cmpi-router

1.0.0

Generated by Doxygen 1.6.1

Tue Sep 22 18:42:12 2009

Contents

Chapter 1

Todo List

Global datetime_str_interval_to_ms move this function to the proper file.

Class LANEndpoint how can obtain installation date form a device? char installDate[64];

How can obtain device last change. char timeOfLastStateChange[64];

File OSBase_Zebra.c WILL BE USED IN THE FUTURE TO OBTAIN INFO FROM ZEBRA/QUAGGA.

File OSBase_Zebra.h WILL BE USED IN THE FUTURE TO OBTAIN INFO FROM ZEBRA/QUAGGA.

Chapter 2

Data Structure Index

2.1 Data Structures

Here are the data structures with brief descriptions:

Chapter 3

File Index

3.1 File List

Here is a list of all files with brief descriptions:

Chapter 4

Data Structure Documentation

4.1 LANEndpoint Struct Reference

`#include <OSBase_LANEndpoint.h>`

Data Fields

- char aliasAddresses [32]
- unsigned short availReqStates [9]
- char caption [64]
- unsigned short communicationStatus
- char creationClassName [256]
- char description [256]
- unsigned short detailedStatus
- char elementName [64]
- unsigned short enabledDefault
- unsigned short enabledState
- char groupAddresses [32]
- unsigned short healthState
- char instanceID [64]
- char lanID [64]
- char macAddress [32]
- unsigned int maxDataSize
- char name [256]
- char nameFormat [256]
- unsigned short operatingStatus
- unsigned short operationalStatus
- char otherEnabledState [64]
- char otherTypeDescription [64]
- unsigned short primaryStatus
- unsigned short protocolIFType
- unsigned short requestedState

- char statusDescriptions [256]
- char systemCreationClassName [256]
- char systemName [256]
- unsigned short transitioningToState

4.1.1 Detailed Description

This structure stores the LANEndpoint's properties.

Todo

how can obtain installation date form a device? char installDate[64];
How can obtain device last change. char timeOfLastStateChange[64];

Definition at line 300 of file OSBase_LANEndpoint.h.

4.1.2 Field Documentation

4.1.2.1 char aliasAddresses[32]

Other unicast addresses that may be used to communicate with the LANEndpoint.

Definition at line 301 of file OSBase_LANEndpoint.h.

4.1.2.2 unsigned short availReqStates[9]

The possible values for the RequestedState parameter of the method RequestStateChange.

Definition at line 304 of file OSBase_LANEndpoint.h.

4.1.2.3 char caption[64]

Short textual description of the object.

Definition at line 307 of file OSBase_LANEndpoint.h.

4.1.2.4 unsigned short communicationStatus

The ability of the instrumentation to communicate with the underlying ManagedElement.

Definition at line 309 of file OSBase_LANEndpoint.h.

4.1.2.5 char creationClassName[256]

The name of the class or the subclass used in the creation of an instance.

Definition at line 312 of file OSBase_LANEndpoint.h.

4.1.2.6 char description[256]

Textual description of the object.

Definition at line 315 of file OSBase_LANEndpoint.h.

4.1.2.7 unsigned short detailedStatus

Compliments PrimaryStatus with additional status detail.

Definition at line 317 of file OSBase_LANEndpoint.h.

4.1.2.8 char elementName[64]

A user-friendly name for the object.

Definition at line 319 of file OSBase_LANEndpoint.h.

4.1.2.9 unsigned short enabledDefault

Administrator's default configuration for the Enabled State.

Definition at line 321 of file OSBase_LANEndpoint.h.

4.1.2.10 unsigned short enabledState

Indicates the enabled and disabled states of an element.

Definition at line 324 of file OSBase_LANEndpoint.h.

4.1.2.11 char groupAddresses[32]

Multicast addresses to which the LANEndpoint listens.

Definition at line 326 of file OSBase_LANEndpoint.h.

4.1.2.12 unsigned short healthState

Current health of the element.

Definition at line 328 of file OSBase_LANEndpoint.h.

4.1.2.13 char instanceID[64]

Opaquely and uniquely identify an instance of this class.

Definition at line 329 of file OSBase_LANEndpoint.h.

4.1.2.14 char lanID[64]

Identifier for the LAN Segment to which the Endpoint is connected.

Definition at line 331 of file OSBase_LANEndpoint.h.

4.1.2.15 char macAddress[32]

The principal unicast address used in communication with the LANEndpoint.

Definition at line 333 of file OSBase_LANEndpoint.h.

4.1.2.16 unsigned int maxDataSize

The largest information field that may be sent or received by the LANEndpoint.

Definition at line 336 of file OSBase_LANEndpoint.h.

4.1.2.17 char name[256]

Identifies this ProtocolEndpoint.

Definition at line 339 of file OSBase_LANEndpoint.h.

4.1.2.18 char nameFormat[256]

The naming heuristic that is selected to ensure that the value of the Name property is unique.

Definition at line 340 of file OSBase_LANEndpoint.h.

4.1.2.19 unsigned short operatingStatus

Current status value for the operational condition of the element.

Definition at line 343 of file OSBase_LANEndpoint.h.

4.1.2.20 unsigned short operationalStatus

Current statuses of the element.

Definition at line 346 of file OSBase_LANEndpoint.h.

4.1.2.21 char otherEnabledState[64]

A string that describes the enabled or disabled state of the element.

Definition at line 347 of file OSBase_LANEndpoint.h.

4.1.2.22 char otherTypeDescription[64]

Type of ProtocolEndpoint.

Definition at line 350 of file OSBase_LANEndpoint.h.

4.1.2.23 unsigned short primaryStatus

High level status value.

Definition at line 351 of file OSBase_LANEndpoint.h.

4.1.2.24 unsigned short protocolIFType

IANA ifType MIB.

Definition at line 352 of file OSBase_LANEndpoint.h.

4.1.2.25 unsigned short requestedState

The last requested or desired state for the element.

Definition at line 353 of file OSBase_LANEndpoint.h.

4.1.2.26 char statusDescriptions[256]

Strings describing the various OperationalStatus array values.

Definition at line 355 of file OSBase_LANEndpoint.h.

4.1.2.27 char systemCreationClassName[256]

The CreationClassName of the scoping System.

Definition at line 357 of file OSBase_LANEndpoint.h.

4.1.2.28 char systemName[256]

The Name of the scoping System.

Definition at line 359 of file OSBase_LANEndpoint.h.

4.1.2.29 unsigned short transitioningToState

The target state to which the instance is transitioning.

Definition at line 360 of file OSBase_LANEndpoint.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_LANEndpoint.h

4.2 LANEndpointList Struct Reference

`#include <OSBase_LANEndpoint.h>`

Data Fields

- struct LANEndpoint * sptr
- struct LANEndpointList * next

4.2.1 Detailed Description

This structure is used as LANEndpoints list.

