

---

# Application Program Interface Document

---

Diameter Base Protocol API Specification

---

Version 1.0

---

**Document Information**

**Document Sign Off**

Project Manager	Inamullah
Development Team	Technical Writing Department

**Document Information**

Version #	
Revision Date	
Prepared By	Shafaq Irshad.

<b>History</b>
----------------

**Document Version Control**

<b>Date</b>	<b>Revision</b>	<b>Author</b>	<b>Description</b>

**Privacy Policy**

Copyright © 2008 Advanced IMS Inc. All rights reserved.

All other product names and trade names used herein are trademarks of their respective owners.

Information in this document is subject to change without notice. No part of this document may be reproduced or transmitted in any form or by any means, electronic or mechanical, including photocopying or recording, for any purpose without the express written permission of Advanced IMS Inc.

Changes are periodically made to this document. Changes, technical inaccuracies, and typographic errors will be corrected in subsequent versions.

---

**Objective**

**Document Scope**

<Brief and quick information explaining the document's scope >

**Table of Contents**

1. Intialization and Configuration. .... 1

1.1. Intialization Structures ..... 1

1.2. Intialization and Configuration API. .... 2

2. Message Processing ..... 4

2.1. Message Structure. .... 4

2.2. Message Processing API. .... 9

3. Tranport and Peer Management..... 14

3.1. Transport and Peer Management Structures..... 14

3.2. Transport and Peer Management API. .... 18

4. Session Management ..... 30

4.1. Session Management Structures ..... 30

4.2. Session Management API..... 31

5. Realm Table Management ..... 35

5.1. Realm Table Structures. .... 35

5.2. Realm Management API. .... 37

6. Utility APIs..... 47

6.1. Global Utility Structures..... 47

6.2. Utlity API..... 49

## 1) Initialization and Configuration

### 1.1. Initialization Structures

#### 1.1.1. DiaStackInitData

*typedef struct DiaStackInitData*

```

{
    DiameterString          DiamAPILogFileFullNameWithPath;
    unsigned int           DiamAPILogLevel;
    DAPI-DList*            pListOfListenAddresses;
    DAPI-DList*            pListOfStaticPeers;
    DAPI-DList*            pListOfRealmEntries;
    unsigned int           tcTimer;
} DiaStackInitData_t;

```

#### Structure Explanation

This structure contains all the data required for the initialization of the diameter base protocol stack .This structure needs to be passed to the InitDiameterStack function.

#### Fields:

##### DiamAPILogFileFullNameWithPath

This string will contain the full path name of the log file to be used for logging by the diameter base protocol stack

##### DiamAPILogLevel

This field contains the logging level to be used during logging by the diameter base protocol stack

##### pListOfListenAddresses

This field is the head node of a list. Each node of the list is of the type of DiameterTransportObject\_t , and contains the transport and network information required for listening.

##### pListOfStaticPeers

This field is the head node of a list. Each node of the list is of the type of staticPeerEntry\_t, and contains the information about peer table entry.

##### pListOfRealmEntries

This field is the head node of a list. Each node of the list is of the type of RealmRtTableEntry\_t and contains information about the realm entry.

tcTimer

This specifies the value of the Tc timer in second .The default value used by the diameter protocol stack is 30 seconds as per RFC 3588.

**1.1.2. staticPeerEntry**

```
typedef struct staticPeerEntry
{
    DiameterIdentity                ServerID;
    DiameterTransportObject_t      TransportObject;
    bool                            bIsTLSEnabled;
    TLSData_t                       optionalTLSdata;
} staticPeerEntry_t;
```

***Structure Explanation***

This structure contains all the information that is needed for the static configuration of peers at initialization time.

**Fields:**ServerID

This fields contains the FQDN of the peer in the peer table

TransportObject

This field contains the transport object that contains all the transport level details of the peer.

bIsTLSEnabled

If this field is set then the Transport connection will be TLS enabled.

optionalTLSdata

If TLS is enabled then this field will contain the data required by TLS connection.

**1.2. Initialization and Configuration APIs****1.2.1. InitDiameterStack**

```
DiamAPIReturnVal InitDiameterStack ( DiameterBaseAPIContext_t* pDiameterContext
, DiaStackInitData_t* pDiaStackInitParams );
```

**Purpose**

The purpose of this API is to initialize and configure the Diameter base protocol stack.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
pDiaStackInitParams	DiaStackInitData_t*	Out	This structure contains all the initialization and configuration parameters for the diameter base stack.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

**1.2.2. InitAVPDictionary**

*DiamAPIRetVal InitAVPDictionary( DiameterBaseAPIContext\_t\* pDiameterContext , DiameterString strXMLDictionaryFileFullPath );*

**Purpose**

This function is used to initialize the AVP dictionary.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
strXMLDictionaryFileFullPath	DiameterString	In	This parameter is the full path name of the XML dictionary file.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

**1.2.3. CleanupDiameterStack**

*DiamAPIRetVal CleanupDiameterStack(DiameterBaseAPIContext\_t\* pDiameterContext );*

**Purpose**

This function cleans the local state of the diameter base API stack.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

**2) Message Processing****2.1. Message Structure****2.1.1. DiameterHeader**

```
typedef struct DiameterHeader
{
    unsigned char Version ;
    unsigned char MessageLength[3];
    unsigned char Flags;
    unsigned char CommandCode[3] ;
    unsigned int ApplicationID;
    unsigned int HopByHopID;
    unsigned int EndToEndID;
} DiameterHeader_t;
```

**Structure Explanation**

This structure represents the Diameter Header

**Fields:**Version

This field contains the version of Diameter Protocol.

MessageLength[3]

This field specifies the length of the diameter message in bytes. This is in Big endian form.

Flags

This field contains diameter flags.

