Operation type="Enumeration" value="Poll"/>
0076 <Operation type="Enumeration" value="Cancel"/>
0077 <Operation type="Enumeration" value="Check"/>
0078 <Operation type="Enumeration" value="GetUsageAllocation"/>
0079 <Operation type="Enumeration" value="CreateKeyPair"/>
0080 <Operation type="Enumeration" value="ReKey"/>
0081 <Operation type="Enumeration" value="Archive"/>
0082 <Operation type="Enumeration" value="Recover"/>
0083 <Operation type="Enumeration" value="ObtainLease"/>
0084 <Operation type="Enumeration" value="Certify"/>
0085 <Operation type="Enumeration" value="ReCertify"/>
0086 <Operation type="Enumeration" value="Notify"/>
0087 <Operation type="Enumeration" value="Put"/>
0088 <ObjectType type="Enumeration" value="Certificate"/>
0089 <ObjectType type="Enumeration" value="SymmetricKey"/>
0090 <ObjectType type="Enumeration" value="SecretData"/>
0091 <ObjectType type="Enumeration" value="PublicKey"/>
0092 <ObjectType type="Enumeration" value="PrivateKey"/>
0093 <ObjectType type="Enumeration" value="Template"/>
0094 <ObjectType type="Enumeration" value="OpaqueObject"/>
0095 <ObjectType type="Enumeration" value="SplitKey"/>
0096 </ResponsePayload>
0097 </BatchItem>
0098 </ResponseMessage>

42007b01000002a042007a0100000048420069010000002042006a02000000040000000100000000
42006b0200000004000000000000000042009209000000080000000051caafbd42000d0200000004
000000010000000042000f010000024842005c0500000004000000180000000042007f0500000004
000000000000000042007c010000022042005c0500000004000000180000000042005c0500000004
000000080000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c0500000004000000010000000042005c0500000004000000030000000042005c0500000004
0000000b0000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
000000120000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
000000110000000042005c050000000400000020000000042005c05000000040000000400000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c0500000004000000060000000042005c05000000040000000700000000
42005c05000000040000001b0000000042005c05000000040000001c000000004200570500000004
0000000100000000420057050000000400000020000000042005705000000040000000700000000
42005705000000040000000300000000420057050000000400000004000000004200570500000004
00000006000000004200570500000004000000080000000042005705000000040000000500000000

```

538

539 **7.2 Mandatory XML Profile Test Cases KMIP v1.1**

540 **7.2.1 MSGENC-XML-M-1-11 - Query, Maximum Response Size**

541 Perform a Query operation, querying the Operations and Objects supported by the server, with a
542 restriction on the Maximum Response Size set in the request header. Since the resulting Query response

- 543 is too big, an error is returned. Increase the Maximum Response Size, resubmit the Query request, and
- 544 get a successful response.
- 545 The specific list of operations and object types returned in the response MAY vary.

<pre> 0001 <RequestMessage> 0002 <RequestHeader> 0003 <ProtocolVersion> 0004 <ProtocolVersionMajor type="Integer" value="1"/> 0005 <ProtocolVersionMinor type="Integer" value="1"/> 0006 </ProtocolVersion> 0007 <MaximumResponseSize type="Integer" value="256"/> 0008 <BatchCount type="Integer" value="1"/> 0009 </RequestHeader> 0010 <BatchItem> 0011 <Operation type="Enumeration" value="Query"/> 0012 <RequestPayload> 0013 <QueryFunction type="Enumeration" value="QueryOperations"/> 0014 <QueryFunction type="Enumeration" value="QueryObjects"/> 0015 </RequestPayload> 0016 </BatchItem> 0017 </RequestMessage> </pre> <p>42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b020000000400000001000000004200500200000004000001000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000</p>	<pre> 0018 <ResponseMessage> 0019 <ResponseHeader> 0020 <ProtocolVersion> 0021 <ProtocolVersionMajor type="Integer" value="1"/> 0022 <ProtocolVersionMinor type="Integer" value="1"/> 0023 </ProtocolVersion> 0024 <TimeStamp type="DateTime" value="2014-06-10T08:03:34+00:00"/> 0025 <BatchCount type="Integer" value="1"/> 0026 </ResponseHeader> <BatchItem> 0027 <Operation type="Enumeration" value="Query"/> 0028 <ResultStatus type="Enumeration" value="OperationFailed"/> 0029 <ResultReason type="Enumeration" value="ResponseTooLarge"/> 0030 <ResultMessage type="TextString" value="TOO_LARGE"/> 0031 </BatchItem> 0032 </ResponseMessage> </pre> <p>42007b01000000a042007a0100000048420069010000002042006a02000000040000000100000000 42006b020000000400000001000000004200920900000008000000005396bc2442000d0200000004 000000010000000042000f010000004842005c0500000004000000180000000042007f0500000004 000000010000000042007e0500000004000000020000000042007d0700000009544f4f5f4c415247 4500000000000000</p>
<pre> 0033 # TIME 1 0034 <RequestMessage> 0035 <RequestHeader> 0036 <ProtocolVersion> 0037 <ProtocolVersionMajor type="Integer" value="1"/> 0038 <ProtocolVersionMinor type="Integer" value="1"/> 0039 </ProtocolVersion> 0040 <MaximumResponseSize type="Integer" value="2048"/> 0041 <BatchCount type="Integer" value="1"/> </pre>	

0042 0043 0044 0045 0046 0047 0048 0049	<pre> <BatchItem> <Operation type="Enumeration" value="Query"/> <RequestPayload> <QueryFunction type="Enumeration" value="QueryOperations"/> <QueryFunction type="Enumeration" value="QueryObjects"/> </RequestPayload> </BatchItem> </RequestMessage> 42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b020000000400000001000000004200500200000004000008000000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000 </pre>
0050 0051 0052 0053 0054 0055 0056 0057 0058 0059 0060 0061 0062 0063 0064 0065 0066 0067 0068 0069 0070 0071 0072 0073 0074 0075 0076 0077 0078 0079 0080 0081 0082 0083 0084 0085 0086 0087 0088 0089 0090 0091 0092 0093 0094	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="1"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2014-06-10T08:03:34+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="Success"/> <ResponsePayload> <Operation type="Enumeration" value="Query"/> <Operation type="Enumeration" value="Locate"/> <Operation type="Enumeration" value="Destroy"/> <Operation type="Enumeration" value="Get"/> <Operation type="Enumeration" value="Create"/> <Operation type="Enumeration" value="Register"/> <Operation type="Enumeration" value="GetAttributes"/> <Operation type="Enumeration" value="GetAttributeList"/> <Operation type="Enumeration" value="AddAttribute"/> <Operation type="Enumeration" value="ModifyAttribute"/> <Operation type="Enumeration" value="DeleteAttribute"/> <Operation type="Enumeration" value="Activate"/> <Operation type="Enumeration" value="Revoke"/> <Operation type="Enumeration" value="Poll"/> <Operation type="Enumeration" value="Cancel"/> <Operation type="Enumeration" value="Check"/> <Operation type="Enumeration" value="GetUsageAllocation"/> <Operation type="Enumeration" value="CreateKeyPair"/> <Operation type="Enumeration" value="ReKey"/> <Operation type="Enumeration" value="Archive"/> <Operation type="Enumeration" value="Recover"/> <Operation type="Enumeration" value="ObtainLease"/> <Operation type="Enumeration" value="ReKeyKeyPair"/> <Operation type="Enumeration" value="Certify"/> <Operation type="Enumeration" value="ReCertify"/> <Operation type="Enumeration" value="DiscoverVersions"/> <Operation type="Enumeration" value="Notify"/> <Operation type="Enumeration" value="Put"/> <ObjectType type="Enumeration" value="Certificate"/> <ObjectType type="Enumeration" value="SymmetricKey"/> <ObjectType type="Enumeration" value="SecretData"/> <ObjectType type="Enumeration" value="PublicKey"/> </ResponsePayload> </BatchItem> </ResponseMessage> </pre>