Definition at line 367 of file OSBase_LANEndpoint.h.

4.2.2 Field Documentation

4.2.2.1 struct LANEndpointList* next [read]

Pointer to next position in the list.

Definition at line 370 of file OSBase_LANEndpoint.h.

4.2.2.2 struct LANEndpoint* sptr [read]

Pointer to current position in the list.

Definition at line 368 of file OSBase_LANEndpoint.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_LANEndpoint.h

4.3 nextHopIP Struct Reference

`#include <OSBase_NextHopIPRoute.h>`

Data Fields

- unsigned short addressType
- unsigned short adminDistance
- char caption [64]
- char description [256]
- char dstAddress [64]
- char dstMask [64]
- char elementName [64]
- char instanceID [64]
- unsigned short isStatic
- char otherDerivation [64]
- unsigned char prefixLength
- unsigned short routeDerivation
- char routeGateway [64]
- unsigned short routeMetric
- unsigned short routeOutputIf
- unsigned short routeScope
- unsigned short routeTable
- unsigned short routeType
- unsigned short typeOfRoute

4.3.1 Detailed Description

This structure stores the route's properties.

Definition at line 132 of file OSBase_NextHopIPRoute.h.

4.3.2 Field Documentation

4.3.2.1 unsigned short addressType

The format of the address properties.

Definition at line 133 of file OSBase_NextHopIPRoute.h.

4.3.2.2 unsigned short adminDistance

Administrative distance of this route.

Definition at line 135 of file OSBase_NextHopIPRoute.h.

4.3.2.3 char caption[64]

Short textual description of the object.

Definition at line 137 of file OSBase_NextHopIPRoute.h.

4.3.2.4 char description[256]

Textual description of the object.

Definition at line 139 of file OSBase_NextHopIPRoute.h.

4.3.2.5 char dstAddress[64]

Destination address to be reached.

Definition at line 141 of file OSBase_NextHopIPRoute.h.

4.3.2.6 char dstMask[64]

The mask for the destination address.

Definition at line 143 of file OSBase_NextHopIPRoute.h.

4.3.2.7 char elementName[64]

User-friendly name for the object.

Definition at line 145 of file OSBase_NextHopIPRoute.h.

4.3.2.8 char instanceID[64]

Opaquely and uniquely identify an instance of this class.

Definition at line 147 of file OSBase_NextHopIPRoute.h.

4.3.2.9 unsigned short isStatic

TRUE indicates that this is a static route.

Definition at line 149 of file OSBase_NextHopIPRoute.h.

4.3.2.10 char otherDerivation[64]

A string describing how the route was derived.

Definition at line 151 of file OSBase_NextHopIPRoute.h.

4.3.2.11 unsigned char prefixLength

The prefix length for the IPv6 destination address.

Definition at line 153 of file OSBase_NextHopIPRoute.h.

4.3.2.12 unsigned short routeDerivation

How the route was derived.

Definition at line 155 of file OSBase_NextHopIPRoute.h.

4.3.2.13 char routeGateway[64]

The gateway of the route.

Definition at line 156 of file OSBase_NextHopIPRoute.h.

4.3.2.14 unsigned short routeMetric

Numeric indication as to the preference of this route.

Definition at line 157 of file OSBase_NextHopIPRoute.h.

4.3.2.15 unsigned short routeOutputIf

Output interface index.

Definition at line 159 of file OSBase_NextHopIPRoute.h.

4.3.2.16 unsigned short routeScope

Sort of distance to the destination.

Definition at line 160 of file OSBase_NextHopIPRoute.h.

4.3.2.17 unsigned short routeTable

Routing table id.

Definition at line 162 of file OSBase_NextHopIPRoute.h.

4.3.2.18 unsigned short routeType

Type of route.

Definition at line 163 of file OSBase_NextHopIPRoute.h.

4.3.2.19 unsigned short typeOfRoute

Administrator Defined Route, Computed Route or Actual Route.

Definition at line 164 of file OSBase_NextHopIPRoute.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_NextHopIPRoute.h

4.4 nextHopIPList Struct Reference

`#include <OSBase_NextHopIPRoute.h>`

Data Fields

- struct nextHopIP * sptr
- struct nextHopIPList * next

4.4.1 Detailed Description

This structure is used as NextHopIPRoutes list.

Definition at line 171 of file OSBase_NextHopIPRoute.h.

4.4.2 Field Documentation

4.4.2.1 struct nextHopIPList* next [read]

Pointer to next position in the list.

Definition at line 174 of file OSBase_NextHopIPRoute.h.

4.4.2.2 struct nextHopIP* sptr [read]

Pointer to current position in the list.

Definition at line 172 of file OSBase_NextHopIPRoute.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_NextHopIPRoute.h

4.5 nlLinkInfo Struct Reference

`#include <OSBase_Netlink.h>`

Data Fields

- unsigned char family
- unsigned short type
- int index
- unsigned int flags
- unsigned int change
- unsigned char address [32]
- int addressLen
- unsigned char broadcast [32]
- int broadcastLen
- char ifname [IFNAMSIZ]
- unsigned int mtu
- int link
- char qdisc [64]
- struct net_device_stats stats
- int txqlen
- struct ifmap map
- unsigned char operstate
- unsigned char linkmode

4.5.1 Detailed Description

Information of a single link.

Definition at line 226 of file OSBase_Netlink.h.

4.5.2 Field Documentation

4.5.2.1 unsigned char address[32]

Interface L2 address (IFLA_ADDRESS).

Definition at line 235 of file OSBase_Netlink.h.

4.5.2.2 int addressLen

Used to store the length of the address.

Definition at line 237 of file OSBase_Netlink.h.

4.5.2.3 unsigned char broadcast[32]

L2 broadcast address (IFLA_BROADCAST).

Definition at line 239 of file OSBase_Netlink.h.

4.5.2.4 int broadcastLen

Used to store the length of the broadcast.

Definition at line 241 of file OSBase_Netlink.h.

4.5.2.5 unsigned int change

Reserved for future use.

Definition at line 234 of file OSBase_Netlink.h.

4.5.2.6 unsigned char family

AF_UNSPEC (from ifinfomsg).

Definition at line 227 of file OSBase_Netlink.h.

4.5.2.7 unsigned int flags

Device flags (see netdevice(7)).

Definition at line 232 of file OSBase_Netlink.h.

4.5.2.8 char ifname[IFNAMSIZ]

Device name (IFLA_IFNAME).

Definition at line 243 of file OSBase_Netlink.h.

4.5.2.9 int index

Unique interface index.

Definition at line 231 of file OSBase_Netlink.h.

4.5.2.10 int link

Link type (IFLA_LINK).

Definition at line 247 of file OSBase_Netlink.h.

4.5.2.11 unsigned char linkmode

Link mode.

Definition at line 262 of file OSBase_Netlink.h.

4.5.2.12 struct ifmap map [read]

Device mapping structure.