CommandCode[3]

This field contains diameter command code. This is in Big endian form.

ApplicationID

This field contains the application ID

HopByHopID

This field contains the hop by hop identifier for the current diameter message

EndToEndID

This field contains the end to end identifier for the current diameter message

**2.1.2. DiameterMessage**

```
typedef struct DiameterMessage
{
    DiameterHeader_t    DiaHeader;
    DAPI-DList*        pListOfAVPs;
} DiameterMessage_t;
```

**Structure Explanation**

This structure specifies the diameter message.

**Fields:**DiaHeader

This field is a structure containing the diameter header

pListOfAVPs

This is the pointer to the list of AVPs. Each node of the list is of the type AVP\_t

**2.1.3. AVPHeader**

```
typedef AVPHeader
{
    unsigned int    AVPCode;
    unsigned char  Flags;
    unsigned char  AVPLength[3];
} AVPHeader_t;
```

**Structure Explanation**

This structure represents the AVP header.

**Fields:**AVPCode

This field contains the AVP code that defines the current AVP.

Flags

This field contains the AVP flags

AVPLength[3]

This field specifies the length of the AVP packet in bytes. This is in Big endian form.

**2.1.4. AVP**

*typedef struct AVP*

```
{
    AVPHeader_t      AVPHeaderInst;
    DAPI-DList*     pListOfData;
} AVP_t;
```

**Structure Explanation**

This structure represents an AVP packet.

**Fields:**AVPHeaderInst

This structure represents the diameter header

pListOfData

This is a pointer to list containing the payload for the AVP packet.

**2.1.5. Macros**

AVP code macros with the numeric value as defined by RFC 3588

<code>#define</code>	<code>AVP-ACCT-INTERIM-INTERVAL</code>	85
<code>#define</code>	<code>AVP-ACCOUNTING-REALTIME-REQUIRED</code>	483
<code>#define</code>	<code>AVP-ACCT-MULTI-SESSION-ID</code>	50
<code>#define</code>	<code>AVP-ACCOUNTING-RECORD-NUMBER</code>	485
<code>#define</code>	<code>AVP-ACCOUNTING-RECORD-TYPE</code>	480
<code>#define</code>	<code>AVP-ACCOUNTING-SESSION-ID</code>	44
<code>#define</code>	<code>AVP-ACCOUNTING-SUB-SESSION-ID</code>	287
<code>#define</code>	<code>AVP-ACCT-APPLICATION-ID</code>	259
<code>#define</code>	<code>AVP-AUTH-APPLICATION-ID</code>	258
<code>#define</code>	<code>AVP-AUTH-REQUEST-TYPE</code>	274
<code>#define</code>	<code>AVP-AUTHORIZATION-LIFETIME</code>	291
<code>#define</code>	<code>AVP-AUTH-GRACE-PERIOD</code>	276
<code>#define</code>	<code>AVP-AUTH-SESSION-STATE</code>	277
<code>#define</code>	<code>AVP-RE-AUTH-REQUEST-TYPE</code>	285
<code>#define</code>	<code>AVP-CLASS</code>	25
<code>#define</code>	<code>AVP-DESTINATION-HOST</code>	293
<code>#define</code>	<code>AVP-DESTINATION-REALM</code>	283
<code>#define</code>	<code>AVP-DISCONNECT-CAUSE</code>	273

#define	AVP-E2E-SEQUENCE	300
#define	AVP-ERROR-MESSAGE	281
#define	AVP-ERROR-REPORTING-HOST	294
#define	AVP-EVENT-TIMESTAMP	55
#define	AVP-EXPERIMENTAL-RESULT	297
#define	AVP-EXPERIMENTATION-RESULT-CODE	298
#define	AVP-FAILED-AVP	279
#define	AVP-FIRMWARE-REVISION	267
#define	AVP-HOST-IP-ADDRESS	257
#define	AVP-INBAND-SECURITY-ID	299
#define	AVP-MULTI-ROUND-TIME-OUT	272
#define	AVP-ORIGIN-HOST	264
#define	AVP-ORIGIN-REALM	296
#define	AVP-ORIGIN-STATE-ID	278
#define	AVP-PRODUCT-NAME	269
#define	AVP-PROXY-HOST	280
#define	AVP-PROXY-INFO	284
#define	AVP-PROXY-STATE	33
#define	AVP-REDIRECT-HOST	292
#define	AVP-REDIRECT-HOST-USAGE	261
#define	AVP-REDIRECT-MAX-CACHE-TIME	262
#define	AVP-RESULT-CODE	268
#define	AVP-ROUTE-RECORD	282
#define	AVP-SESSION-ID	263
#define	AVP-SESSION-TIMEOUT	27
#define	AVP-SESSION-BINDING	270
#define	AVP-SESSION-SERVER-FAILOVER	271
#define	AVP-SUPPORTED-VENDOR-ID	265
#define	AVP-TERMINATION-CAUSE	295
#define	AVP-USER-NAME	1
#define	AVP-VENDOR-ID	266
#define	AVP-VENDOR-SPECIFIC-APPLICATION-ID	260

**Getter macros for diameter header**

#define	GET-VERSION-FROM-DIA-HEADER
#define	GET-MSG-LENGTH-FROM-DIA-HEADER
#define	GET-COMMAND-CODE-FROM-DIA-HEADER
#define	GET-APPID-FROM-DIA-HEADER
#define	GET-HOPBYHOP-ID-FROM-DIA-HEADER
#define	GET-ENDTOEND-ID-FROM-DIA-HEADER
#define	GET-R-FLAG-FROM-DIA-HEADER
#define	GET-P-FLAG-FROM-DIA-HEADER
#define	GET-E-FLAG-FROM-DIA-HEADER
#define	GET-T-FLAG-FROM-DIA-HEADER