```

0095     <ObjectType type="Enumeration" value="PrivateKey"/>
0096     <ObjectType type="Enumeration" value="Template"/>
0097     <ObjectType type="Enumeration" value="OpaqueObject"/>
0098     <ObjectType type="Enumeration" value="SplitKey"/>
0099     </ResponsePayload>
0100 </BatchItem>
0101 </ResponseMessage>

42007b01000002c042007a0100000048420069010000002042006a02000000040000000100000000
42006b0200000004000000001000000004200920900000008000000005396bc2442000d0200000004
000000010000000042000f010000026842005c0500000004000000180000000042007f0500000004
000000000000000042007c010000024042005c0500000004000000180000000042005c0500000004
000000080000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c0500000004000000010000000042005c0500000004000000030000000042005c0500000004
0000000b0000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
000000120000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
000000110000000042005c0500000004000000020000000042005c05000000040000000400000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c05000000040000001d0000000042005c05000000040000000600000000
42005c0500000004000000070000000042005c05000000040000001e0000000042005c0500000004
0000001b0000000042005c05000000040000001c0000000042005705000000040000000100000000
42005705000000040000000200000000420057050000000400000007000000004200570500000004
00000003000000004200570500000004000000040000000042005705000000040000000600000000
4200570500000004000000080000000042005705000000040000000500000000

```

546

547 7.3 Mandatory XML Profile Test Cases KMIP v1.2

548 7.3.1 MSGENC-XML-M-1-12 - Query, Maximum Response Size

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

553 The specific list of operations and object types returned in the response MAY vary.

```

# TIME 0
0001 <RequestMessage>
0002   <RequestHeader>
0003     <ProtocolVersion>
0004       <ProtocolVersionMajor type="Integer" value="1"/>
0005       <ProtocolVersionMinor type="Integer" value="2"/>
0006     </ProtocolVersion>
0007     <MaximumResponseSize type="Integer" value="256"/>
0008     <BatchCount type="Integer" value="1"/>
0009   </RequestHeader>
0010   <BatchItem>
0011     <Operation type="Enumeration" value="Query"/>
0012     <RequestPayload>
0013       <QueryFunction type="Enumeration" value="QueryOperations"/>
0014       <QueryFunction type="Enumeration" value="QueryObjects"/>
0015     </RequestPayload>
0016   </BatchItem>
0017 </RequestMessage>

42007801000000904200770100000048420069010000002042006a02000000040000000100000000
42006b02000000040000000200000000420050020000000400000100000000042000d0200000004
000000010000000042000f01000003842005c050000000400000018000000004200790100000020
4200740500000004000000010000000042007405000000040000000200000000

```

<pre> 0018 0019 0020 0021 0022 0023 0024 0025 0026 0027 0028 0029 0030 0031 0032 </pre>	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="2"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2014-06-10T08:07:28+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="OperationFailed"/> <ResultReason type="Enumeration" value="ResponseTooLarge"/> <ResultMessage type="TextString" value="TOO_LARGE"/> </BatchItem> </ResponseMessage> </pre> <p>42007b01000000a042007a0100000048420069010000002042006a02000000040000000100000000 42006b0200000004000000002000000004200920900000008000000005396bcc042000d0200000004 000000010000000042000f010000004842005c0500000004000000180000000042007f0500000004 000000010000000042007e0500000004000000020000000042007d0700000009544f4f5f4c415247 4500000000000000</p>
<pre> 0033 0034 0035 0036 0037 0038 0039 0040 0041 0042 0043 0044 0045 0046 0047 0048 0049 </pre>	<pre> # TIME 1 <RequestMessage> <RequestHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="2"/> </ProtocolVersion> <MaximumResponseSize type="Integer" value="2048"/> <BatchCount type="Integer" value="1"/> </RequestHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <RequestPayload> <QueryFunction type="Enumeration" value="QueryOperations"/> <QueryFunction type="Enumeration" value="QueryObjects"/> </RequestPayload> </BatchItem> </RequestMessage> </pre> <p>42007801000000904200770100000048420069010000002042006a02000000040000000100000000 42006b020000000400000000200000000420050020000000400000800000000042000d0200000004 000000010000000042000f010000003842005c050000000400000018000000004200790100000020 4200740500000004000000010000000042007405000000040000000200000000</p>
<pre> 0050 0051 0052 0053 0054 0055 0056 0057 0058 0059 0060 0061 </pre>	<pre> <ResponseMessage> <ResponseHeader> <ProtocolVersion> <ProtocolVersionMajor type="Integer" value="1"/> <ProtocolVersionMinor type="Integer" value="2"/> </ProtocolVersion> <TimeStamp type="DateTime" value="2014-06-10T08:07:28+00:00"/> <BatchCount type="Integer" value="1"/> </ResponseHeader> <BatchItem> <Operation type="Enumeration" value="Query"/> <ResultStatus type="Enumeration" value="Success"/> </BatchItem> </ResponseMessage> </pre>

```
0062 <ResponsePayload>
0063 <Operation type="Enumeration" value="Query"/>
0064 <Operation type="Enumeration" value="Locate"/>
0065 <Operation type="Enumeration" value="Destroy"/>
0066 <Operation type="Enumeration" value="Get"/>
0067 <Operation type="Enumeration" value="Create"/>
0068 <Operation type="Enumeration" value="Register"/>
0069 <Operation type="Enumeration" value="GetAttributes"/>
0070 <Operation type="Enumeration" value="GetAttributeList"/>
0071 <Operation type="Enumeration" value="AddAttribute"/>
0072 <Operation type="Enumeration" value="ModifyAttribute"/>
0073 <Operation type="Enumeration" value="DeleteAttribute"/>
0074 <Operation type="Enumeration" value="Activate"/>
0075 <Operation type="Enumeration" value="Revoke"/>
0076 <Operation type="Enumeration" value="Poll"/>