Definition at line 258 of file OSBase_Netlink.h.

4.5.2.13 unsigned int mtu

MTU of the device (IFLA_MTU).

Definition at line 245 of file OSBase_Netlink.h.

4.5.2.14 unsigned char operstate

Operational state (rfc 2863).

Definition at line 260 of file OSBase_Netlink.h.

4.5.2.15 char qdisc[64]

Queueing discipline (IFLA_QDISC).

Definition at line 248 of file OSBase_Netlink.h.

4.5.2.16 struct net_device_stats stats [read]

Interface statistics (IFLA_STATS).

Definition at line 250 of file OSBase_Netlink.h.

4.5.2.17 int txqlen

Tx queue length.

Definition at line 257 of file OSBase_Netlink.h.

4.5.2.18 unsigned short type

Link type - ARPHRD_∗ (from ifinfomsg).

Definition at line 229 of file OSBase_Netlink.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_Netlink.h

4.6 nlLinkInfoList Struct Reference

`#include <OSBase_Netlink.h>`

Data Fields

- struct nlLinkInfo * sptr
- struct nlLinkInfoList * next

4.6.1 Detailed Description

Links container.

Definition at line 273 of file OSBase_Netlink.h.

4.6.2 Field Documentation

4.6.2.1 struct nlLinkInfoList* next [read]

Pointer to next position in the list.

Definition at line 276 of file OSBase_Netlink.h.

4.6.2.2 struct nlLinkInfo* sptr [read]

Pointer to current position in the list.

Definition at line 274 of file OSBase_Netlink.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_Netlink.h

4.7 nlRouteInfo Struct Reference

`#include <OSBase_Netlink.h>`

Data Fields

- int family
- int type
- int protocol
- int scope
- int srcLen
- int dstLen
- int tos
- char dstAddr [64]
- char srcAddr [64]
- int inputIf
- int outputIf
- char gw [64]
- int priority
- char prefSrc [64]
- int metrics
- int table

4.7.1 Detailed Description

Information of a single route.

Definition at line 116 of file OSBase_Netlink.h.

4.7.2 Field Documentation

4.7.2.1 char dstAddr[64]

Destination address to be reached (RTA_DST).

Definition at line 124 of file OSBase_Netlink.h.

4.7.2.2 int dstLen

Destination Route mask prefix (from rtmsg).

Definition at line 122 of file OSBase_Netlink.h.

4.7.2.3 int family

IPv4=AF_INET ; IPv6=AF_INET6 (from rtmsg).

Definition at line 117 of file OSBase_Netlink.h.

4.7.2.4 char gw[64]

Gateway of the route (RTA_GATEWAY).

Definition at line 128 of file OSBase_Netlink.h.

4.7.2.5 int inputIf

Input interface index (RTA_IIF).

Definition at line 126 of file OSBase_Netlink.h.

4.7.2.6 int metrics

Route metrics (RTA_METRICS).

Definition at line 131 of file OSBase_Netlink.h.

4.7.2.7 int outputIf

Output interface index (RTA_OIF).

Definition at line 127 of file OSBase_Netlink.h.

4.7.2.8 char prefSrc[64]

Prefered source (RTA_PREFSRC).

Definition at line 130 of file OSBase_Netlink.h.

4.7.2.9 int priority

Priority of the route (RTA_PRIORITY).

Definition at line 129 of file OSBase_Netlink.h.

4.7.2.10 int protocol

Route origin (from rtmsg).

Definition at line 119 of file OSBase_Netlink.h.

4.7.2.11 int scope

Distance to the destination (from rtmsg).

Definition at line 120 of file OSBase_Netlink.h.

4.7.2.12 char srcAddr[64]

Source address (RTA_SRC).

Definition at line 125 of file OSBase_Netlink.h.

4.7.2.13 int srcLen

Source Route mask prefix (from rtmsg).

Definition at line 121 of file OSBase_Netlink.h.

4.7.2.14 int table

Route table (RTA_TABLE).

Definition at line 132 of file OSBase_Netlink.h.

4.7.2.15 int tos

Type of service (from rtmsg).

Definition at line 123 of file OSBase_Netlink.h.

4.7.2.16 int type

Route type (from rtmsg).

Definition at line 118 of file OSBase_Netlink.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_Netlink.h

4.8 nlRouteInfoList Struct Reference

`#include <OSBase_Netlink.h>`

Data Fields

- struct nlRouteInfo * sptr
- struct nlRouteInfoList * next

4.8.1 Detailed Description

Routes container.

Definition at line 138 of file OSBase_Netlink.h.

4.8.2 Field Documentation

4.8.2.1 struct nlRouteInfoList* next [read]

Pointer to next position in the list.

Definition at line 141 of file OSBase_Netlink.h.

4.8.2.2 struct nlRouteInfo* sptr [read]

Pointer to current position in the list.

Definition at line 139 of file OSBase_Netlink.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_Netlink.h

4.9 nlSockHandle Struct Reference

`#include <OSBase_Netlink.h>`

Data Fields

- int fd
- struct sockaddr_nl local

4.9.1 Detailed Description

Netlink socket handler.

Definition at line 67 of file OSBase_Netlink.h.

4.9.2 Field Documentation

4.9.2.1 int fd

File descriptor for the new socket.

Definition at line 68 of file OSBase_Netlink.h.

4.9.2.2 struct sockaddr_nl local [read]

Netlink client in user-space.

Definition at line 69 of file OSBase_Netlink.h.

The documentation for this struct was generated from the following file:

- /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_Netlink.h

Chapter 5

File Documentation

5.1 /mnt/storage/TESIS/sblim/cmpi-router/include/cmpiOSBase_-LANEndpoint.h File Reference

```
#include "cmpidt.h"
#include "OSBase_LANEndpoint.h"
```

Functions

- CMPIObjectPath * _makePath_LANEndpoint (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIObjectPath *cop, const struct LANEndpoint *pLANEP, CMPIStatus *rc)
- CMPIInstance * _makeInst_LANEndpoint (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIObjectPath *cop, const char **properties, const struct LANEndpoint *pLANEP, CMPIStatus *rc)

Variables

- static char * _ClassName = "Linux_LANEndpoint"

5.1.1 Detailed Description

cmpiOSBase_LANEndpoint.h

THIS FILE IS PROVIDED UNDER THE TERMS OF THE ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THIS FILE CONSTITUTES RECIPIENTS ACCEPTANCE OF THE AGREEMENT.

You can obtain a current copy of the Eclipse Public License from http://www.opensource.org/licenses/eclipse-1.0.php

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This file defines the interfaces for the factory implementation of the CIM class Linux_LANEndpoint.

Interface Type : Common Magabeablity Programming Interface (CMPI)

Definition in file cmpiOSBase_LANEndpoint.h.