**Setter macros for diameter header**

```
#define SET-VERSION-IN-DIA-HEADER
#define SET-MSG-LENGTH-IN-DIA-HEADER
#define SET-COMMAND-CODE-IN-DIA-HEADER
#define SET-APPID-IN-DIA-HEADER
#define SET-HOPBYHOP-ID-IN-DIA-HEADER
#define SET-ENDTOEND-ID-IN-DIA-HEADER
#define SET-R-FLAG-IN-DIA-HEADER
#define SET-P-FLAG-IN-DIA-HEADER
#define SET-E-FLAG-IN-DIA-HEADER
#define SET-T-FLAG-IN-DIA-HEADER
```

**Getter macros for AVP header**

```
#define GET-AVP-CODE-FROM-AVP-HEADER
#define GET-AVP-LENGTH-FROM-AVP-HEADER
#define GET-VENDOR-ID-FROM-AVP-HEADER
#define GET-V-FLAG-FROM-AVP-HEADER
#define GET-M-FLAG-FROM-AVP-HEADER
#define GET-P-FLAG-FROM-AVP-HEADER
```

**Setter macros for AVP header**

```
#define SET-AVP-CODE-IN-AVP-HEADER
#define SET-AVP-LENGTH-IN-AVP-HEADER
#define SET-VENDOR-ID-IN-AVP-HEADER
#define SET-V-FLAG-IN-AVP-HEADER
#define SET-M-FLAG-IN-AVP-HEADER
#define SET-P-FLAG-IN-AVP-HEADER
```

**2.1.7. Type Definitions**

```
typedef AVP_t* DAPI-HANDLE-AVP;
typedef DiameterMessage_t* DAPI-HANDLE-DIAM-MSG;
```

## 2.2. Message Processing APIs

### 2.2.1. CreateAVP

```
DiamAPIRetVal CreateAVP(DiameterBaseAPIContext_t* pDiameterContext ,
    DAPI-HANDLE-AVP* pAVPHandle , unsigned int AVPCode ,void* pAVPData
);
```

#### Purpose

This function creates a new AVP.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
pAVPHandle	DAPI-HANDLE-AVP*	Out	This is a pointer to DAPI-HANDLE-AVP ,which is a handle of the newly created AVP.
AVPCode	unsigned int	In	This contains the code for the AVP being created.
pAVPData	void*	In	This contains the value of the current AVP.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 2.2.2 CreateGroupedAVP

```
DiamAPIRetVal CreateGroupedAVP (
    DiameterBaseAPIContext_t*pDiameterContext , DAPI-HANDLE-AVP*
    pGroupedAVPHandle )
```

#### Purpose

This function creates a grouped AVP structure.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
pGroupedAVPHandle	DAPI-HANDLE-AVP*	Out	This is a pointer to DAPI-HANDLE-AVP ,which is a handle of the newly created grouped AVP.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

**2.2.3. AddAVPToGroup**

*DiamAPIRetVal AddAVPToGroup ( DiameterBaseAPIContext\_t\* pDiameterContext , DAPI-HANDLE-AVP hGroupedAVP , DAPI-HANDLE-AVP hAVPToAdd );*

**Purpose**

This function adds an AVP to a grouped AVP.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hGroupedAVP	DAPI-HANDLE-AVP	Out	This is a handle to the grouped AVP.
hAVPToAdd	DAPI-HANDLE-AVP	In	This is a handle to the AVP to be added.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 2.2.4. AppendAVPToList

```
DiamAPIReturnVal AppendAVPToList ( DiameterBaseContext_t*
pDiameterContext , DAPI-DList* pAVPList , DAPI-HANDLE-AVP hAVP );
```

#### Purpose

This is a utility function that appends an AVP to the AVP list.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
pAVPList	DAPI-DList*	Out	This is pointer to the list of AVPs in which the new AVP will be appended.
hAVP	DAPI-HANDLE-AVP	In	This is the AVP which needs to be appended to the AVP list.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 2.2.5. AddAVPListToDiameterMessage

```
DiamAPIReturnVal AddAVPListToDiameterMessage(DiameterBaseContext_t*
pDiameterContext , DAPI-HANDLE-DIAM-MSG hDiaMsg , DAPI-DList*
pAVPList);
```

#### Purpose

This function adds the AVP list to the diameter message.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hDiaMsg	DAPI-HANDLE-DIAM-MSG	Out	This is a handle to the diameter message in which the list will be added.
pAVPList	DAPI-DList*	In	This is the list of AVPs that needs to be added to the diameter

message.
----------

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 2.2.6. CreateAVPList

*DiamAPIReturnVal CreateAVPList(DiameterBaseAPIContext\_t\* pDiameterContext, DAPI-DList\*\* ppAVPList);*

### Purpose

This function creates an empty AVP list.

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
ppAVPList	DAPI-DList**	Out	This is a pointer to the list of AVPs.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 2.2.7. ParseDiameterPacket

*DiamAPIReturnVal ParseDiameterPacket (DiameterBaseAPIContext\_t\* pDiameterContext, void \* pRawDiameterPacket, DAPI-HANDLE-DIAM-MSG \* pDiaMsgHandle );*

### Purpose

This function parses a diameter packet.

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API..
pRawDiameterPacket	void *	In	This is the raw diameter packet.
pDiaMsgHandle	DAPI-HANDLE-DIAM-MSG *	Out	This is a pointer to diameter message handle in which structured data will

be returned.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3) Transport and Peer Management

#### 3.1. Transport and Peer Management Enums & Structures

##### 3.1.1. PeerState

```
typedef enum PeerState
    {
        Closed,
        Wait-Conn-Ack,
        Wait-I-CEA ,
        Wait-Conn-ACK/Elect,
        Wait>Returns,
        R-Open,
        I-Open,
        Closing
    } PeerState_t;
```

##### Enum Explanation

This is an enumeration that enumerates the different states of a diameter peer.