0077 <Operation type="Enumeration" value="Cancel"/>
0078 <Operation type="Enumeration" value="Check"/>
0079 <Operation type="Enumeration" value="GetUsageAllocation"/>
0080 <Operation type="Enumeration" value="CreateKeyPair"/>
0081 <Operation type="Enumeration" value="ReKey"/>
0082 <Operation type="Enumeration" value="Archive"/>
0083 <Operation type="Enumeration" value="Recover"/>
0084 <Operation type="Enumeration" value="ObtainLease"/>
0085 <Operation type="Enumeration" value="ReKeyKeyPair"/>
0086 <Operation type="Enumeration" value="Certify"/>
0087 <Operation type="Enumeration" value="ReCertify"/>
0088 <Operation type="Enumeration" value="DiscoverVersions"/>
0089 <Operation type="Enumeration" value="Notify"/>
0090 <Operation type="Enumeration" value="Put"/>
0091 <Operation type="Enumeration" value="RNGRetrieve"/>
0092 <Operation type="Enumeration" value="RNGSeed"/>
0093 <Operation type="Enumeration" value="Encrypt"/>
0094 <Operation type="Enumeration" value="Decrypt"/>
0095 <Operation type="Enumeration" value="Sign"/>
0096 <Operation type="Enumeration" value="SignatureVerify"/>
0097 <Operation type="Enumeration" value="MAC"/>
0098 <Operation type="Enumeration" value="MACVerify"/>
0099 <Operation type="Enumeration" value="Hash"/>
0100 <Operation type="Enumeration" value="CreateSplitKey"/>
0101 <Operation type="Enumeration" value="JoinSplitKey"/>
0102 <ObjectType type="Enumeration" value="Certificate"/>
0103 <ObjectType type="Enumeration" value="SymmetricKey"/>
0104 <ObjectType type="Enumeration" value="SecretData"/>
0105 <ObjectType type="Enumeration" value="PublicKey"/>
0106 <ObjectType type="Enumeration" value="PrivateKey"/>
0107 <ObjectType type="Enumeration" value="Template"/>
0108 <ObjectType type="Enumeration" value="OpaqueObject"/>
0109 <ObjectType type="Enumeration" value="SplitKey"/>
0110 <ObjectType type="Enumeration" value="PGPKey"/>
0111 </ResponsePayload>
0112 </BatchItem>
0113 </ResponseMessage>

42007b010000038042007a0100000048420069010000002042006a02000000040000000100000000
42006b0200000004000000002000000004200920900000008000000005396bcc042000d0200000004
000000010000000042000f010000032842005c0500000004000000180000000042007f0500000004
000000000000000042007c010000030042005c0500000004000000180000000042005c0500000004
000000080000000042005c0500000004000000140000000042005c05000000040000000a00000000
42005c0500000004000000010000000042005c0500000004000000030000000042005c0500000004
```

0000000b0000000042005c05000000040000000c0000000042005c05000000040000000d00000000
42005c05000000040000000e0000000042005c05000000040000000f0000000042005c0500000004
000000120000000042005c0500000004000000130000000042005c05000000040000001a00000000
42005c0500000004000000190000000042005c0500000004000000090000000042005c0500000004
000000110000000042005c0500000004000000020000000042005c05000000040000000400000000
42005c0500000004000000150000000042005c0500000004000000160000000042005c0500000004
000000100000000042005c05000000040000001d0000000042005c05000000040000000600000000
42005c0500000004000000070000000042005c05000000040000001e0000000042005c0500000004
0000001b0000000042005c05000000040000001c0000000042005c05000000040000002500000000
42005c0500000004000000260000000042005c05000000040000001f0000000042005c0500000004
000000200000000042005c0500000004000000210000000042005c05000000040000002200000000
42005c0500000004000000230000000042005c0500000004000000240000000042005c0500000004
000000270000000042005c0500000004000000280000000042005c05000000040000002900000000
42005705000000040000000100000000420057050000000400000002000000004200570500000004
00000007000000004200570500000004000000030000000042005705000000040000000400000000
42005705000000040000000600000000420057050000000400000008000000004200570500000004
000000050000000042005705000000040000000900000000

555 8 Conformance

556 8.1 HTTPS Profile

557 8.1.1 HTTPS Client KMIP v1.0 Profile Conformance

558 KMIP client implementations conformant to this profile:

- 559 1. SHALL support the Authentication Suite conditions as specified in Section 2.1 of this profile.
- 560 2. SHALL support the KMIP Port Number conditions as specified in Section 2.2 of this profile.
- 561 3. SHALL support the Request URL conditions as specified in Section 2.3 of this profile.
- 562 4. SHALL support the HTTP Encoding conditions as specified in Section 2.4 of this profile.
- 563 5. SHALL support all the Mandatory HTTPS Profile Test Cases KMIP v1.0 (3.1)

564 8.1.2 HTTPS Client KMIP v1.1 Profile Conformance

565 KMIP client implementations conformant to this profile:

- 566 1. SHALL support the Authentication Suite conditions as specified in Section 2.1 of this profile.
- 567 2. SHALL support the KMIP Port Number conditions as specified in Section 2.2 of this profile.
- 568 3. SHALL support the Request URL conditions as specified in Section 2.3 of this profile.
- 569 4. SHALL support the HTTP Encoding conditions as specified in Section 2.4 of this profile.
- 570 5. SHALL support all the Mandatory HTTPS Profile Test Cases KMIP v1.1 (3.2)

571 8.1.3 HTTPS Client KMIP v1.2 Profile Conformance

572 KMIP client implementations conformant to this profile:

- 573 1. SHALL support the Authentication Suite conditions as specified in Section 2.1 of this profile.
- 574 2. SHALL support the KMIP Port Number conditions as specified in Section 2.2 of this profile.
- 575 3. SHALL support the Request URL conditions as specified in Section 2.3 of this profile.
- 576 4. SHALL support the HTTP Encoding conditions as specified in Section 2.4 of this profile.
- 577 5. SHALL support all the Mandatory HTTPS Profile Test Cases KMIP v1.2 (3.3)

578 8.1.4 HTTPS Server KMIP v1.0 Profile Conformance

579 KMIP server implementations conformant to this profile:

- 580 1. SHALL support the Authentication Suite conditions as specified in Section 2.1 of this profile.
- 581 2. SHALL support the KMIP Port Number conditions as specified in Section 2.2 of this profile.
- 582 3. SHALL support the Request URL conditions as specified in Section 2.3 of this profile.
- 583 4. SHALL support the HTTP Encoding conditions as specified in Section 2.5 of this profile.
- 584 5. SHALL support all the Mandatory HTTPS Profile Test Cases KMIP v1.0 (3.1)

585 8.1.5 HTTPS Server KMIP v1.1 Profile Conformance

586 KMIP server implementations conformant to this profile:

- 587 1. SHALL support the Authentication Suite conditions as specified in Section 2.1 of this profile.
- 588 2. SHALL support the KMIP Port Number conditions as specified in Section 2.2 of this profile.
- 589 3. SHALL support the Request URL conditions as specified in Section 2.3 of this profile.