5.1.2 Function Documentation

5.1.2.1 CMPIInstance* _makeInst_LANEndpoint (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIObjectPath * *cop*, const char ** *properties*, const struct LANEndpoint * *pLANEP*, CMPIStatus * *rc*)

Method to create a CMPIInstance of this class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] Context object.

cop [in] Source objectPath.

properties [in]

pLANEP [in] LANEndpoint instance.

rc [in] cim status.

Returns:

created LANEndpoint instance.

Definition at line 85 of file cmpiOSBase_LANEndpoint.c.

5.1.2.2 CMPIObjectPath* _makePath_LANEndpoint (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIObjectPath * *cop*, const struct LANEndpoint * *pLANEP*, CMPIStatus * *rc*)

Method to create a CMPIObjectPath of this class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] Context object.

cop [in] Source objectPath.

pLANEP [in] LANEndpoint instance.

rc [in] cim status.

Returns:

created LANEndpoint ObjectPath.

Definition at line 41 of file cmpiOSBase_LANEndpoint.c.

5.1.3 Variable Documentation

5.1.3.1 char* _ClassName = "Linux_LANEndpoint" [static]

Provider ClassName.

Definition at line 33 of file cmpiOSBase_LANEndpoint.h.

5.2 /mnt/storage/TESIS/sblim/cmpi-router/include/cmpiOSBase_-NextHopIPRoute.h File Reference

```
#include "cmpidt.h"
#include "OSBase_NextHopIPRoute.h"
```

Functions

- CMPIObjectPath * _makePath_NextHopIPRoute (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIObjectPath *cop, const struct nextHopIP *pNHop, CMPIStatus *rc)
- CMPIInstance * _makeInst_NextHopIPRoute (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIObjectPath *cop, const char **properties, const struct nextHopIP *pNHop, CMPIStatus *rc)

Variables

- static char * _ClassName = "Linux_NextHopIPRoute"

5.2.1 Detailed Description

cmpiOSBase_NextHopIPRoute.h

THIS FILE IS PROVIDED UNDER THE TERMS OF THE ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THIS FILE CONSTITUTES RECIPIENTS ACCEPTANCE OF THE AGREEMENT.

You can obtain a current copy of the Eclipse Public License from http://www.opensource.org/licenses/eclipse-1.0.php

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This file defines the interfaces for the factory implementation of the CIM class Linux_NextHopIPRoute.

Interface Type : Common Magabeablity Programming Interface (CMPI)

Definition in file cmpiOSBase_NextHopIPRoute.h.

5.2.2 Function Documentation

5.2.2.1 CMPIInstance* _makeInst_NextHopIPRoute (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIObjectPath * *cop*, const char ** *properties*, const struct nextHopIP * *pNHop*, CMPIStatus * *rc*)

Method to create a CMPIInstance of this class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] Context object.

cop [in] Source objectPath.

properties [in]

pNHop [in] nextHopIP instance.

rc [in] cim status.

Returns:

created NextHopIPRoute instance.

Definition at line 82 of file cmpiOSBase_NextHopIPRoute.c.

5.2.2.2 CMPIObjectPath* _makePath_NextHopIPRoute (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIObjectPath * *cop*, const struct nextHopIP * *pNHop*, CMPIStatus * *rc*)

Method to create a CMPIObjectPath of this class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] Context object.

cop [in] Source objectPath.

pNHop [in] nextHopIP instance.

rc [in] cim status.

Returns:

created NextHopIPRoute ObjectPath.

Definition at line 42 of file cmpiOSBase_NextHopIPRoute.c.

5.2.3 Variable Documentation

5.2.3.1 char* _ClassName = "Linux_NextHopIPRoute" [static]

Provider ClassName.

Definition at line 33 of file cmpiOSBase_NextHopIPRoute.h.

5.3 /mnt/storage/TESIS/sblim/cmpi-router/include/cmpiOSBase_-RouteUsesEndpoint.h File Reference

```
#include "cmpidt.h"
```

Enumerations

- enum { ATYPE_ASSOC, ATYPE_ASSOCN, ATYPE_REFER, ATYPE_REFERN }

Functions

- CMPIObjectPath * _makePath_RouteUsesEndpoint (const CMPIBroker *_broker, const CMPIObjectPath *ops, const CMPIObjectPath *opt, CMPIStatus *rc)
- CMPIInstance * _makeInst_RouteUsesEndpoint (const CMPIBroker *_broker, const CMPIObjectPath *ops, const CMPIObjectPath *opt, CMPIStatus *rc)
- CMPIStatus * _assoc_get_NextHopRoute_insts (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const CMPIInstance *sourceInst, const char *targetClass, int assocType, CMPIStatus *rc)
- CMPIStatus * _assoc_get_ProtocolEndpoint_insts (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const CMPIInstance *sourceInst, const char *targetClass, int assocType, CMPIStatus *rc)
- CMPIStatus _assoc_RouteUsesEndpoint (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char *targetClass, int assocType, CMPIStatus *rc)

Variables

- static char * _ClassName = "Linux_RouteUsesEndpoint"
- static char * _RefLeft = "Antecedent"
- static char * _RefRight = "Dependent"
- static char * _RefLeftClass = "CIM_ProtocolEndpoint"
- static char * _RefRightClass = "CIM_NextHopRoute"
- static char * _RefLeftClasses [ ] = { "Linux_LANEndpoint" }
- static char * _RefRightClasses [ ] = { "Linux_NextHopIPRoute" }

5.3.1 Detailed Description

cmpiOSBase_RouteUsesEndpoint.h

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This is the factory implementation for creating instances of CIM class Linux_RouteUsesEndpoint.

Interface Type : Common Magabeablity Programming Interface (CMPI)

Definition in file cmpiOSBase_RouteUsesEndpoint.h.

5.3.2 Enumeration Type Documentation

5.3.2.1 anonymous enum

Association type.

Enumerator:

ATYPE_ASSOC

ATYPE_ASSOCN

ATYPE_REFER

ATYPE_REFERN

Definition at line 32 of file cmpiOSBase_RouteUsesEndpoint.h.

5.3.3 Function Documentation

5.3.3.1 CMPIStatus* _assoc_get_NextHopRoute_insts (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const CMPIInstance * *sourceInst*, const char * *targetClass*, int *assocType*, CMPIStatus * *rc*)

Retrieve a list of instances from target class (CIM_NextHopRoute subclass), associated to source class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] context object.

rslt [in] result.

ref [in] source objectPath.

sourceInst [in] source class instance.

targetClass [in] target class name.
assocType [in] association type.
rc [in] cim status.

Returns:

cim status.

Definition at line 120 of file cmpiOSBase_RouteUsesEndpoint.c.

5.3.3.2 CMPIStatus* _assoc_get_ProtocolEndpoint_insts (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const CMPIInstance * *sourceInst*, const char * *targetClass*, int *assocType*, CMPIStatus * *rc*)

Retrieve a list of instances from target class (CIM_ProtocolEndpoint subclass), associated to source class.

