AAA Working Group
Internet-Draft
Category: Informational

David Frascone
Sun Microsystems, Inc.
Mark Jones
Bridgewater Systems
Erik Guttman
Sun Microsystems, Inc.
February 2002

Diameter XML Dictionary
<draft-frascone-aaa-xml-dictionary-00.txt>

Status of this Memo

Copyright Notice

Abstract

This document describes a coding of Diameter dictionaries in XML. This coding is intended for use by Diameter implementations to represent Applications, Commands, Vendors, and AVPs.

Discussion of this document

Send comments to the AAA Working Group mailing list <aaa-wg@merit.edu>. To subscribe to the list, send a message with the body 'subscribe aaa-wg' to <majordomo@merit.edu>

Table of Contents

1.0 Introduction

Diameter [1] 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 [2].

2.0 Specifications and Requirements

In this document, the key words "MAY", "MUST, "MUST NOT", "optional", "recommended", "SHOULD", and "SHOULD NOT", are to be interpreted as described in [3].

3.0 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.

Syntax:

```
<!ELEMENT dictionary (
      vendor*,
      base,
      application*)>
```

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" [4].

Syntax:

```
<!ELEMENT vendor EMPTY>
<!ATTLIST vendor
      id CDATA #REQUIRED,
      name CDATA #REQUIRED>
```

Attribute	Presence	Constraints	Values
id	Required	UniqueKey	Integer
name	Required	None	String

3.1.1 "id" Attribute

The "id" attribute is the vendor code assigned by IANA [4]. The "id" MUST be unique across all Vendor element definitions although this constraint can not be enforced by the DTD.

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 [1].

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.

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.

Syntax:

```
<!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>
```

Attribute	Presence	Constraints	Values
uri	Optional	None	String
name	Optional (application only)	None	String
id	Required (application only)	None	String

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 draft.

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)

Syntax:

```
<!ELEMENT command (
      requestrules*,
      answerrules*)>
<!ATTLIST command
      name CDATA #REQUIRED
      code CDATA #REQUIRED
      vendor-id CDATA #IMPLIED>
```

Attribute	Presence	Constraints	Values
name	Required	None	String
code	Required	None	Integer
vendor-id	Optional	Reference	Integer

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.

Syntax:

```
<!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">
```

Attribute	Presence	Constraints	Values
name	Required	Reference	String
position	Required	None	first, last, or unspecified (default is unspecified)
maximum	Required	None	Integer or "none"
minimum	Required	None	Integer

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.

Syntax:

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

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.

Syntax:

```
<!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>
```

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 "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 "mandatory" Attribute

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

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 "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.

Syntax:

```
<!ELEMENT type EMPTY>
<!ATTLIST type
      type-name IDREF #REQUIRED>
```

Attribute	Presence	Constraints	Values
type-name	Required	UniqueKey	String

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.

Syntax:

```
<!ELEMENT grouped (
      gavp+)>

<!ELEMENT gavp EMPTY>
<!ATTLIST gavp
      name IDREF #REQUIRED
      vendor-id CDATA #IMPLIED>
```

Attribute	Presence	Constraints	Values
name	Required	UniqueKey	String
vendor-id	Optional	Reference	Integer

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:

```
<!ELEMENT enum EMPTY>
<!ATTLIST enum
      name CDATA #REQUIRED,
      code CDATA #REQUIRED>
```

Attribute	Presence	Constraints	Values
name	Required	None	String
code	Required	None	Integer

3.11.1 "name" Attribute

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

3.11.3 "code" Attribute

The "code" attribute is the Unsigned32 value corresponding to this enumerated value.

4.0 References

[1] Calhoun, P., et. al., "The DIAMETER Base Protocol," draft-ietf- aaa-diameter-08.txt, a work in progress.

[2] "Extensible Markup Language (XML) 1.0" W3C Recommendation 10-February-1998 http://www.w3.org/TR/1998/REC-xml-19980210

[3] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997.

[4] Reynolds, J. and J. Postel, "ASSIGNED NUMBERS", STD 2, RFC 1700, October 1994.

5.0 Acknowledgments

The authors would like to thank Brian Cain for his proposal for the

command element definition.

6.0 Authors' Addresses

Questions about this memo can be directed to:

David Frascone
Sun Microsystems, Inc.
Sun Labs
605 N. Frances Street
Terrell, Texas 75160
USA
Phone: +1 972-386-1283
Fax: +1 978-334-0249
E-mail: codemonkey@sun.com

Mark Jones
Bridgewater Systems
303 Terry Fox Drive, Suite 100,
Kanata, Ontario. K2K 3J1
Canada
Phone: +1 613-591-6655
Fax: +1 613-591-6656
E-mail: mjones@bridgewatersystems.com

Erik Guttman
Sun Microsystems, Inc.
Eichhoelzelstr. 7
74914 Waibstadt Germany
Phone: +49 6227 356 202
Fax: +49 7263 911 701
E-mail: Erik.Guttman@sun.com

Appendix A. Document Type Definition

```
     <?xml version="1.0" encoding="UTF-8"?>
     <!--
        $Log: dictionary.dtd,v $
        Revision 1.7  2002/02/06 15:39:22  dave
        Vendor name changed from #IMPLIED to #REQUIRED

        Revision 1.6  2001/12/13 23:07:26  dave
        Updated DTD and dictionary files with new changes.  Please review
        and send me e-mail with any comments.

        Revision 1.5  2001/09/27 16:50:44  dave
        Testing CVS to mailing list.  Nothing changed in file.

        Revision 1.4  2001/09/27 16:44:19  dave
        Test for cvs

        Revision 1.3  2001/09/19 21:38:57  mjones
        Removed #PCDATA from command element.

        Revision 1.2  2001/09/19 19:46:38  mjones
        Moved the vendor element to be the same level as base and application.
        Modified vendor-id to be SMI Private Enterprise Code instead of a
label.
        Removed vendor-id="None" since vendor-id was IMPLIED.
        Added type attribute to command (request or answer).
        Removed duplicate AVPs from nasreq.xml (Acct-Session-Id, Acct-Multi-
Session-Id)
        Corrected typos in enum codes for Auth-Session-State and Disconnect-
Cause.

        Revision 1.1  2001/08/24 18:04:44  chaos
        Added per Mark's request

        Revision 1.3  2001/07/31 17:43:36  chaos
        Oops, forgot to turn on validity checking.  Fixed some errors found
with validity checking turned on

        Revision 1.2  2001/07/31 16:56:15  chaos
        Lots of changes to support flags like in the draft, and to support
commands

     -->
     <!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
>

<!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 B. 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

Mirror:

DTD:
http://diameter.frascone.com/diameter/dictionary.dtd

dictionary:
http://diameter.frascone.com/diameter/dictionary.xml
http://diameter.frascone.com/diameter/nasreq.xml
http://diameter.frascone.com/diameter/mobileipv4.xml
http://diameter.frascone.com/diameter/sunping.xml