- 590 4. SHALL support the HTTP Encoding conditions as specified in Section 2.5 of this profile.
591 5. Mandatory HTTPS Profile Test Cases KMIP v1.1 (3.2)

592 **8.1.6 HTTPS Server KMIP v1.2 Profile Conformance**

593 KMIP server implementations conformant to this profile:

- 594 1. SHALL support the Authentication Suite conditions as specified in Section 2.1 of this profile.
595 2. SHALL support the KMIP Port Number conditions as specified in Section 2.2 of this profile.
596 3. SHALL support the Request URL conditions as specified in Section 2.3 of this profile.
597 4. SHALL support the HTTP Encoding conditions as specified in Section 2.5 of this profile.
598 5. SHALL support all the Mandatory HTTPS Profile Test Cases KMIP v1.2 (3.3)

599 **8.2 JSON Profile**

600 **8.2.1 JSON Client KMIP v1.0 Profile Conformance**

601 KMIP client implementations conformant to this profile:

- 602 1. SHALL support JSON message encoding for request and response messages as specified in
603 Section 4.1 of this profile.
604 2. SHALL support all the Mandatory JSON Profile Test Cases KMIP v1.0 (5.1)
605 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
606 [KMIP-SPEC] that is supported
607 4. SHALL support user defined extensions containing additional tags and enumerations not
608 specified within [KMIP-SPEC]

609 **8.2.2 JSON Client KMIP v1.1 Profile Conformance**

610 KMIP client implementations conformant to this profile:

- 611 1. SHALL support JSON message encoding for request and response messages as specified in
612 Section 4.1 of this profile.
613 2. SHALL support all the Mandatory JSON Profile Test Cases KMIP v1.1 (5.2)
614 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
615 [KMIP-SPEC] that is supported
616 4. SHALL support user defined extensions containing additional tags and enumerations not
617 specified within [KMIP-SPEC]

618 **8.2.3 JSON Client KMIP v1.2 Profile Conformance**

619 KMIP client implementations conformant to this profile:

- 620 1. SHALL support JSON message encoding for request and response messages as specified in
621 Section 4.1 of this profile.
622 2. SHALL support all the Mandatory JSON Profile Test Cases KMIP v1.2(5.3)
623 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
624 [KMIP-SPEC] that is supported
625 4. SHALL support user defined extensions containing additional tags and enumerations not
626 specified within [KMIP-SPEC]

627 **8.2.4 JSON Server KMIP v1.0 Profile Conformance**

628 KMIP server implementations conformant to this profile:

- 629 1. SHALL support JSON message encoding for request and response messages as specified in
630 Section 4.1 of this profile.
- 631 2. SHALL support all the Mandatory JSON Profile Test Cases KMIP v1.0 (5.1)
- 632 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
633 [KMIP-SPEC] that is supported
- 634 4. SHALL support user defined extensions containing additional tags and enumerations not
635 specified within [KMIP-SPEC]

636 8.2.5 JSON Server KMIP v1.1 Profile Conformance

637 KMIP server implementations conformant to this profile:

- 638 1. SHALL support JSON message encoding for request and response messages as specified in
639 Section 4.1 of this profile.
- 640 2. SHALL support all the Mandatory JSON Profile Test Cases KMIP v1.1 (5.2)
- 641 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
642 [KMIP-SPEC] that is supported
- 643 4. SHALL support user defined extensions containing additional tags and enumerations not
644 specified within [KMIP-SPEC]

645 8.2.6 JSON Server KMIP v1.2 Profile Conformance

646 KMIP server implementations conformant to this profile:

- 647 1. SHALL support JSON message encoding for request and response messages as specified in
648 Section 4.1 of this profile.
- 649 2. SHALL support all the Mandatory JSON Profile Test Cases KMIP v1.2(5.3)
- 650 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
651 [KMIP-SPEC] that is supported
- 652 4. SHALL support user defined extensions containing additional tags and enumerations not
653 specified within [KMIP-SPEC]

654 8.3 XML Profile

655 8.3.1 XML Client KMIP v1.0 Profile Conformance

656 KMIP client implementations conformant to this profile:

- 657 1. SHALL support XML message encoding for request and response messages as specified in
658 Section 6.1 of this profile.
- 659 2. SHALL support all the Mandatory XML Profile Test Cases KMIP v1.0(7.1)
- 660 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
661 [KMIP-SPEC] that is supported.
- 662 4. SHALL support user defined extensions containing additional tags and enumerations not
663 specified within [KMIP-SPEC].

664 8.3.2 XML Client KMIP v1.1 Profile Conformance

665 KMIP client implementations conformant to this profile:

- 666 1. SHALL support XML message encoding for request and response messages as specified in
667 Section 6.1 of this profile.
- 668 2. SHALL support all the Mandatory XML Profile Test Cases KMIP v1.1(7.2)
- 669 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
670 [KMIP-SPEC] that is supported.

- 671 4. SHALL support user defined extensions containing additional tags and enumerations not
672 specified within [KMIP-SPEC].

673 **8.3.3 XML Client KMIP v1.2 Profile Conformance**

674 KMIP client implementations conformant to this profile:

- 675 1. SHALL support XML message encoding for request and response messages as specified in
676 Section 6.1 of this profile.
- 677 2. SHALL support all the Mandatory XML Profile Test Cases KMIP v1.2(7.3)
- 678 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
679 [KMIP-SPEC] that is supported.
- 680 4. SHALL support user defined extensions containing additional tags and enumerations not
681 specified within [KMIP-SPEC].

682 **8.3.4 XML Server KMIP v1.0 Profile Conformance**

683 KMIP server implementations conformant to this profile:

- 684 1. SHALL support XML message encoding for request and response messages as specified in
685 Section 6.1 of this profile.
- 686 2. SHALL support all the Mandatory XML Profile Test Cases KMIP v1.0(7.1)
- 687 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
688 [KMIP-SPEC] that is supported.
- 689 4. SHALL support user defined extensions containing additional tags and enumerations not
690 specified within [KMIP-SPEC].

691 **8.3.5 XML Server KMIP v1.1 Profile Conformance**

692 KMIP server implementations conformant to this profile:

- 693 1. SHALL support XML message encoding for request and response messages as specified in
694 Section 6.1 of this profile.
- 695 2. SHALL support all the Mandatory XML Profile Test Cases KMIP v1.1(7.2)
- 696 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
697 [KMIP-SPEC] that is supported.
- 698 4. SHALL support user defined extensions containing additional tags and enumerations not
699 specified within [KMIP-SPEC].

700 **8.3.6 XML Server KMIP v1.2 Profile Conformance**

701 KMIP server implementations conformant to this profile:

- 702 1. SHALL support XML message encoding for request and response messages as specified in
703 Section 6.1 of this profile.
- 704 2. SHALL support all the Mandatory XML Profile Test Cases KMIP v1.2(7.3)
- 705 3. SHALL support mapping of all TTLV tags and enumerations specified within each version of the
706 [KMIP-SPEC] that is supported.