Parameters:

_broker [in] CIM Object Manager.
ctx [in] context object.
rslt [in] result.
ref [in] source objectPath.
sourceInst [in] source class instance.
targetClass [in] target class name.
assocType [in] association type.
rc [in] cim status.

Returns:

cim status.

Definition at line 242 of file cmpiOSBase_RouteUsesEndpoint.c.

5.3.3.3 CMPIStatus _assoc_RouteUsesEndpoint (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char * *targetClass*, int *assocType*, CMPIStatus * *rc*)

Create CMPIInstances of association.

Parameters:

_broker [in] CIM Object Manager.
ctx [in] context object.
rslt [in] result.
ref [in] source objectPath.
targetClass [in] target class name.
assocType [in] association type.
rc [in] cim status.

Returns:

cim status.

Definition at line 368 of file cmpiOSBase_RouteUsesEndpoint.c.

5.3.3.4 CMPIInstance* _makeInst_RouteUsesEndpoint (const CMPIBroker * *_broker*, const CMPIObjectPath * *ops*, const CMPIObjectPath * *opt*, CMPIStatus * *rc*)

Method to create a CMPIInstance of this class.

Parameters:

_broker [in] CIM Object Manager.

ops [in] Source objectPath.

opt [in] Target objectPath.

rc [in] cim status.

Returns:

created Instance.

Definition at line 79 of file cmpiOSBase_RouteUsesEndpoint.c.

5.3.3.5 CMPIObjectPath* _makePath_RouteUsesEndpoint (const CMPIBroker * *_broker*, const CMPIObjectPath * *ops*, const CMPIObjectPath * *opt*, CMPIStatus * *rc*)

Method to create a CMPIObjectPath of this class.

Parameters:

_broker [in] CIM Object Manager.

ops [in] Source objectPath.

opt [in] Target objectPath.

rc [in] cim status.

Returns:

created ObjectPath.

Definition at line 41 of file cmpiOSBase_RouteUsesEndpoint.c.

5.3.4 Variable Documentation

5.3.4.1 char* _ClassName = "Linux_RouteUsesEndpoint" [static]

Definition at line 39 of file cmpiOSBase_RouteUsesEndpoint.h.

5.3.4.2 char* _RefLeft = "Antecedent" [static]

Definition at line 40 of file cmpiOSBase_RouteUsesEndpoint.h.

5.3.4.3 char* _RefLeftClass = "CIM_ProtocolEndpoint" [static]

Definition at line 42 of file cmpiOSBase_RouteUsesEndpoint.h.

5.3.4.4 char* _RefLeftClasses[ ] = { "Linux_LANEndpoint" } [static]

Definition at line 44 of file cmpiOSBase_RouteUsesEndpoint.h.

5.3.4.5 char* _RefRight = "Dependent" [static]

Definition at line 41 of file cmpiOSBase_RouteUsesEndpoint.h.

5.3.4.6 char* _RefRightClass = "CIM_NextHopRoute" [static]

Definition at line 43 of file cmpiOSBase_RouteUsesEndpoint.h.

5.3.4.7 char* _RefRightClasses[ ] = { "Linux_NextHopIPRoute" } [static]

Definition at line 45 of file cmpiOSBase_RouteUsesEndpoint.h.

5.4 /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_-LANEndpoint.h File Reference

```
#include "OSBase_Netlink.h"
```

Data Structures

- struct LANEndpoint
- struct LANEndpointList

Defines

- #define MAXHOSTNAMELEN 256
- #define LANENDPOINT_CAPTION "LAN EndPoint"
- #define CREATION_CLASS_NAME "Linux_LANEndpoint"
- #define LANENDPOINT_DESC

Enumerations

- enum {

  LEP_ES_UNKNOWN, LEP_ES_OTHER, LEP_ES_ENABLED, LEP_ES_DISABLED,

  LEP_ES_SHUTTING_DOWN, LEP_ES_NOT_APPLICABLE, LEP_ES_ENABLED_BUT_OFFLINE, LEP_ES_IN_TEST,

  LEP_ES_DEFERRED, LEP_ES_QUIESCE, LEP_ES_STARTING }
- enum {

  LEP_ED_ENABLED = 2, LEP_ED_DISABLED, LEP_ED_NOT_APPLICABLE = 5, LEP_ED_ENABLED_BUT_OFFLINE,

  LEP_ED_NO_DEFAULT, LEP_ED_QUIESCE = 9 }
- enum {

  LEP_RS_UNKNOWN, LEP_RS_ENABLED = 2, LEP_RS_DISABLED, LEP_RS_SHUT_DOWN,

  LEP_RS_NO_CHANGE, LEP_RS_OFFLINE, LEP_RS_TEST, LEP_RS_DEFERRED,

  LEP_RS_QUIESCE, LEP_RS_REBOOT, LEP_RS_RESET, LEP_RS_NOT_APPLICABLE }
- enum {

  LEP_CS_UNKNOWN, LEP_CS_NOT_AVAILABLE, LEP_CS_COMMUNICATION_OK, LEP_CS_LOST_COMMUNICATION,

  LEP_CS_NO_CONTACT }
- enum {

  LEP_DS_NOT_AVAILABLE, LEP_DS_NO_ADDITIONAL_INFORMATION, LEP_DS_STRESSED, LEP_DS_PREDICTIVE_FAILURE,

  LEP_DS_NON_RECOVERABLE_ERROR, LEP_DS_SUPPORTING_ENTITY_IN_ERROR }

- enum {
  LEP_HS_UNKNOWN = 0, LEP_HS_OK = 5, LEP_HS_DEGRADED_WARNING = 10, LEP_HS_MINOR_FAILURE = 15,
  LEP_HS_MAJOR_FAILURE = 20, LEP_HS_CRITICAL_FAILURE = 25, LEP_HS_NON_RECOVERABLE_ERROR = 30 }
- enum {
  LEP_OS_UNKNOWN, LEP_OS_NOT_AVAILABLE, LEP_OS_SERVICING, LEP_OS_STARTING,
  LEP_OS_STOPPING, LEP_OS_STOPPED, LEP_OS_ABORTED, LEP_OS_DORMANT,
  LEP_OS_COMPLETED, LEP_OS_MIGRATING, LEP_OS_EMIGRATING, LEP_OS_IMMIGRATING,
  LEP_OS_SNAPSHOTTING, LEP_OS_SHUTTING_DOWN, LEP_OS_IN_TEST, LEP_OS_TRANSITIONING,
  LEP_OS_IN_SERVICE }
- enum {
  LEP_OPS_UNKNOWN, LEP_OPS_OTHER, LEP_OPS_OK, LEP_OPS_DEGRADED,
  LEP_OPS_STRESSED, LEP_OPS_PREDICTIVE_FAILURE, LEP_OPS_ERROR, LEP_OPS_NON_RECOVERABLE_ERROR,
  LEP_OPS_STARTING, LEP_OPS_STOPPING, LEP_OPS_STOPPED, LEP_OPS_IN_SERVICE,
  LEP_OPS_NO_CONTACT, LEP_OPS_LOST_COMMUNICATION, LEP_OPS_ABORTED, LEP_OPS_DORMANT,
  LEP_OPS_SUPPORTING_ENTITY_IN_ERROR, LEP_OPS_COMPLETED, LEP_OPS_POWER_MODE }
- enum { LEP_PS_UNKNOWN, LEP_PS_OK, LEP_PS_DEGRADED, LEP_PS_ERROR }

