Diameter XML Dictionary
draft-frascone-xml-dictionary-00

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with Section 6 of BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/ietf/1id-abstracts.txt.

The list of Internet-Draft Shadow Directories can be accessed at http://www.ietf.org/shadow.html.

This Internet-Draft will expire on March 30, 2008.

Copyright Notice

Copyright (C) The IETF Trust (2007).
Abstract

This document describes the representation of the Diameter dictionary in XML. The resulting XML dictionary describes both the attribute-value pairs (AVPs) and command structures.

Table of Contents

1.  Introduction .................................................. 4
2.  Conventions Used in This Document ........................... 5
3.  Dictionary Layout .............................................. 6
  3.1.  Vendor Element ............................................ 6
       3.1.1.  'id' Attribute .................................... 6
       3.1.2.  'name' Attribute .................................. 6
  3.2.  Base Element ................................................ 7
  3.3.  Application Element ........................................ 7
  3.4.  Base Protocol and Application Elements ..................... 7
       3.4.1.  'id' Attribute .................................... 8
       3.4.2.  'name' Attribute .................................. 8
       3.4.3.  'uri' Attribute ................................... 8
  3.5.  Command Element ........................................... 8
       3.5.1.  'name' Attribute .................................. 9
       3.5.2.  'code' Attribute .................................. 9
       3.5.3.  'vendor-id' Attribute ............................. 9
  3.6.  AVP Rule Element ........................................... 9
       3.6.1.  'name' Attribute .................................. 10
       3.6.2.  'position' Attribute ................................ 10
       3.6.3.  'maximum' Attribute ................................ 10
       3.6.4.  'minimum' Attribute ................................ 11
  3.7.  Type Definition Element ................................... 11
       3.7.1.  'type-name' Attribute .............................. 11
       3.7.2.  'type-parent' Attribute ............................ 11
       3.7.3.  'description' Attribute ............................ 11
  3.8.  Attribute Value Pair Element .............................. 12
       3.8.1.  'name' Attribute .................................. 13
       3.8.2.  'description' Attribute ............................ 13
       3.8.3.  'code' Attribute ................................... 13
       3.8.4.  3.8.4 'may-encrypt' Attribute ...................... 13
       3.8.5.  3.8.5 'mandatory' Attribute ........................ 13
       3.8.6.  3.8.6 'protected' Attribute ........................ 13
       3.8.7.  3.8.7 'vendor-id' Attribute ........................ 13
  3.9.  Type Element ............................................... 13
       3.9.1.  'type-name' attribute .............................. 14
  3.10. Grouped AVPs Element ...................................... 14
        3.10.1.  'name' Attribute .................................. 14
        3.10.2.  'vendor-id' Attribute ............................ 14
  3.11. Enumerated Element ........................................ 14
1. Introduction

Diameter [RFC3588] is an extensible protocol used to provide AAA services to different access technologies. To maintain extensibility, Diameter uses a dictionary to provide it with the format of commands and AVPs. This document describes the representation of the Diameter dictionary using XML [xml]
2. Conventions Used in This Document

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].
3. Dictionary Layout

The root or top-level element of a Diameter dictionary is the 'dictionary' element. The dictionary element contains zero or more 'vendor' elements, the 'base' element and zero or more 'application' elements.

The top-level XML file containing the 'dictionary' element SHOULD be named 'dictionary.xml'. Each 'application' element SHOULD be defined in a separate XML file and referenced from the top-level XML file using an external entity declaration.

'dictionary' Element Syntax:

```
+------------+----------------+
|  Element   | Classification |
+------------+----------------+
|vendor      |   Zero or More  |
+------------+----------------+
|base        |       Required  |
+------------+----------------+
|application |   Zero or More  |
+----------------+
```

3.1. Vendor Element

The Vendor element defines a vendor by a name and associated IANA assigned ‘SMI Network Management Private Enterprise Codes’ [iana].

'vendor' Attribute Syntax:

```
+----------+----------+-------------+--------+
|Attribute | Presence | Constraints | Values |
+----------+----------+-------------+--------+
|   id     | Required |  UniqueKey  | String |
+----------+----------+-------------+--------+
|  name    | Required |    None     | String |
+----------+----------+-------------+--------+
```

3.1.1. ‘id’ Attribute

The ‘id’ attribute is the vendor code assigned by IANA [iana]. The ‘id’ MUST be unique across all Vendor element definitions.