- 707 4. SHALL support user defined extensions containing additional tags and enumerations not
708 specified within [KMIP-SPEC].

709 **8.4 Permitted Test Case Variations**

710 Whilst the test cases provided in this Profile define the allowed request and response content, some
711 inherent variations MAY occur and are permitted within a successfully completed test case.

712 Each test case MAY include allowed variations in the description of the test case in addition to the
713 variations noted in this section.

714 Other variations not explicitly noted in this Profile SHALL be deemed non-conformant.

715 8.4.1 Variable Items

716 An implementation conformant to this Profile MAY vary the following values:

- 717 1. UniqueIdentifier
- 718 2. PrivateKeyUniqueIdentifier
- 719 3. PublicKeyUniqueIdentifier
- 720 4. UniqueBatchItemIdentifier
- 721 5. AsynchronousCorrelationValue
- 722 6. TimeStamp
- 723 7. KeyValue / KeyMaterial including:
 - 724 a. key material content returned for managed cryptographic objects which are generated by
725 the server
 - 726 b. wrapped versions of keys where the wrapping key is dynamic or the wrapping contains
727 variable output for each wrap operation
- 728 8. For response containing the output of cryptographic operation in Data / SignatureData/ MACData
729 / IVCounterNonce where:
 - 730 a. the managed object is generated by the server; or
 - 731 b. the operation inherently contains variable output
- 732 9. For the following DateTime attributes where the value is not specified in the request as a fixed
733 DateTime value:
 - 734 a. ActivationDate
 - 735 b. ArchiveDate
 - 736 c. CompromiseDate
 - 737 d. CompromiseOccurrenceDate
 - 738 e. DeactivationDate
 - 739 f. DestroyDate
 - 740 g. InitialDate
 - 741 h. LastChangeDate
 - 742 i. ProtectStartDate
 - 743 j. ProcessStopDate
 - 744 k. ValidityDate
 - 745 l. OriginalCreationDate
- 746 10. LinkedObjectIdentifier
- 747 11. DigestValue
 - 748 a. For those managed cryptographic objects which are dynamically generated
- 749 12. KeyFormatType
 - 750 a. The key format type selected by the server when it creates managed objects
- 751 13. Digest
 - 752 a. The HashingAlgorithm selected by the server when it calculates the digest for a managed
753 object for which it has access to the key material
 - 754 b. The Digest Value

- 755 14. Extensions reported in Query for ExtensionList and ExtensionMap
756 15. Application Namespaces reported in Query
757 16. Object Types reported in Query other than those noted as required in this profile
758 17. Operation Types reported in Query other than those noted as required in this profile (or any
759 referenced profile documents)
760 18. For TextString attribute values containing test identifiers:
761 a. Additional vendor or application prefixes
762 19. Additional attributes beyond those noted in the response
763

764 An implementation conformant to this Profile MAY allow the following response variations:

- 765 20. Object Group values – May or may not return one or more Object Group values not included in
766 the requests
767 21. y-CustomAttributes – May or may not include additional server-specific associated attributes not
768 included in requests
769 22. Message Extensions – May or may not include additional (non-critical) vendor extensions
770 23. TemplateAttribute – May or may not be included in responses where the Template Attribute
771 response is noted as optional in [KMIP-SPEC]
772 24. AttributeIndex – May or may not include Attribute Index value where the Attribute Index value is 0
773 for Protocol Versions 1.1 and above.
774 25. ResultMessage – May or may not be included in responses and the value (if included) may vary
775 from the text contained within the test case.
776 26. The list of Protocol Versions returned in a DiscoverVersion response may include additional
777 protocol versions if the request has not specified a list of client supported Protocol Versions.
778 27. VendorIdentification - The value (if included) may vary from the text contained within the test
779 case.

780 8.4.2 Variable behavior

781 An implementation conformant to this Profile SHALL allow variation of the following behavior:

- 782 1. A test MAY omit the clean-up requests and responses (containing Revoke and/or Destroy) at the
783 end of the test provided there is a separate mechanism to remove the created objects during
784 testing.
785 2. A test MAY omit the test identifiers if the client is unable to include them in requests. This
786 includes the following attributes:
787 a. Name; and
788 b. x-ID
789 3. A test MAY perform requests with multiple batch items or as multiple requests with a single batch
790 item provided the sequence of operations are equivalent
791 4. A request MAY contain an optional *Authentication* [KMIP_SPEC] structure within each request.

Appendix A. Acknowledgments

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

793 **Original HTTPS Profile Proposal:**

794 Alan Frindell, SafeNet, Inc.

795

796 **Original HTTPS Profile Contributors:**

797 Mathias Björkqvist, IBM

Participants:

798 Hal Aldridge, Sypris Electronics
799 Mike Allen, Symantec
800 Gordon Arnold, IBM
801 Todd Arnold, IBM
802 Richard Austin, Hewlett-Packard
803 Lars Bagnert, PrimeKey
804 Elaine Barker, NIST
805 Peter Bartok, Venafi, Inc.
806 Tom Benjamin, IBM
807 Anthony Berglas, Cryptsoft
808 Mathias Björkqvist, IBM
809 Kevin Bocket, Venafi
810 Anne Bolgert, IBM
811 Alan Brown, Thales e-Security
812 Tim Bruce, CA Technologies
813 Chris Burchett, Credant Technologies, Inc.
814 Kelley Burgin, National Security Agency
815 Robert Burns, Thales e-Security
816 Chuck Castleton, Venafi
817 Kenli Chong, QuintessenceLabs
818 John Clark, Hewlett-Packard
819 Tom Clifford, Symantec Corp.
820 Doron Cohen, SafeNet, Inc
821 Tony Cox, Cryptsoft
822 Russell Dietz, SafeNet, Inc
823 Graydon Dodson, Lexmark International Inc.
824 Vinod Duggirala, EMC Corporation
825 Chris Dunn, SafeNet, Inc.
826 Michael Duren, Sypris Electronics
827 James Dzierzanowski, American Express CCoE
828 Faisal Faruqui, Thales e-Security
829 Stan Feather, Hewlett-Packard
830 David Finkelstein, Symantec Corp.
831 James Fitzgerald, SafeNet, Inc.
832 Indra Fitzgerald, Hewlett-Packard
833 Judith Furlong, EMC Corporation
834 Susan Gleeson, Oracle
835 Robert Griffin, EMC Corporation
836 Paul Grojean, Individual
837 Robert Haas, IBM
838 Thomas Hardjono, M.I.T.
839 ChengDong He, Huawei Technologies Co., Ltd.
840 Steve He, Vormetric