Functions

- int getLANEndpoints (struct LANEndpointList **lanEPList, const struct nlLinkInfo *nlLinkInfo)
- int getALLLANEndpoint (struct LANEndpointList **lanEPList)
- int getLANEndpoint (struct LANEndpointList **lanEPList, const char *linkName)
- unsigned short changeLinkOPState (const char *linkName, unsigned short enabledState, unsigned long timeoutPeriod)
- int nlListTOLanEPList (struct nlLinkInfoList **nlLinkInfoList, struct LANEndpointList **lanEPList)
- int nlInfoTOLanEP (struct nlLinkInfo *nlLinkInfo, struct LANEndpoint *lanEP)
- void freeLANEndpointList (struct LANEndpointList *lptr)
- void freeLANEndpoint (struct LANEndpoint *sptr)

5.4.1 Detailed Description

OSBase_LANEndpoint.h

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This file defines the interfaces for the resource access implementation of the CIM class Linux_-LANEndpoint.

Definition in file OSBase_LANEndpoint.h.

5.4.2 Define Documentation

5.4.2.1 #define CREATION_CLASS_NAME "Linux_LANEndpoint"

CIM - Creation class name.

Definition at line 48 of file OSBase_LANEndpoint.h.

5.4.2.2 #define LANENDPOINT_CAPTION "LAN EndPoint"

CIM - LANEndpoint caption.

Definition at line 43 of file OSBase_LANEndpoint.h.

5.4.2.3 #define LANENDPOINT_DESC

Value:

```
"A communication endpoint which, when "        \
    "its associated interface device is connected to a LAN, "          \
    "may send and receive data frames."
```

CIM - LANEndpoint desctiption.

Definition at line 53 of file OSBase_LANEndpoint.h.

5.4.2.4 #define MAXHOSTNAMELEN 256

Maximum hostName length.

Definition at line 37 of file OSBase_LANEndpoint.h.

5.4.3 Enumeration Type Documentation

5.4.3.1 anonymous enum

CIM - EnabledState.

Enumerator:

LEP_ES_UNKNOWN

LEP_ES_OTHER

LEP_ES_ENABLED Indicates that the element is or could be executing commands, will process any queued commands, and queues new requests.

LEP_ES_DISABLED Indicates that the element will not execute commands and will drop any new requests.

LEP_ES_SHUTTING_DOWN Indicates that the element is in the process of going to a Disabled state.

LEP_ES_NOT_APPLICABLE Indicates the element does not support being enabled or disabled.

LEP_ES_ENABLED_BUT_OFFLINE Indicates that the element might be completing commands, and will drop any new requests.

LEP_ES_IN_TEST Indicates that the element is in a test state.

LEP_ES_DEFERRED Indicates that the element might be completing commands, but will queue any new requests.

LEP_ES_QUIESCE Indicates that the element is enabled but in a restricted mode.

LEP_ES_STARTING Indicates that the element is in the process of going to an Enabled state.

Definition at line 60 of file OSBase_LANEndpoint.h.

5.4.3.2 anonymous enum

CIM - EnabledDefault.

Enumerator:

LEP_ED_ENABLED Default.

LEP_ED_DISABLED

LEP_ED_NOT_APPLICABLE

LEP_ED_ENABLED_BUT_OFFLINE

LEP_ED_NO_DEFAULT

LEP_ED_QUIESCE

Definition at line 91 of file OSBase_LANEndpoint.h.

5.4.3.3 anonymous enum

CIM - RequestedState & TransitioningToState.

Enumerator:

LEP_RS_UNKNOWN Indicates the last requested state for the element is unknown.

LEP_RS_ENABLED

LEP_RS_DISABLED

LEP_RS_SHUT_DOWN

LEP_RS_NO_CHANGE Deprecated.

LEP_RS_OFFLINE Change to "Enabled but Offline" status.

LEP_RS_TEST

LEP_RS_DEFERRED

LEP_RS_QUIESCE

LEP_RS_REBOOT Reboot refers to doing a "Shut Down" and then moving to an "Enabled" state.

LEP_RS_RESET Reset indicates that the element is first "Disabled" and then "Enabled".

LEP_RS_NOT_APPLICABLE

Definition at line 103 of file OSBase_LANEndpoint.h.

5.4.3.4 anonymous enum

CIM - CommunicationStatus.

Enumerator:

LEP_CS_UNKNOWN Indicates the implementation is in general capable of returning this property, but is unable to do so at this time.

LEP_CS_NOT_AVAILABLE Indicates that the implementation is capable of returning a value for this property, but not ever for this particular piece of hardware/software.

LEP_CS_COMMUNICATION_OK Indicates communication is established with the element, but does not convey any quality of service.

LEP_CS_LOST_COMMUNICATION Indicates that the Managed Element is known to exist and has been contacted successfully in the past, but is currently unreachable.

LEP_CS_NO_CONTACT Indicates that the monitoring system has knowledge of this element, but has never been able to establish communications with it.

Definition at line 124 of file OSBase_LANEndpoint.h.

5.4.3.5 anonymous enum

CIM - DetailedStatus.

Enumerator:

LEP_DS_NOT_AVAILABLE Indicates that the implementation is capable of returning a value for this property, but not ever for this particular piece of hardware/software.

LEP_DS_NO_ADDITIONAL_INFORMATION Indicates that the element is functioning normally as indicated by PrimaryStatus = "OK".

LEP_DS_STRESSED Indicates that the element is functioning, but needs attention.

LEP_DS_PREDICTIVE_FAILURE Indicates that an element is functioning normally but a failure is predicted in the near future.

LEP_DS_NON_RECOVERABLE_ERROR Indicates that this element is in an error condition that requires human intervention.

LEP_DS_SUPPORTING_ENTITY_IN_ERROR Indicates that this element might be "OK" but that another element, on which it is dependent, is in error.

Definition at line 147 of file OSBase_LANEndpoint.h.

5.4.3.6 anonymous enum

CIM - HealthState.

Enumerator:

LEP_HS_UNKNOWN The implementation cannot report on HealthState at this time.

LEP_HS_OK The element is fully functional and is operating within normal operational parameters and without error.

LEP_HS_DEGRADED_WARNING The element is in working order and all functionality is provided. However, the element is not working to the best of its abilities.