3.1.2. ‘name’ Attribute

The ‘name’ attribute is some text describing the vendor. Although the Diameter protocol only requires the vendor code for encoding and
decoding messages, the vendor name MAY be used in trace utilities to facilitate debugging.

3.2. Base Element

The base element defines the commands, data types and AVPs that are part of the Diameter Base Protocol [RFC3588].

‘base’ Element Syntax:

<table>
<thead>
<tr>
<th>Element</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>command</td>
<td>One or More</td>
</tr>
<tr>
<td>typedefn</td>
<td>One or More</td>
</tr>
<tr>
<td>avp</td>
<td>One or More</td>
</tr>
</tbody>
</table>

3.3. Application Element

One of the ways in which the Diameter protocol can be extended is through the addition of new applications. The application element defines the new Commands, Types or AVPs needed to support a new Diameter application. It may also reference elements defined in the base protocol.

‘application’ Element Syntax:

<table>
<thead>
<tr>
<th>Element</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>command</td>
<td>Zero or More</td>
</tr>
<tr>
<td>typedefn</td>
<td>Zero or More</td>
</tr>
<tr>
<td>avp</td>
<td>Zero or More</td>
</tr>
</tbody>
</table>

3.4. Base Protocol and Application Elements

The base commands, and application specific commands have identical syntax, with the exception that the base protocol requires at least one type, avp, and command be defined. So, they are being described together.

Applications must define any Commands, Types, or AVPs that they
create. The application (or base) element holds those definitions.

‘base’ Attribute Syntax:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Presence</th>
<th>Constraints</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>uri</td>
<td>Optional</td>
<td>None</td>
<td>String</td>
</tr>
</tbody>
</table>

‘application’ Attribute Syntax:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Presence</th>
<th>Constraints</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>Required</td>
<td>UniqueKey</td>
<td>String</td>
</tr>
<tr>
<td>name</td>
<td>Optional</td>
<td>None</td>
<td>String</td>
</tr>
<tr>
<td>uri</td>
<td>Optional</td>
<td>None</td>
<td>String</td>
</tr>
</tbody>
</table>

3.4.1. ‘id’ Attribute

The ‘id’ attribute is the IANA assigned Application Identifier for this application.

3.4.2. ‘name’ Attribute

The ‘name’ attribute is the human readable name of this application.

3.4.3. ‘uri’ Attribute

The ‘uri’ attribute is an optional reference used to provide useful information about this application. For example, the base protocol has a URI that points to the most recent Diameter Base Protocol RFC.

3.5. Command Element

A command element defines the attributes for a command, and contains any rules to be applied to it. (Note: See Section 3.6 AVP Rule Element)

‘command’ Element Syntax:
<table>
<thead>
<tr>
<th>Element</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>requestrules</td>
<td>Zero or More</td>
</tr>
<tr>
<td>answerrules</td>
<td>Zero or More</td>
</tr>
</tbody>
</table>

‘command’ Attribute Syntax:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Presence</th>
<th>Constraints</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Required</td>
<td>None</td>
<td>String</td>
</tr>
<tr>
<td>code</td>
<td>Required</td>
<td>None</td>
<td>Integer</td>
</tr>
<tr>
<td>vendor-id</td>
<td>Optional</td>
<td>Reference</td>
<td>Integer</td>
</tr>
</tbody>
</table>

3.5.1. ‘name’ Attribute

The ‘name’ attribute defines the name of the command. Only one command is defined for both ‘Request’ and ‘Answer’ portions. So, the ‘Capabilities-Exchange’ command defines the messages, ‘Capabilities-Exchange-Request’, and ‘Capabilities-Exchange-Answer’.

3.5.2. ‘code’ Attribute

The ‘code’ attribute defines the command code used to transmit this command.

3.5.3. ‘vendor-id’ Attribute

If this is a vendor specific command, then the ‘vendor-id’ attribute MUST correspond to a vendor element’s ‘id’ field.

3.6. AVP Rule Element

AVP rules elements define the placement of key AVPs within commands. They are used to do some semantic checking at the protocol layer. For example, a particular AVP might be required to be first in a particular message. This element can define those rules.