841 Kurt Heberlein, Hewlett-Packard
842 Larry Hofer, Emulex Corporation
843 Maryann Hondo, IBM
844 Walt Hubis, NetApp
845 Tim Hudson, Cryptsoft
846 Jonas Iggbom, Venafi, Inc.
847 Sitaram Inguva, American Express CCoE
848 Jay Jacobs, Target Corporation
849 Glen Jaquette, IBM
850 Mahadev Karadiguddi, NetApp
851 Greg Kazmierczak, Wave Systems Corp.
852 Marc Kenig, SafeNet, Inc.
853 Mark Knight, Thales e-Security
854 Kathy Kriese, Symantec Corporation
855 Mark Lambiase, SecureAuth
856 John Leiseboer, Quintessence Labs
857 Hal Lockhart, Oracle Corporation
858 Robert Lockhart, Thales e-Security
859 Anne Luk, Cryptsoft
860 Sairam Manidi, Freescale
861 Luther Martin, Voltage Security
862 Neil McEvoy, iFOSSF
863 Marina Milshtein, Individual
864 Dale Moberg, Axway Software
865 Jishnu Mukeri, Hewlett-Packard
866 Bryan Olson, Hewlett-Packard
867 John Peck, IBM
868 Rob Philpott, EMC Corporation
869 Denis Pochuev, SafeNet, Inc.
870 Reid Poole, Venafi, Inc.
871 Ajai Puri, SafeNet, Inc.
872 Saravanan Ramalingam, Thales e-Security
873 Peter Reed, SafeNet, Inc.
874 Bruce Rich, IBM
875 Christina Richards, American Express CCoE
876 Warren Robbins, Dell
877 Peter Robinson, EMC Corporation
878 Scott Rotondo, Oracle
879 Saikat Saha, SafeNet, Inc.
880 Anil Saldhana, Red Hat
881 Subhash Sankuratipati, NetApp
882 Boris Schumperli, Cryptomathic
883 Greg Singh, QuintessenceLabs
884 David Smith, Venafi, Inc
885 Brian Spector, Certivox
886 Terence Spies, Voltage Security
887 Deborah Steckroth, RouteOne LLC
888 Michael Stevens, QuintessenceLabs
889 Marcus Streets, Thales e-Security
890 Satish Sundar, IBM
891 Kiran Thota, VMware
892 Somanchi Trinath, Freescale Semiconductor, Inc.
893 Nathan Turajski, Thales e-Security
894 Sean Turner, IECA, Inc.
895 Paul Turner, Venafi, Inc.
896 Rod Wideman, Quantum Corporation
897 Steven Wierenga, Hewlett-Packard

898 Jin Wong, QuintessenceLabs
899 Sameer Yami, Thales e-Security
900 Peter Yee, EMC Corporation
901 Krishna Yellepeddy, IBM
902 Catherine Ying, SafeNet, Inc.
903 Tatu Ylonen, SSH Communications Security (Tectia Corp)
904 Michael Yoder, Vormetric. Inc.
905 Magda Zdunkiewicz, Cryptsoft
906 Peter Zelechowski, Election Systems & Software

Appendix B. KMIP Specification Cross Reference

Reference Term	KMIP 1.0	KMIP 1.1	KMIP 1.2
1 Introduction			
<i>Non-Normative References</i>	1.3.	1.3.	1.3.
<i>Normative References</i>	1.2.	1.2.	1.2.
<i>Terminology</i>	1.1.	1.1.	1.1.
2 Objects			
<i>Attribute</i>	2.1.1.	2.1.1.	2.1.1.
<i>Base Objects</i>	2.1.	2.1.	2.1.
<i>Certificate</i>	2.2.1.	2.2.1.	2.2.1.
<i>Credential</i>	2.1.2.	2.1.2.	2.1.2.
<i>Data</i>	-	-	2.1.10.
<i>Data Length</i>	-	-	2.1.11.
<i>Extension Information</i>	-	2.1.9.	2.1.9.
<i>Key Block</i>	2.1.3.	2.1.3.	2.1.3.
<i>Key Value</i>	2.1.4.	2.1.4.	2.1.4.
<i>Key Wrapping Data</i>	2.1.5.	2.1.5.	2.1.5.
<i>Key Wrapping Specification</i>	2.1.6.	2.1.6.	2.1.6.
<i>MAC Data</i>	-	-	2.1.13.
<i>Managed Objects</i>	2.2.	2.2.	2.2.
<i>Nonce</i>	-	-	2.1.14.
<i>Opaque Object</i>	2.2.8.	2.2.8.	2.2.8.
<i>PGP Key</i>	-	-	2.2.9.
<i>Private Key</i>	2.2.4.	2.2.4.	2.2.4.
<i>Public Key</i>	2.2.3.	2.2.3.	2.2.3.
<i>Secret Data</i>	2.2.7.	2.2.7.	2.2.7.
<i>Signature Data</i>	-	-	2.1.12.
<i>Split Key</i>	2.2.5.	2.2.5.	2.2.5.
<i>Symmetric Key</i>	2.2.2.	2.2.2.	2.2.2.
<i>Template</i>	2.2.6.	2.2.6.	2.2.6.
<i>Template-Attribute Structures</i>	2.1.8.	2.1.8.	2.1.8.
<i>Transparent DH Private Key</i>	2.1.7.6.	2.1.7.6.	2.1.7.6.
<i>Transparent DH Public Key</i>	2.1.7.7.	2.1.7.7.	2.1.7.7.
<i>Transparent DSA Private Key</i>	2.1.7.2.	2.1.7.2.	2.1.7.2.
<i>Transparent DSA Public Key</i>	2.1.7.3.	2.1.7.3.	2.1.7.3.
<i>Transparent ECDH Private Key</i>	2.1.7.10.	2.1.7.10.	2.1.7.10.
<i>Transparent ECDH Public Key</i>	2.1.7.11.	2.1.7.11.	2.1.7.11.
<i>Transparent ECDSA Private Key</i>	2.1.7.8.	2.1.7.8.	2.1.7.8.
<i>Transparent ECDSA Public Key</i>	2.1.7.9.	2.1.7.9.	2.1.7.9.
<i>Transparent ECMQV Private Key</i>	2.1.7.12.	2.1.7.12.	2.1.7.12.
<i>Transparent ECMQV Public Key</i>	2.1.7.13.	2.1.7.13.	2.1.7.13.
<i>Transparent Key Structures</i>	2.1.7.	2.1.7.	2.1.7.
<i>Transparent RSA Private Key</i>	2.1.7.4.	2.1.7.4.	2.1.7.4.
<i>Transparent RSA Public Key</i>	2.1.7.5.	2.1.7.5.	2.1.7.5.
<i>Transparent Symmetric Key</i>	2.1.7.1.	2.1.7.1.	2.1.7.1.
3 Attributes			
<i>Activation Date</i>	3.19.	3.24.	3.24.
<i>Alternative Name</i>	-	-	3.40.
<i>Application Specific Information</i>	3.30.	3.36.	3.36.
<i>Archive Date</i>	3.27.	3.32.	3.32.

Reference Term	KMIP 1.0	KMIP 1.1	KMIP 1.2
<i>Attributes</i>	3	3	3
<i>Certificate Identifier</i>	3.9.	3.13.	3.13.