##### 3.1.2. PeerTable Entry

```
typedef struct PeerTableEntry
    {
        DiameterString                HostIdentity;
        PeerState                    PeerStatus;
        bool                          isStatic;
        DiaTime_t                     ExpiryTime;
        bool                          isTLSEnabled;
        TLSDData_t                   optionalTLSDData;
        DiameterTransportObject_t    TransportObject ;
    } PeerTableEntry_t;
```

##### Structure Explanation

This structure contains the information that will be present in each entry of the peer table.

##### Fields:

###### HostIdentity

This field contains the value of the Host identity

###### PeerStatus

This field contains the peer status of the current peer, as defined by the peer state machine in RFC 3588

isStatic

This field specifies whether the current peer table entry was statically configured or dynamically discovered and configured.

ExpiryTime

This field contains the expiry time of the current entry if the peer is dynamically discovered.

isTLSEnabled

This field specifies if TLS is enabled in the transport with the peer referred by the current entry.

optionalTLSData

If TLS is enabled, then this field contains the optional data required by TLS.

TransportObject

This field contains the transport object associated with the current peer entry

**3.1.3. Type Defines**

```
typedef PeerTableEntry_t * DAPIHANDLE-PEERTABLE-ENTRY;
typedef DiameterTransportObject_t* DAPIHANDLE-TRANSPORT- OBJECT;
```

**3.1.4. enum IPAddrType**

```
typedef enum IPAddrType
{
    IP_v4,
    IP_v6
}IPAddrType_t;
```

**Enum Explanation**

This enumerates the type of the IP address

**Fields**

IP\_v4,  
IP version 4

IP\_v6  
IP version 6

### 3.1.5. DiameterTCPTransport

```
typedef struct DiameterTCPTransport
{
    DiameterString    strIPAddress;
    DiameterString    strFQDN;
    unsigned int      port;
    IPAddrType_t     IPAddrType;
} DiameterTCPTransport_t;
```

#### Structure Explanation

This structure contains the basic transport information for a TCP connection

#### Fields:

##### strIPAddress

String containing the IP address

##### strFQDN

String containing the fully qualified domain name

##### port

Port number

##### IPAddrType

Enumeration specifying the IP addresses type

### 3.1.6. DiameterSCTPTransport

```
typedef struct DiameterSCTPTransport
{
    DAPI-DList*       pListOfIPAddresses;
    DAPI-Dlist*       pListOfFQDNs;
    unsigned int      port;
    IPAddrType_t     IPAddrType;
} DiameterSCTPTransport_t;
```

#### Structure Explanation

This structure contains the basic transport information for an SCTP connection

#### Fields:

##### pListOfIPAddresses

The pointer to the list of strings where each string contains an IP address

##### pListOfFQDNs

The pointer to the list of strings where each string contains an FQDN

port  
Port number

IPAddrType  
Enumeration enumerating the IP address type

### 3.1.7. DiameterTransportObject

```
typedef struct DiameterTransportObject
{
    DiamTransport_t transportType;

    union DiamTransportInfo
    {
        DiameterTCPTransport_t DiamTCPTransport;
        DiameterSCTPTransport_t DiamSCTPTransport;
    };
    void* pDiamAppContext;
    void* pTransportLevelHandle
} DiameterTransportObject_t;
```

#### Structure Explanation

This is a structure that specifies the opaque transport object.

#### Fields:

transportType  
An enumeration enumerating the type of transport connection.

```
union DiamTransportInfo
{
    DiameterTCPTransport_t DiamTCPTransport;
    //Union member to be used if the transport type is TCP
    DiameterSCTPTransport_t DiamSCTPTransport;
    //Union member to be used if the transport type is SCTP.
};
```

pDiamAppContext  
A void pointer field to be used as application context.

pTransportLevelHandle

This Void pointer contains the transport handle received from the underlying framework

**3.1.8. DiamTransport**

```
typedef enum DiamTransport
{
    TCP,
    SCTP
} DiamTransport_t;
```

**3.1.9. TLSData**

```
typedef struct TLSData
{

} TLSData_t;
```

**Structure Explanation**

*This will contain all the relevant data required for TLS initialization and configuration*

**3.2. Transport and Peer Management API****3.2.1. OpenTransportConnection**

```
DiamAPIRetVal OpenTransportConnection(DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-TRANSPORT-OBJECT hTransObj );
```

**Purpose**

This function opens a transport connection.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hTransObj	DAPIHANDLE-TRANSPORT-OBJECT	In	This is the handle to the transport object.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 3.2.2. ListenForTransportConnections

*DiamAPIRetVal ListenForTransportConnections( DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-TRANSPORT-OBJECT hTransObj );*

#### Purpose

This function starts listening on a transport connection.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hTransObj	DAPIHANDLE-TRANSPORT-OBJECT	In	This is the handle to the transport object.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 3.2.3. SendMessageOnTransportConnection

*DiamAPIRetVal SendMessageOnTransportConnection( DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-DIAM-MSG hDiaMsg, DAPIHANDLE-TRANSPORT-OBJECT hTransport);*

#### Purpose

This function sends a diameter message on a transport connection.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hDiaMsg	DAPIHANDLE-DIAM-MSG	In	This is the handle to the diameter message that needs to be sent.
hTransport	DAPIHANDLE-TRANSPORT-OBJECT	In	This is the handle to the transport object on which the message needs to be sent.

## Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 3.2.4. CloseTransportConnection

*DiamAPIRetVal CloseTransportConnection ( DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-TRANSPORT-OBJECT hTransportObj);*

## Purpose

This message closes the transport connection.

## Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hTransObj	DAPIHANDLE-TRANSPORT-OBJECT	In	This is the handle to the transport object which needs to be closed.

## Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 3.2.5. SetMessageReceptionFromTransport

*DiamAPIRetVal SetMessageReceptionFromTransport( DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-TRANSPORT-OBJECT hTransObj);*

## Purpose

This sets the message reception to initiate and continue from the transport object.

## Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hTransObj	DAPIHANDLE-TRANSPORT-OBJECT	In	This is the handle to the transport object on which the message reception needs to be set and initiated.

## Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.6. CreatePeerTableEntry

```
DiamAPIReturnVal CreatePeerTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY* phPeerTableEntry );
```

## Purpose

This function creates a peer table entry.

## Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
phPeerTableEntry	DAPIHANDLE-PEERTABLE-ENTRY	Out	This is a pointer to the handle to peer table entry.

## Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.7. LookupPeerTableEntry

```
DiamAPIReturnVal LookupPeerTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY* phPeerTableEntry,
DiameterString strHostname);
```

## Purpose

This function looks up a peer table entry in the peer table.

## Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
phPeerTableEntry	DAPIHANDLE-PEERTABLE-ENTRY*	Out	This is a pointer to the handle to peer table entry.
strHostname	DiameterString	In	This is the hostname against which a look

up will be made in the peer table.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.8. InsertEntryIntoPeerTable

*DiamAPIReturnVal InsertEntryIntoPeerTable(DiameterBaseAPIContext\_t\* pDiameterContext, DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry);*

### Purpose

This function inserts an entry into the Peer table.

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the entry which needs to be inserted in the peer table.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.9. GetHostIdentityFromPeerTableEntry

*DiamAPIReturnVal GetHostIdentityFromPeerTableEntry (DiameterBaseAPIContext\_t\* pDiameterContext, DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry, DiameterString\* pHostIdentity);*

### Purpose

This function gets the host identity string associated with a Peer table entry.

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-	In	This is the handle to

	PEERTABLE-ENTRY		the peer table entry.
pHostIdentity	DiameterString*	Out	This is a pointer to a string in which the host identity will be returned.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 3.2.10. GetStaticStatusFromPeerTableEntry

```
DiamAPIRetVal GetStaticStatusFromPeerTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry ,BOOL*
pIsStatic );
```

#### Purpose

This function retrieves the value of Static field from a given peer table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
pIsStatic	BOOL*	Out	This is a pointer to the BOOL in which static field is returned.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 3.2.11. GetExpiryTimeFromPeerTableEntry

```
DiamAPIRetVal GetExpiryTimeFromPeerTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry
,DiaTime_t *pExpiryTime );
```

#### Purpose

This function gets the expiry time associated with a given Peer table entry.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
pExpiryTime	DiaTime_t *	Out	This is a pointer to a DiaTime_t structure in which the expiry time is returned.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

**3.2.12. GetTLSEnableStatusFromPeerTableEntry**

*DiamAPIReturnVal GetTLSEnableStatusFromPeerTableEntry ( DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry , BOOL\* pIsTLSEnabled );*

**Purpose**

This function retrieves the value of the IsTLSEnabled field from the peer table.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
pIsTLSEnabled	BOOL*	Out	This is a pointer to a boolean variable in which the status of the IsTLSEnabled field is returned.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.13. GetOptionalTLSDataFromPeerTableEntry

```
DiamAPIReturnVal GetOptionalTLSDataFromPeerTableEntry (
    DiameterBaseAPIContext_t* pDiameterContext , DAPIHANDLE-
    PEERTABLE-ENTRY hPeerEntry ,TLSData_t* pOptTLSData );
```

#### Purpose

This function retrieves the **optionalTLSdata** from a given peer table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
pOptTLSData	TLSData_t*	Out	This is a pointer which points to the variable in which the optional TLS data associated with the given peer table entry will be retrieved.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.14. GetTransportObjectFromPeerTableEntry

```
DiamAPIReturnVal GetTransportObjectFromPeerTableEntry (
    DiameterBaseAPIContext_t* pDiameterContext , DAPIHANDLE-PEERTABLE-
    ENTRY hPeerEntry ,DiameterTransportObject_t* pTransportObject );
```

#### Purpose

This function retrieves the Transport Object associated with a given peer table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.

pTransportObject	DiameterTransportObject_t*	Out	This is a pointer which points to a variable in which the transport Object will be filled.
------------------	----------------------------	-----	--

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.15. SetHostIdentityInPeerTableEntry

```
DiamAPIReturnVal SetHostIdentityInPeerTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry
,DiameterString strHostIdentity);
```

#### Purpose

This function sets the value of the host identity string in a given peer table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
strHostIdentity	DiameterString	In	This contains the host identity string to be set in the peer table entry.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.16. SetStaticStatusInPeerTableEntry

```
DiamAPIReturnVal SetStaticStatusInPeerTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry ,BOOL
IsStatic );
```

#### Purpose

This function sets the value of the static status field in the given peer table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the

			diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
IsStatic	BOOL	In	This contains the value of the isStatic field to be set in the peer table entry.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 3.2.17. SetExpiryTimeInPeerTableEntry

*DiamAPIRetVal SetExpiryTimeInPeerTableEntry ( DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry ,DiaTime\_t ExpiryTime );*

### Purpose

This function sets the value of the Expiry Time field in the peer table entry.