LEP_HS_MINOR_FAILURE All functionality is available but some might be degraded.

LEP_HS_MAJOR_FAILURE The element is failing. It is possible that some or all of the functionality of this component is degraded or not working.

LEP_HS_CRITICAL_FAILURE The element is non-functional and recovery might not be possible.

LEP_HS_NON_RECOVERABLE_ERROR The element has completely failed, and recovery is not possible.

Definition at line 173 of file OSBase_LANEndpoint.h.

5.4.3.7 anonymous enum

CIM - OperatingStatus.

Enumerator:

LEP_OS_UNKNOWN Indicates the implementation is in general capable of returning this property, but is unable to do so at this time.

LEP_OS_NOT_AVAILABLE Indicates that the implementation is capable of returning a value for this property, but not ever for this particular piece of hardware/software.

LEP_OS_SERVICING Describes an element being configured, maintained, cleaned, or otherwise administered.

LEP_OS_STARTING Describes an element being initialized.

LEP_OS_STOPPING Describes an element being brought to an orderly stop.

LEP_OS_STOPPED Clean and orderly stop.

LEP_OS_ABORTED Abrupt stop.

LEP_OS_DORMANT Indicates that the element is inactive or quiesced.

LEP_OS_COMPLETED Indicates that the element has completed its operation.

LEP_OS_MIGRATING Element is being moved between host elements.

LEP_OS_EMIGRATING Element is being moved away from host element.

LEP_OS_IMMIGRATING Element is being moved to new host element.

LEP_OS_SNAPSHOTTING

LEP_OS_SHUTTING_DOWN Describes an element being brought to an abrupt stop.

LEP_OS_IN_TEST Element is performing test functions.

LEP_OS_TRANSITIONING Describes an element that is between states.

LEP_OS_IN_SERVICE Describes an element that is in service and operational.

Definition at line 199 of file OSBase_LANEndpoint.h.

5.4.3.8 anonymous enum

CIM - OperationalStatus.

Enumerator:

LEP_OPS_UNKNOWN

LEP_OPS_OTHER Other status.

LEP_OPS_OK The element is fully functional and is operating within normal operational.

LEP_OPS_DEGRADED The element is in working order and all functionality is provided. However, the element is not working to the best of its abilities.

LEP_OPS_STRESSED The element is functioning, but needs attention.

LEP_OPS_PREDICTIVE_FAILURE Element is functioning nominally but predicting a failure in the near future.

LEP_OPS_ERROR In error state.

LEP_OPS_NON_RECOVERABLE_ERROR Indicates that this element is in an error condition that requires human intervention.

LEP_OPS_STARTING Indicates that the element is in the process of going to an Enabled state.

LEP_OPS_STOPPING Describes an element being brought to an orderly stop.

LEP_OPS_STOPPED Clean and orderly stop.

LEP_OPS_IN_SERVICE Element being configured, maintained, cleaned, or otherwise administered.

LEP_OPS_NO_CONTACT The monitoring system has knowledge of this element, but has never been able to establish communications with it.

LEP_OPS_LOST_COMMUNICATION The ManagedSystem Element is known to exist and has been contacted successfully in the past, but is currently unreachable.

LEP_OPS_ABORTED Abrupt stop.

LEP_OPS_DORMANT The element is inactive or quiesced.

LEP_OPS_SUPPORTING_ENTITY_IN_ERROR This element might be "OK" but that another element, on which it is dependent, is in error.

LEP_OPS_COMPLETED Tthe element has completed its operation.

LEP_OPS_POWER_MODE The element has additional power model information.

Definition at line 235 of file OSBase_LANEndpoint.h.

5.4.3.9 anonymous enum

CIM - PrimaryStatus.

Enumerator:

LEP_PS_UNKNOWN

LEP_PS_OK

LEP_PS_DEGRADED

LEP_PS_ERROR

Definition at line 286 of file OSBase_LANEndpoint.h.

5.4.4 Function Documentation

5.4.4.1 unsigned short changeLinkOPState (const char * *linkName*, unsigned short *enabledState*, unsigned long *timeoutPeriod*)

Used to set device status.

Parameters:

linkName [in] link name.

enabledState [in] desired state for the device.

timeoutPeriod [in] maximum amount of time that the client expects the transition to the new state to take.

Returns:

0 = Completed with No Error 1 = Not Supported 2 = Unknown or Unspecified Error 3 = Cannot complete within Timeout Period 4 = Failed 5 = Invalid Parameter 6 = In Use 7..4095 = DMTF Reserved 4096 = Method Parameters Checked - Job Started 4097 = Invalid State Transition 4098 = Use of Timeout Parameter Not Supported 4099 = Busy 4100..32767 = Method Reserved 32768..65535 = Vendor Specific

Definition at line 157 of file OSBase_LANEndpoint.c.

5.4.4.2 void freeLANEndpoint (struct LANEndpoint * *sptr*)

This function is used to clean a LANEndpoint structure.

Parameters:

sptr [in] structure to be cleaned.

Definition at line 501 of file OSBase_LANEndpoint.c.

5.4.4.3 void freeLANEndpointList (struct LANEndpointList * *lptr*)

This function is used to clean a LANEndpoint list.

Parameters:

lptr [in] list to be cleaned.

Definition at line 478 of file OSBase_LANEndpoint.c.

5.4.4.4 int getALLLANEndpoint (struct LANEndpointList ** *lanEPList*)

This functions is used to get whole LANEndpoints available in the system.

Parameters:

lanEPList [out] LANEndpoint struct to be filled.

Returns:

0=succesful | 1=fail

5.4.4.5 int getLANEndpoint (struct LANEndpointList ** *lanEPList*, const char * *linkName*)

This functions is used to get a route base on specified InstanceID.

Parameters:

lanEPList [out] LANEndpoint struct to be filled.

linkName [in] link ifname.

Returns:

0=succesful | 1=fail

Definition at line 130 of file OSBase_LANEndpoint.c.

5.4.4.6 int getLANEndpoints (struct LANEndpointList ** *lanEPList*, const struct nlLinkInfo * *nlLinkInfo*)

This functions is used to get a list of LANEndpoints based on nlLinkInfo filter.

Parameters:

lanEPList [out] LANEndpoint struct to be filled.

nlLinkInfo [in] link filter info.

Returns:

0=succesful | 1=fail

Definition at line 58 of file OSBase_LANEndpoint.c.

5.4.4.7 int nlInfoTOLanEP (struct nlLinkInfo * *nlLinkInfo*, struct LANEndpoint * *lanEP*)

Converts a nlLinkInfo structure to LANEndpoint structure.

Parameters:

nlLinkInfo [in] structure to be converted.

lanEP [out] converted structure.

Returns:

0=succesful | 1=fail

Definition at line 287 of file OSBase_LANEndpoint.c.