<i>Certificate Issuer</i>	3.11.	3.15.	3.15.
<i>Certificate Length</i>	-	3.9.	3.9.
<i>Certificate Subject</i>	3.10.	3.14.	3.14.
<i>Certificate Type</i>	3.8.	3.8.	3.8.
<i>Compromise Date</i>	3.25.	3.30.	3.30.
<i>Compromise Occurrence Date</i>	3.24.	3.29.	3.29.
<i>Contact Information</i>	3.31.	3.37.	3.37.
<i>Cryptographic Algorithm</i>	3.4.	3.4.	3.4.
<i>Cryptographic Domain Parameters</i>	3.7.	3.7.	3.7.
<i>Cryptographic Length</i>	3.5.	3.5.	3.5.
<i>Cryptographic Parameters</i>	3.6.	3.6.	3.6.
<i>Custom Attribute</i>	3.33.	3.39.	3.39.
<i>Deactivation Date</i>	3.22.	3.27.	3.27.
<i>Default Operation Policy</i>	3.13.2.	3.18.2.	3.18.2.
<i>Default Operation Policy for Certificates and Public Key Objects</i>	3.13.2.2.	3.18.2.2.	3.18.2.2.
<i>Default Operation Policy for Secret Objects</i>	3.13.2.1.	3.18.2.1.	3.18.2.1.
<i>Default Operation Policy for Template Objects</i>	3.13.2.3.	3.18.2.3.	3.18.2.3.
<i>Destroy Date</i>	3.23.	3.28.	3.28.
<i>Digest</i>	3.12.	3.17.	3.17.
<i>Digital Signature Algorithm</i>	-	3.16.	3.16.
<i>Fresh</i>	-	3.34.	3.34.
<i>Initial Date</i>	3.18.	3.23.	3.23.
<i>Key Value Location</i>	-	-	3.42.
<i>Key Value Present</i>	-	-	3.41.
<i>Last Change Date</i>	3.32.	3.38.	3.38.
<i>Lease Time</i>	3.15.	3.20.	3.20.
<i>Link</i>	3.29.	3.35.	3.35.
<i>Name</i>	3.2.	3.2.	3.2.
<i>Object Group</i>	3.28.	3.33.	3.33.
<i>Object Type</i>	3.3.	3.3.	3.3.
<i>Operation Policy Name</i>	3.13.	3.18.	3.18.
<i>Operations outside of operation policy control</i>	3.13.1.	3.18.1.	3.18.1.
<i>Original Creation Date</i>	-	-	3.43.
<i>Process Start Date</i>	3.20.	3.25.	3.25.
<i>Protect Stop Date</i>	3.21.	3.26.	3.26.
<i>Revocation Reason</i>	3.26.	3.31.	3.31.
<i>State</i>	3.17.	3.22.	3.22.
<i>Unique Identifier</i>	3.1.	3.1.	3.1.
<i>Usage Limits</i>	3.16.	3.21.	3.21.
<i>X.509 Certificate Identifier</i>	-	3.10.	3.10.
<i>X.509 Certificate Issuer</i>	-	3.12.	3.12.
<i>X.509 Certificate Subject</i>	-	3.11.	3.11.
4 Client-to-Server Operations			
<i>Activate</i>	4.18.	4.19.	4.19.
<i>Add Attribute</i>	4.13.	4.14.	4.14.
<i>Archive</i>	4.21.	4.22.	4.22.
<i>Cancel</i>	4.25.	4.27.	4.27.
<i>Certify</i>	4.6.	4.7.	4.7.
<i>Check</i>	4.9.	4.10.	4.10.
<i>Create</i>	4.1.	4.1.	4.1.
<i>Create Key Pair</i>	4.2.	4.2.	4.2.

Reference Term	KMIP 1.0	KMIP 1.1	KMIP 1.2
<i>Create Split Key</i>	-	-	4.38.
<i>Decrypt</i>	-	-	4.30.
<i>Delete Attribute</i>	4.15.	4.16.	4.16.
<i>Derive Key</i>	4.5.	4.6.	4.6.
<i>Destroy</i>	4.20.	4.21.	4.21.
<i>Discover Versions</i>	-	4.26.	4.26.
<i>Encrypt</i>	-	-	4.29.
<i>Get</i>	4.10.	4.11.	4.11.
<i>Get Attribute List</i>	4.12.	4.13.	4.13.
<i>Get Attributes</i>	4.11.	4.12.	4.12.
<i>Get Usage Allocation</i>	4.17.	4.18.	4.18.
<i>Hash</i>	-	-	4.37.
<i>Join Split Key</i>	-	-	4.39.
<i>Locate</i>	4.8.	4.9.	4.9.
<i>MAC</i>	-	-	4.33.
<i>MAC Verify</i>	-	-	4.34.
<i>Modify Attribute</i>	4.14.	4.15.	4.15.
<i>Obtain Lease</i>	4.16.	4.17.	4.17.
<i>Poll</i>	4.26.	4.28.	4.28.
<i>Query</i>	4.24.	4.25.	4.25.
<i>Re-certify</i>	4.7.	4.8.	4.8.
<i>Recover</i>	4.22.	4.23.	4.23.
<i>Register</i>	4.3.	4.3.	4.3.
<i>Re-key</i>	4.4.	4.4.	4.4.
<i>Re-key Key Pair</i>	-	4.5.	4.5.
<i>Revoke</i>	4.19.	4.20.	4.20.
<i>RNG Retrieve</i>	-	-	4.35.
<i>RNG Seed</i>	-	-	4.36.
<i>Sign</i>	-	-	4.31.
<i>Signature Verify</i>	-	-	4.32.
<i>Validate</i>	4.23.	4.24.	4.24.
5 Server-to-Client Operations			
<i>Notify</i>	5.1.	5.1.	5.1.
<i>Put</i>	5.2.	5.2.	5.2.
6 Message Contents			
<i>Asynchronous Correlation Value</i>	6.8.	6.8.	6.8.
<i>Asynchronous Indicator</i>	6.7.	6.7.	6.7.
<i>Attestation Capable Indicator</i>	-	-	6.17.
<i>Batch Count</i>	6.14.	6.14.	6.14.
<i>Batch Error Continuation Option</i>	6.13.	6.13.	6.13.
<i>Batch Item</i>	6.15.	6.15.	6.15.
<i>Batch Order Option</i>	6.12.	6.12.	6.12.
<i>Maximum Response Size</i>	6.3.	6.3.	6.3.
<i>Message Extension</i>	6.16.	6.16.	6.16.
<i>Operation</i>	6.2.	6.2.	6.2.
<i>Protocol Version</i>	6.1.	6.1.	6.1.
<i>Result Message</i>	6.11.	6.11.	6.11.
<i>Result Reason</i>	6.10.	6.10.	6.10.
<i>Result Status</i>	6.9.	6.9.	6.9.
<i>Time Stamp</i>	6.5.	6.5.	6.5.
<i>Unique Batch Item ID</i>	6.4.	6.4.	6.4.