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
ExpiryTime	DiaTime_t	In	This is the expiry time value that needs to be set in the given peer table entry.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 3.2.18. SetTLSEnableStatusInPeerTableEntry

*DiamAPIRetVal SetTLSEnableStatusInPeerTableEntry ( DiameterBaseAPIContext\_t\* pDiameterContext ,DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry , BOOL IsTLSEnabled );*

**Purpose**

This function sets the value of the TLSEnableStatus field in the given peer table entry.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
IsTLSEnabled	BOOL	In	This is the value of the TLSEnabled field to be set in the given peer table entry.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

**3.2.19. SetOptionalTLSDataInPeerTableEntry**

*DiamAPIReturnVal SetOptionalTLSDataInPeerTableEntry ( DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry , TLSData\_t OptTLSData );*

**Purpose**

This function sets the optional TLS data in the given peer table entry.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
OptTLSData	TLSData_t	In	This is the optional TLS data to be set in the given peer table entry.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 3.2.20. SetTransportObjectInPeerTableEntry

```
DiamAPIReturnVal SetTransportObjectInPeerTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-PEERTABLE-ENTRY hPeerEntry
,DiameterTransportObject_t* pTransportObject );
```

#### Purpose

This function sets the value of the Transport Object in the given peer table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hPeerEntry	DAPIHANDLE-PEERTABLE-ENTRY	In	This is the handle to the peer table entry.
pTransportObject	DiameterTransportObject_t*	In	This is a pointer to the diameter transport object that needs to be set in the given peer table entry.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

## 4) Session Management API

### 4.1. Session Management Structures

#### 4.1.1. DiameterSessionInfo

```
Typedef struct DiameterSessionInfo
{
    DAPIHANDLE-TRANSPORT-OBJECT    hFirstTransLeg;
    DAPIHANDLE-TRANSPORT-OBJECT    hSecondTransLeg;
    DiameterString                 Session-ID;
} DiameterSessionInfo_t;
```

#### Structure Explanation

This structure contains the information related to a diameter session.

#### Fields:

##### hFirstTransLeg

This field is the transport object that contains the transport related information for the first leg of the session.

##### hSecondTransLeg

This field is the transport object that contains the transport related information for the second leg of the session.

##### Session-ID

This contains the actual string that is the session-ID for the current session object

#### 4.1.2. Type Definitions

```
typedef    DiameterSessionInfo_t*    DAPIHANDLE-SESSION-OBJECT;
```

## 4.2. Session Management API

### 4.2.1. CreateNewSession

```
DiamAPIReturnVal CreateNewSession ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-SESSION-OBJECT * pDiaSessObject );
```

#### Purpose

This function creates a new session.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
pDiaSessObject	DAPIHANDLE-SESSION-OBJECT *	Out	This is pointer to the session handle in which the new session object will be returned.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 4.2.2. GetTransportHandleFromSession

```
DiamAPIReturnVal GetTransportHandleFromSession ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-SESSION-OBJECT hSessObj ,
DAPIHANDLE-TRANSPORT-OBJECT* pTransHandle , BOOL
bIsFirstTransLeg );
```

#### Purpose

This function returns the transport handle associated with the current session object.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure the context of the diameter base protocol API.
hSessObj	DAPIHANDLE-SESSION-OBJECT	In	This contains the handle to the session object.
pTransHandle	, DAPIHANDLE-TRANSPORT-OBJECT*	Out	This is a pointer to the transport handle in

			which the function will return the transport object.
bIsFirstTransLeg	BOOL	In	if this is true , the handle of first leg transport will be returned , otherwise that of the second transport leg.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 4.2.3. GetSessionIDStringFromSessionObject

```
DiamAPIRetVal GetSessionIDStringFromSessionObject (
    DiameterBaseAPIContext_t* pDiameterContext , DAPIHANDLE-SESSION-
    OBJECT hSessObj , DiameterString* pstrSessID );
```

### Purpose

This function returns the SessionID string associated with a session object.

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hSessObj	DAPIHANDLE-SESSION-OBJECT	In	This contains the handle to the session object.
pstrSessID	DiameterString*	Out	This is a pointer to a diameter string in which the sessionID string will be filled.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

#### 4.2.4. AddTransportToSession

*DiamAPIReturnVal AddTransportToSession (DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-SESSION-OBJECT hSessObj , DAPIHANDLE-TRANSPORT-OBJECT hTransHandle , BOOL bIsFirstTransLeg) ;*

##### Purpose

This function associates the handle of a transport object with the session object.

##### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hSessObj	DAPIHANDLE-SESSION-OBJECT	out	This is the handle to the session object.
hTransHandle	DAPIHANDLE-TRANSPORT-OBJECT	in	This is the handle of the transport object that needs to be associated with the session object.
bisFirstTransLeg	BOOL	in	If true , then the association is for the first leg otherwise for second leg.

##### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

#### 4.2.5. TerminateSession

*DiamAPIReturnVal TerminateSession (DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-SESSION-OBJECT hSessObject);*

##### Purpose

This function terminates a diameter session.

##### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.

hSessObj	DAPIHANDLE-SESSION- OBJECT	In	This is the handle to the session object which needs to be terminated.
----------	-------------------------------	----	--

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

## 5) Realm Table Management

### 5.1. Realm Table Structures

#### 5.1.1. ApplicationIdentifier

```
typedef struct ApplicationIdentifier  
{  
    unsigned int AppID;  
    unsigned int vendorID;  
}ApplicationIdentifier_t;
```

#### Structure Explanation

This structure defines the Application Identifier

#### Fields:

AppID           The Application ID  
vendorID        The vendor ID

#### 5.1.2. LocalAction

```
typedef enum LocalAction  
{  
    LOCAL,  
    RELAY,  
    PROXY,  
    REDIRECT  
} LocalAction_t;
```

#### Enum Explanation

This enumerates message forwarding role played by the diameter node.