The requestrules and answerrules elements define the placement of key AVPs within request and answer commands respectively. These elements may be used to perform syntax checking at the protocol layer.
Since a command might have different rules for requests and responses, both requestrules and answerrules may be defined. Both elements must have at least one rule if they are defined.

'requestrules'/'answerrules' Element Syntax:

<table>
<thead>
<tr>
<th>Element</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>avprule</td>
<td>One or More</td>
</tr>
</tbody>
</table>

'requestrules'/'answerrules' Attribute Syntax:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Presence</th>
<th>Constraints</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Required</td>
<td>Reference</td>
<td>String</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>position</th>
<th>Required</th>
<th>None</th>
<th>first, last, or unspecified (default is unspecified)</th>
</tr>
</thead>
<tbody>
<tr>
<td>maximum</td>
<td>Required</td>
<td>None</td>
<td>Integer or &quot;none&quot;</td>
</tr>
<tr>
<td>minimum</td>
<td>Required</td>
<td>None</td>
<td>Integer</td>
</tr>
</tbody>
</table>

3.6.1. 'name' Attribute

This rule applies to the previously defined AVP with the matching 'name' attribute.

3.6.2. 'position' Attribute

The named AVP must be either 'first' in the command, 'last' in the command, or its position does not matter ('unspecified').

3.6.3. 'maximum' Attribute

The 'maximum' attribute defines the maximum number of times this AVP can occur in the command. 0 means the AVP MUST not occur in the command. 'none' means that there is no limit to the number of times this AVP may be present.
3.6.4. ‘minimum’ Attribute

The ‘minimum’ attribute defines the maximum number of times this AVP can occur in the command. 0 means the AVP is optional.

3.7. Type Definition Element

The Type Definition element defines a valid Diameter data type. Every attribute value pair definition MUST refer to a type definition. The most common use of this container is to indicate which base type a derived type derives from. This helps the server to know how (and if) an AVP should be displayed.

‘typedefn’ Attribute Syntax:

+----------------+----------+----------------+--------+
| Attribute       | Presence | Constraints    | Values |
+----------------+----------+----------------+--------+
| type-name       | Required | UniqueKey      | String |
+----------------+----------+----------------+--------+
| type-parent     | Optional | Reference      | String |
+----------------+----------+----------------+--------+
| description     | Optional | None           | String |
+----------------+----------+----------------+--------+

3.7.1. ‘type-name’ Attribute

The attribute, ‘type-name’ contains an ASCII representation of the type. This attribute is of type ID allowing the DTD to enforce its uniqueness across all typedefn elements. This also permits other attributes of type IDREF to refer to the type-name value and have the DTD enforce the referential integrity.

3.7.2. ‘type-parent’ Attribute

The ‘type-parent’ attribute is required for all derived types. This attribute is of type IDREF ensuring that its value MUST correspond to the value of a type-name attribute of a pre-defined typedefn element.

3.7.3. ‘description’ Attribute

The ‘description’ attribute is an optional attribute describing the type. Typically a human readable description of what the type is used for would be included here.
3.8. Attribute Value Pair Element

The avp element completely defines one Attribute Value Pair and is the most frequently used element in the dictionary. The avp element contains either a type element, or a grouped element, and zero or more enum elements together with attributes that completely define the AVP including format and flags.

An AVP should only contain enumerations if the type is Unsigned32.

‘avp’ Element Syntax:

+----------------+----------------+
| Element         | Classification |
+----------------+----------------+
| type            |       Optional  |
+----------------+----------------+
| grouped         |       Optional  |
+----------------+----------------+
| avp             |   Zero or More  |
+----------------+----------------+

‘avp’ Attribute Syntax:

+----------------+----------+--------------+----------------+
| Attribute       | Presence | Constraints  | Values          |
| name            | Required |       UniqueKey |       String     |
+----------------+----------+--------------+----------------+
| description     | Optional |    None      |       String     |
+----------------+----------+--------------+----------------+
| code            | Required |       UniqueKey |        Integer   |
+----------------+----------+--------------+----------------+
| may-encrypt     | Optional |    None      |     yes or no   |
|                 |          |             |     (default is yes) |
+----------------+----------+--------------+----------------+
| mandatory       | Optional |    None      | must, may,     |
|                 |          |             | mustnot, shouldnot |
|                 |          |             |  (default is may) |
+----------------+----------+--------------+----------------+
| protected       | Optional |    None      | must, may,     |
|                 |          |             | mustnot, shouldnot |
|                 |          |             |  (default is may) |
+----------------+----------+--------------+----------------+
| vendor-id       | Optional |    Reference |        Integer   |
+----------------+----------+--------------+----------------+
3.8.1. 'name' Attribute

The 'name' attribute is the mnemonic describing this attribute, for example, 'User-Name'.