7 Message Format			

Reference Term	KMIP 1.0	KMIP 1.1	KMIP 1.2
<i>Message Structure</i>	7.1.	7.1.	7.1.
<i>Operations</i>	7.2.	7.2.	7.2.
8 Authentication			
<i>Authentication</i>	8	8	8
9 Message Encoding			
<i>Alternative Name Type Enumeration</i>	-	-	9.1.3.2.34.
<i>Attestation Type Enumeration</i>	-	-	9.1.3.2.36.
<i>Batch Error Continuation Option Enumeration</i>	9.1.3.2.29.	9.1.3.2.30.	9.1.3.2.30.
<i>Bit Masks</i>	9.1.3.3.	9.1.3.3.	9.1.3.3.
<i>Block Cipher Mode Enumeration</i>	9.1.3.2.13.	9.1.3.2.14.	9.1.3.2.14.
<i>Cancellation Result Enumeration</i>	9.1.3.2.24.	9.1.3.2.25.	9.1.3.2.25.
<i>Certificate Request Type Enumeration</i>	9.1.3.2.21.	9.1.3.2.22.	9.1.3.2.22.
<i>Certificate Type Enumeration</i>	9.1.3.2.6.	9.1.3.2.6.	9.1.3.2.6.
<i>Credential Type Enumeration</i>	9.1.3.2.1.	9.1.3.2.1.	9.1.3.2.1.
<i>Cryptographic Algorithm Enumeration</i>	9.1.3.2.12.	9.1.3.2.13.	9.1.3.2.13.
<i>Cryptographic Usage Mask</i>	9.1.3.3.1.	9.1.3.3.1.	9.1.3.3.1.
<i>Defined Values</i>	9.1.3.	9.1.3.	9.1.3.
<i>Derivation Method Enumeration</i>	9.1.3.2.20.	9.1.3.2.21.	9.1.3.2.21.
<i>Digital Signature Algorithm Enumeration</i>	-	9.1.3.2.7.	9.1.3.2.7.
<i>Encoding Option Enumeration</i>	-	9.1.3.2.32.	9.1.3.2.32.
<i>Enumerations</i>	9.1.3.2.	9.1.3.2.	9.1.3.2.
<i>Examples</i>	9.1.2.	9.1.2.	9.1.2.
<i>Hashing Algorithm Enumeration</i>	9.1.3.2.15.	9.1.3.2.16.	9.1.3.2.16.
<i>Item Length</i>	9.1.1.3.	9.1.1.3.	9.1.1.3.
<i>Item Tag</i>	9.1.1.1.	9.1.1.1.	9.1.1.1.
<i>Item Type</i>	9.1.1.2.	9.1.1.2.	9.1.1.2.
<i>Item Value</i>	9.1.1.4.	9.1.1.4.	9.1.1.4.
<i>Key Compression Type Enumeration</i>	9.1.3.2.2.	9.1.3.2.2.	9.1.3.2.2.
<i>Key Format Type Enumeration</i>	9.1.3.2.3.	9.1.3.2.3.	9.1.3.2.3.
<i>Key Role Type Enumeration</i>	9.1.3.2.16.	9.1.3.2.17.	9.1.3.2.17.
<i>Key Value Location Type Enumeration</i>	-	-	9.1.3.2.35.
<i>Link Type Enumeration</i>	9.1.3.2.19.	9.1.3.2.20.	9.1.3.2.20.
<i>Name Type Enumeration</i>	9.1.3.2.10.	9.1.3.2.11.	9.1.3.2.11.
<i>Object Group Member Enumeration</i>	-	9.1.3.2.33.	9.1.3.2.33.
<i>Object Type Enumeration</i>	9.1.3.2.11.	9.1.3.2.12.	9.1.3.2.12.
<i>Opaque Data Type Enumeration</i>	9.1.3.2.9.	9.1.3.2.10.	9.1.3.2.10.
<i>Operation Enumeration</i>	9.1.3.2.26.	9.1.3.2.27.	9.1.3.2.27.
<i>Padding Method Enumeration</i>	9.1.3.2.14.	9.1.3.2.15.	9.1.3.2.15.
<i>Put Function Enumeration</i>	9.1.3.2.25.	9.1.3.2.26.	9.1.3.2.26.
<i>Query Function Enumeration</i>	9.1.3.2.23.	9.1.3.2.24.	9.1.3.2.24.
<i>Recommended Curve Enumeration for ECDSA, ECDH, and ECMQV</i>	9.1.3.2.5.	9.1.3.2.5.	9.1.3.2.5.
<i>Result Reason Enumeration</i>	9.1.3.2.28.	9.1.3.2.29.	9.1.3.2.29.
<i>Result Status Enumeration</i>	9.1.3.2.27.	9.1.3.2.28.	9.1.3.2.28.
<i>Revocation Reason Code Enumeration</i>	9.1.3.2.18.	9.1.3.2.19.	9.1.3.2.19.
<i>Secret Data Type Enumeration</i>	9.1.3.2.8.	9.1.3.2.9.	9.1.3.2.9.
<i>Split Key Method Enumeration</i>	9.1.3.2.7.	9.1.3.2.8.	9.1.3.2.8.
<i>State Enumeration</i>	9.1.3.2.17.	9.1.3.2.18.	9.1.3.2.18.
<i>Storage Status Mask</i>	9.1.3.3.2.	9.1.3.3.2.	9.1.3.3.2.
<i>Tags</i>	9.1.3.1.	9.1.3.1.	9.1.3.1.
<i>TTLV Encoding</i>	9.1.	9.1.	9.1.
<i>TTLV Encoding Fields</i>	9.1.1.	9.1.1.	9.1.1.
<i>Usage Limits Unit Enumeration</i>	9.1.3.2.30.	9.1.3.2.31.	9.1.3.2.31.

Reference Term	KMIP 1.0	KMIP 1.1	KMIP 1.2
<i>Validity Indicator Enumeration</i>	9.1.3.2.22.	9.1.3.2.23.	9.1.3.2.23.
<i>Wrapping Method Enumeration</i>	9.1.3.2.4.	9.1.3.2.4.	9.1.3.2.4.
<i>XML Encoding</i>	9.2.	-	-
10 Transport			
<i>Transport</i>	10	10	10
12 KMIP Server and Client Implementation Conformance			
<i>Conformance clauses for a KMIP Server</i>	12.1.	-	-
<i>KMIP Client Implementation Conformance</i>	-	12.2.	12.2.
<i>KMIP Server Implementation Conformance</i>	-	12.1.	12.1.

907 **Appendix C. Revision History**

908

Revision	Date	Editor	Changes Made
wd01	26-June-2013	Tim Hudson	Merged version of the three committee draft documents. Updated conformance wording style. Updated test case style. Applied new OASIS template.
wd02	6-August-2013	Tim Hudson	Updated to include Permitted Test Case Variations
wd03	10-August-2013	Tim Hudson	Updated Permitted Test Case Variations
pr01update	11-June-2014	Tim Hudson	Updated following Public Review 01

909