### 5.1.3. RealmRtTableEntry

```
typedef struct RealmRtTableEntry
{
    DiameterString      RealmName;
    ApplicationIdentifier AppIdent;
    LocalAction         LocAct;
    DAPI-Dlist*        ServerIDList;
    bool               isStatic;
    DiaTime_t          ExpirationTime;
} RealmRtTableEntry_t;
```

#### Structure Explanation

This structure contains the information present in each entry of the realm based routing table.

#### Fields:

##### RealmName

String containing the Realm Name

##### AppIdent

Structure containing the Application identifier

##### LocAct

This enumerates message forwarding role played by the diameter node.

##### ServerIDList

The pointer to the list of server Ids . Each node of the list is of the type DiameterIdentity.

##### isStatic

This field specifies whether this entry was statically configured or dynamically discovered.

##### ExpirationTime

If the current entry was dynamically discovered, then this field contains the time after which the current connection will expire.

### 5.1.4. Type Defines

```
typedef RealmRtTableEntry_t* DAPIHANDLE-REALMTABLE-ENTRY;
```

## 5.2. Realm Table Management API

### 5.2.1. CreateRealmRoutingTableEntry

```
DiamAPIReturnVal CreateRealmRoutingTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-REALMTABLE-ENTRY* phRealmTabEntry
);
```

#### Purpose

This function creates an entry for the realm based routing table.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
phRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY*	Out	This is a pointer to the realm table entry handle that will be filled.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 5.2.2. LookupRealmRoutingTableEntry

```
DiamAPIReturnVal LookupRealmRoutingTableEntry (DiameterBaseAPIContext_t*
pDiameterContext , DiameterString RealmName , ApplicationIdentifier AppIdent
,DAPIHANDLE-REALMTABLE-ENTRY* phRealmTabEntry);
```

#### Purpose

This function looks up a Realm Routing Table Entry based on the criteria of matching RealmName and Application Identifier

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
RealmName	DiameterString	In	This is a string containing the realm

			name, and is part of the criteria for lookup.
AppIdent	ApplicationIdentifier	In	This is the application identifier and is part of the criteria for lookup.
phRealmTabEntry	DAPIHANDLE- REALMTABLE-ENTRY*	Out	This is a pointer to the handle of the realm routing table entry which matches the search criteria.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

**5.2.3. GetRealmNameFromRealmTableEntry**

```
DiamAPIRetVal GetRealmNameFromRealmTableEntry (
    DiameterBaseAPIContext_t* pDiameterContext ,DAPIHANDLE-
    REALMTABLE-ENTRY hRealmTabEntry , DiameterString* pstrRealmName);
```

**Purpose**

This function retrieves the value of the realm name associated with a given realm table object .

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE- REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
pstrRealmName	DiameterString*	Out	This is a pointer to the string in which the matching realm name will be returned.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

## 5.4. GetAppIDFromRealmTableEntry

```
DiamAPIReturnVal GetAppIDFromRealmTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-REALMTABLE-ENTRY hRealmTabEntry ,
ApplicationIdentifier* pAppIdent );
```

### Purpose

This function retrieves the value of the ApplicationID associated with a given real table object.

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
pAppIdent	Application Identifier*	Out	This is a pointer to the variable in which the matching applicationid will be returned.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

## 5.2.5. GetLocalActionFromRealmTableEntry

```
DiamAPIReturnVal GetLocalActionFromRealmTableEntry (
DiameterBaseAPIContext_t* pDiameterContext , DAPIHANDLE-
REALMTABLE-ENTRY hRealmTabEntry ,LocalAction* pLocAct);
```

### Purpose

This function retrieves the value of the local action field from a given real routing table entry

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
pLocAct	LocalAction*	Out	This is a pointer to the variable in which the matching local action

field will be returned.
-------------------------

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 5.6. GetServerListFromRealmTableEntry

*DiamAPIReturnVal GetServerListFromRealmTableEntry ( DiameterBaseAPIContext\_t\* pDiameterContext , DAPIHANDLE-REALMTABLE-ENTRY hRealmTabEntry ,DAPI-Dlist\*\* ppServerList );*

### Purpose

This function retrieves the value of the server list associated with the given realm table entry.

### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
ppServerList	DAPI-Dlist**	Out	This is the pointer to the list which will be filled by the server list.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 5.7. GetStaticStatusFromRealmTableEntry

```
DiamAPIReturnVal GetStaticStatusFromRealmTableEntry (
    DiameterBaseAPIContext_t* pDiameterContext, DAPIHANDLE-
    REALMTABLE-ENTRY hRealmTabEntry, BOOL* bIsStatic);
```

#### Purpose

This function retrieves the value of the static status field from a given realm routing table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
bIsStatic	BOOL*	Out	This is a pointer to a variable in which the static status field will be returned.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 5.8. GetExpirationTimeFromRealmTableEntry

```
DiamAPIReturnVal GetExpirationTimeFromRealmTableEntry (
    DiameterBaseAPIContext_t* pDiameterContext, DAPIHANDLE-
    REALMTABLE-ENTRY hRealmTabEntry, DiaTime_t* pExpiryTime );
```

#### Purpose

This function retrieves the value of the expiration time associated with the given realm routing table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.

pExpiryTime	DiaTime_t*	Out	This is a pointer to the variable in which expiry time value will be returned.
-------------	------------	-----	--

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 5.2.9. SetRealmNameInRealmTableEntry

*DiamAPIRetVal SetRealmNameInRealmTableEntry ( DiameterBaseContext\_t\* pDiameterContext , DAPIHANDLE-REALMTABLE-ENTRY hRealmTabEntry , DiameterString strRealmName);*

#### Purpose

This function sets the value of the realm name for a realm routing table entry

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
strRealmName	DiameterString	In	This is the realm name to be associated with the routing table entry.

### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIRetVal enumeration.