3.8.2. 'description' Attribute

The 'description' attribute is an optional attribute used to describe the use of the AVP.

3.8.3. 'code' Attribute

The 'code' attribute defines the integer value used to encode the AVP for transmission on the network.

3.8.4. 3.8.4 'may-encrypt' Attribute

If the 'may-encrypt' attribute is 'yes', then this AVP will be sent encrypted if the connection uses CMS security.

3.8.5. 3.8.5 'mandatory' Attribute

The 'mandatory' attribute defines whether the mandatory bit of this AVP should or should not be set.

3.8.6. 3.8.6 'protected' Attribute

The 'protected' attribute defines whether the protected bit of this AVP should or should not be set.

3.8.7. 3.8.7 'vendor-id' Attribute

The 'vendor-id' attribute should be set to the 'id' attribute of a 'vendor' element, if this is a vendor specific AVP.

3.9. Type Element

The type element defines the data type of the AVP in which it appears. This element MUST appear in all non-grouped AVP definitions.

'type' Attribute Syntax:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Presence</th>
<th>Constraints</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Required</td>
<td>UniqueKey</td>
<td>String</td>
</tr>
<tr>
<td>-----------</td>
<td>----------</td>
<td>-------------</td>
<td>--------</td>
</tr>
</tbody>
</table>

+---------------------------------+-------------+-------------+----------+
|                                  |             |             |          |
+---------------------------------+-------------+-------------+----------+

3.9.1. ‘type-name’ attribute

The ‘type-name’ attribute contains the data type name. This attribute is of type IDREF and MUST refer to the ‘type-name’ value of a previously defined ‘typedefn’ element.

3.10. Grouped AVPs Element

The grouped element is used define an AVP which encapsulates a sequence of AVPs together as a single payload.

A ‘grouped’ element consists of one or more ‘gavp’ elements. Each ‘gavp’ element holds an avp ‘name’ and ‘vendor-id’. This way, a single ‘grouped’ element can contain references to multiple AVPs.

‘grouped’ Element Syntax:

<table>
<thead>
<tr>
<th>Element</th>
<th>Classification</th>
</tr>
</thead>
<tbody>
<tr>
<td>gavp</td>
<td>One or More</td>
</tr>
</tbody>
</table>

‘gavp’ Attribute Syntax:

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Presence</th>
<th>Constraints</th>
<th>Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>name</td>
<td>Required</td>
<td>UniqueKey</td>
<td>String</td>
</tr>
<tr>
<td>vendor-id</td>
<td>Optional</td>
<td>Reference</td>
<td>Integer</td>
</tr>
</tbody>
</table>

3.10.1. ‘name’ Attribute

The ‘name’ attribute MUST correspond to some existing AVP’s ‘name’ attribute.

3.10.2. ‘vendor-id’ Attribute

If this ‘gavp’ element refers to a vendor specific AVP, then the ‘vendor-id’ attribute MUST correspond to a Vendor’s ‘id’ attribute.

3.11. Enumerated Element

The enum element defines a name to value mapping for use in encoding and decoding AVPs of type Unsigned32.
For example, if a particular AVP had two values, Yes and No corresponding to 1 and 0, then the entry would look like this:

```
<enum name="Yes" code="1">
<enum name="No" code="0">
```

Enumerated elements should only be used with Unsigned32 typed AVPs.

Syntax:

```
+----------+----------+-------------+---------+
| Attribute | Presence | Constraints | Values  |
+----------+----------+-------------+---------+
|     name  | Required |    None     |  String |
+----------+----------+-------------+---------+
|     code  | Required |    None     |  Integer |
+----------+----------+-------------+---------+
```

3.11.1  3.11.1 'name' Attribute

The 'name' attribute is the text corresponding to a particular value for the attribute.

3.11.2  3.11.3 'code' Attribute

The 'code' attribute is the Unsigned32 value corresponding to this enumerated value.
4. Security Considerations

This document describes a dictionary and therefore depends on the security mechanisms defined in the Diameter protocol [RFC3588].
5. IANA Considerations

This document has no actions for IANA.
6. Acknowledgments

The authors would like to thank James Kempf for his input to this document.
7. References

7.1. Normative References


7.2. Informative References
Appendix A. Acknowledgments

The authors would like to thank Brian Cain for his proposal for the command element definition.
Appendix B. Document Type Definition

The following is a copy of the DTD. It is also available at http://www.diameter.org/diameter/dictionary.dtd.

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!ELEMENT dictionary (vendor*, base, application*)>
<!ELEMENT vendor EMPTY>
<!ATTLIST vendor
  id CDATA #REQUIRED
  name CDATA #REQUIRED>

<!ELEMENT base (command*, typedefn+, avp+)>
<!ATTLIST base
  uri CDATA #IMPLIED>

<!ELEMENT application (command*, typedefn*, avp*)>
<!ATTLIST application
  id CDATA #REQUIRED
  name CDATA #IMPLIED
  uri CDATA #IMPLIED>

<!ELEMENT command (requestrules*, answerrules*)>
<!ATTLIST command
  name CDATA #REQUIRED
  code CDATA #REQUIRED
  vendor-id CDATA #IMPLIED
  pbit (0 | 1) "1">

<!ELEMENT typedefn EMPTY>
<!ATTLIST typedefn
  type-name ID #REQUIRED
  type-parent IDREF #IMPLIED
  description CDATA #IMPLIED>

<!ELEMENT avp ((type | grouped), (enum*))>
<!ATTLIST avp
  name ID #REQUIRED
  description CDATA #IMPLIED
  code CDATA #REQUIRED
  may-encrypt (yes | no) "yes"
  mandatory (must | may | mustnot | shouldnot) "may"
  protected (must | may | mustnot | shouldnot) "may"
  vendor-id CDATA #IMPLIED>
```
<!ELEMENT type EMPTY>
<!ATTLIST type
type-name IDREF #REQUIRED
>
<!ELEMENT grouped (gavp+)>
<!ELEMENT gavp EMPTY>
<!ATTLIST gavp
name IDREF #REQUIRED
vendor-id CDATA #IMPLIED
>
<!ELEMENT enum EMPTY>
<!ATTLIST enum
name CDATA #REQUIRED
code CDATA #REQUIRED
>
<!ELEMENT requestrules (avprule+)>
<!ELEMENT answerrules (avprule+)>

<!ELEMENT avprule EMPTY>
<!ATTLIST avprule
name IDREF #REQUIRED
position (first | last | unspecified) "unspecified"
maximum CDATA "none"
minimum CDATA "0"
>
Appendix C. DTD & Dictionary Links

DTD:  http://www.diameter.org/diameter/dictionary.dtd

dictionary:  http://www.diameter.org/diameter/dictionary.xml
http://www.diameter.org/diameter/nasreq.xml
http://www.diameter.org/diameter/mobileipv4.xml
http://www.diameter.org/diameter/sunping.xml
Authors’ Addresses

David Frascone
Cisco Systems, Inc.
605 N. Frances Street
Terrell, TX  75160

Phone: +1 972-524-6346
Fax:   +1 978-334-0249
Email: dave@frascone.com

Mark Jones
Bridgewater Systems
303 Terry Fox Drive, Suite 500
Kanata, Ontario  K2K 3J1

Phone: +1 613-591-6655
Fax:   +1 613-591-6656
Email: mjones@bridgewatersystems.com

Erik Guttman
Sun Microsystems, Inc
Eichholzelstr, 7
Waibstadt  74914
Germany

Phone: +49 6227 356 202
Fax:   +49 7263 911 701
Email: Erik.Guttman@sun.com
Full Copyright Statement

Copyright (C) The IETF Trust (2007).

This document is subject to the rights, licenses and restrictions contained in BCP 78, and except as set forth therein, the authors retain all their rights.

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY, THE IETF TRUST AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Intellectual Property

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights 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; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in BCP 78 and BCP 79.

Copies of IPR disclosures made to the IETF Secretariat 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 specification can be obtained from the IETF on-line IPR repository at http://www.ietf.org/ipr.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).