### 5.2.10. SetLocalActionInRealmTableEntry

```
DiamAPIReturnVal SetLocalActionInRealmTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-REALMTABLE-ENTRY hRealmTabEntry
,LocalAction LocAct);
```

#### Purpose

This function sets the Local Action value for a given realm routing table entry

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
LocAct	LocalAction	In	This contains the value of the local action field to be set.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 5.2.11. SetServerListInRealmTableEntry

```
DiamAPIReturnVal SetServerListInRealmTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-REALMTABLE-ENTRY hRealmTabEntry
,DAPI-Dlist* pServerList );
```

#### Purpose

This function sets the server list for a given realm routing table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
pServerList	DAPI-Dlist*	In	This is a pointer to the list containing the server list to be set.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

**5.2.12. SetStaticStatusInRealmTableEntry**

```
DiamAPIReturnVal SetStaticStatusInRealmTableEntry ( DiameterBaseAPIContext_t*
pDiameterContext , DAPIHANDLE-REALMTABLE-ENTRY hRealmTabEntry
,BOOL IsStatic);
```

**Purpose**

This function sets the value of the static status field for a given realm routing table entry.

**Parameters**

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
IsStatic	BOOL	In	This is the value of the IsStatic field to be set.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 5.2.13. Set SetExpirationTimeInRealmTableEntry

```
DiamAPIReturnVal SetExpirationTimeInRealmTableEntry (
    DiameterBaseAPIContext_t* pDiameterContext , DAPIHANDLE-
    REALMTABLE-ENTRY hRealmTabEntry ,DiaTime_t
    ExpiryTime );
```

#### Purpose

This function sets the value of the Expiration Time field in the given realm based routing table entry.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.
ExpiryTime	DiaTime_t	In	This is the value of the Expiration time that needs to be set in the realm table entry.

#### Return value

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

### 5.2.14. InsertEntryIntoRealmRoutingTable

```
DiamAPIReturnVal InsertEntryIntoRealmRoutingTable ( DiameterBaseAPIContext_t*
    pDiameterContext , DAPIHANDLE-REALMTABLE-ENTRY hRealmTabEntry );
```

#### Purpose

This function inserts the given realm routing table entry into the actual realm routing table.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure the context of the diameter base protocol API.
hRealmTabEntry	DAPIHANDLE-REALMTABLE-ENTRY	In	This is the handle to the realm table entry object.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.

## 6) Utility APIs

### 6.1. Global Utility Structures

#### 6.1.1. DiamAPIVersion

```
typedef struct DiamAPIVersion
    {
        unsigned int majVersion;
        unsigned int minVersion;
    } DiamAPIVersion_t;
```

#### Structure Explanation

This structure contains the version information for the diameter base protocol stack

#### Fields:

##### majVersion

The major version

##### minVersion

The minor version.

#### 6.1.2. DiaTime

```
typedef struct DiaTime
    {
        unsigned int hours;
        unsigned int mins;
        unsigned int seconds;
        unsigned int millisecs;
    } DiaTime_t;
```

#### Structure Explanation

This structure represents the value of time

#### Fields

##### hours

Integer representing the number of hours

##### mins

Integer representing the number of minutes

##### seconds

Integer representing the number of seconds

##### millisecs

Integer representing the number of milliseconds

### 6.1.3. strStruct

```
typedef struct strStruct
    {
        char*          pBuff;
        unsigned int   strLength;
    } strStruct_t;
```

#### Structure Explanation

This structure defines a very basic string.

#### Fields:

##### pBuff

The actual buffer containing the string data.

##### strLength

The amount of memory allocated to the string data buffer.

### 6.1.4. DiamAPIRetVal

```
Typedef enum DiamAPIRetVal
    {
        DAPI-Retval-Success,
        DAPI-Retval-Failure,
        DAPI-Retval-IncorrectParameters,
        DAPI-Retval-MoreInfoRequired,
        DAPI-Retval-AccessDenied,
        DAPI-Retval-UnableToOpenDevice,
        DAPI-Retval-Insufficient-Buffer
    } DiamAPIRetVal_t ;
```

### 6.1.5. PeerTable

```
typedef struct PeerTable
    {
    } PeerTable_t;
```

#### Structure Explanation

This structure defines the Peer Table

### 6.1.6. RealmTable

```
typedef struct RealmTable
{
    }RealmTable_t;
```

#### Structure Explanation

This structure defines the Ream Based Routing Table

### 6.1.7. DiameterBaseAPIContext

```
typedef struct DiameterBaseAPIContext
{
    PeerTableObject*          pPeerTable;
    RealmTableObject*        pRealmTable;
    } DiameterBaseAPIContext_t ;
```

#### Structure Explanation

This structure will serve as the main context used through out the Diameter Base Protocol stack

### 6.1.8. Type Defines

```
typedef PeerTable_t          PeerTableObject;
typedef RealmTable_t        RealmTableObject;
typedef DiameterString      DiameterIdentity
typedef DAPI-DList          t_AMPSDList ;
typedef strStruct           DiameterString
```

## 6.2. Utility APIs

### 6.2.1. GetDiameterAPIVersion

```
DiamAPIReturnVal GetDiameterAPIVersion(DiameterBaseAPIContext_t*
pDiameterContext , DiamAPIVersion_t * pDiaVer );
```

#### Purpose

This function retrieves the value of the current version of the diameter base protocol stack.

#### Parameters

Name	Type	In/out	Description
pDiameterContext	DiameterBaseAPIContext_t*	In/Out	This structure contains the context of the diameter base protocol API.
pDiaVer	DiamAPIVersion_t *	Out	This is a pointer to the

variable that will be filled with the version information.

**Return value**

The function returns DAPI-Retval-Success in case of success and DAPI-Retval-Failure in case of failure. Other possible return types can be seen in the help for DiamAPIReturnVal enumeration.