5.4.4.8 int nlListTOLanEPList (struct nlLinkInfoList ** *nlLinkInfoList*, struct LANEndpointList ** *lanEPList*)

Converts a nlLinkInfoList structure to LANEndpointList strucutre.

Parameters:

nlLinkInfoList [in] list to be converted.

lanEPList [out] converted list.

Returns:

0=succesful | 1=fail

Definition at line 244 of file OSBase_LANEndpoint.c.

5.5 /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_-Netlink.h File Reference

```
#include <sys/socket.h>
#include <linux/netlink.h>
#include <linux/netdevice.h>
#include <linux/if_arp.h>
```

Data Structures

- struct nlSockHandle
- struct nlRouteInfo
- struct nlRouteInfoList
- struct nlLinkInfo
- struct nlLinkInfoList

Defines

- #define NL_SOCK_SND_BUFF_LEN 32768
- #define NL_SOCK_RCV_BUFF_LEN 32768
- #define ARRAY_SIZE(arr) (sizeof(arr) / sizeof((arr)[0]))
- #define FREE_SAFE(ptr) { if(ptr != NULL) free(ptr); ptr = NULL; }
- #define NLMSG_TAIL(nmsg) ((struct rtattr∗) (((void∗) (nmsg)) + NLMSG_ALIGN((nmsg)->nlmsg_len)))

Functions

- int nlOpenSocket ()
- int nlCloseSocket ()
- int nlAddAttrToMsg (struct nlmsghdr ∗h, int maxlen, int attrType, const void ∗attrData, int attrBytelen)
- int nlAddAttrToMsg32 (struct nlmsghdr ∗h, int maxlen, int attrType, unsigned int attrData)
- int nlGetRoutes (struct nlRouteInfoList ∗∗nlRtInfoList, struct nlRouteInfo ∗nlRtInfo)
- int nlModifyRoute (struct nlRouteInfo ∗nlRtInfo, int hType, unsigned int hFlags)
- int nlAddRouteToList (const struct nlmsghdr ∗rcvH, struct nlRouteInfoList ∗∗nlRtInfoList, struct nlRouteInfo ∗nlRtInfo)
- int nlGenRouteFilter (struct nlRouteInfo ∗nlRtInfo)
- void nlResetRouteFilter ()
- int nlCreateDefaultRtInfo (struct nlRouteInfo ∗nlRtInfo)
- int nlGetLinks (struct nlLinkInfoList ∗∗nlLinkInfoList, const struct nlLinkInfo ∗nlLinkInfo)
- int nlModifyLink (struct nlLinkInfo ∗nlLinkInfo, int hType, unsigned int hFlags)
- int nlAddLinkToList (const struct nlmsghdr ∗rcvH, struct nlLinkInfoList ∗∗nlLinkInfoList, const struct nlLinkInfo ∗nlLinkInfo)
- int nlGenLinkFilter (const struct nlLinkInfo ∗nlLinkInfo)

- void nlResetLinkFilter ()
- int nlCreateDefaultLinkInfo (struct nlLinkInfo *nlLinkInfo)
- void nlAddr_n2a (const unsigned char *addr, int alen, int type, char *buf, int blen)
- unsigned short nlGetLinkTypePos (unsigned short devType)

Variables

- struct {
  unsigned int family: 1
  unsigned int type: 1
  unsigned int protocol: 1
  unsigned int scope: 1
  unsigned int srcLen: 1
  unsigned int dstLen: 1
  unsigned int tos: 1
  unsigned int dstAddr: 1
  unsigned int srcAddr: 1
  unsigned int inputIf: 1
  unsigned int outputIf: 1
  unsigned int gw: 1
  unsigned int priority: 1
  unsigned int prefSrc: 1
  unsigned int metrics: 1
  unsigned int table: 1
  } rtFlt

- struct {
  unsigned int family: 1
  unsigned int type: 1
  unsigned int index: 1
  unsigned int flags: 1
  unsigned int change: 1
  unsigned int address: 1
  unsigned int broadcast: 1
  unsigned int ifname: 1
  unsigned int mtu: 1
  unsigned int link: 1
  unsigned int qdisc: 1
  unsigned int stats: 1
  unsigned int txqlen: 1
  unsigned int map: 1
  unsigned int operstate: 1
  unsigned int linkmode: 1
  } linkFlt

- static const unsigned short linkType [ ]
- static const char * linkTypeName [ ]

5.5.1 Detailed Description

OSBase_NetLink.h

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

Netlink support lib.

Definition in file OSBase_Netlink.h.

5.5.2 Define Documentation

5.5.2.1 #define ARRAY_SIZE(arr) (sizeof(arr) / sizeof((arr)[0]))

Get the size of an array.

Definition at line 51 of file OSBase_Netlink.h.

5.5.2.2 #define FREE_SAFE(ptr) { if(ptr != NULL) free(ptr); ptr = NULL; }

Used to safe clean pointers.

Definition at line 56 of file OSBase_Netlink.h.

5.5.2.3 #define NL_SOCK_RCV_BUFF_LEN 32768

Netlink socket receive buffer length.

Definition at line 46 of file OSBase_Netlink.h.

5.5.2.4 #define NL_SOCK_SND_BUFF_LEN 32768

Netlink socket send buffer length.

Definition at line 41 of file OSBase_Netlink.h.

5.5.2.5 #define NLMSG_TAIL(nmsg) ((struct rtattr∗) (((void∗) (nmsg)) + NLMSG_ALIGN((nmsg)->nlmsg_len)))

Netlink attributes.

Definition at line 61 of file OSBase_Netlink.h.

5.5.3 Function Documentation

5.5.3.1 int nlAddAttrToMsg (struct nlmsghdr * *h*, int *maxlen*, int *attrType*, const void * *attrData*, int *attrBytelen*)

This function is used to add an attribute to a netlink message.

Parameters:

h [out] message.

maxlen [in] message length.

attrType [in] attribute type.

attrData [in] attribute data.

attrBytelen [in] attribute length.

Returns:

0=succesful | 1=fail

Definition at line 147 of file OSBase_Netlink.c.

5.5.3.2 int nlAddAttrToMsg32 (struct nlmsghdr * *h*, int *maxlen*, int *attrType*, unsigned int *attrData*)

This function is used to add an attribute to a netlink message.

Parameters:

h [out] message.

maxlen [in] message length.

attrType [in] attribute type.

attrData [in] attribute data.

Returns:

0=succesful | 1=fail

Definition at line 175 of file OSBase_Netlink.c.

5.5.3.3 int nlAddLinkToList (const struct nlmsghdr * *rcvH*, struct nlLinkInfoList ** *nlLinkInfoList*, const struct nlLinkInfo * *nlLinkInfo*)

This function is used to add a link to specified links list.

Parameters:

rcvH [in] link message.

nlLinkInfoList [out] list where link will be added.

nlLinkInfo [in] link info.

Returns:

0=succesful | 1=fail

Definition at line 1490 of file OSBase_Netlink.c.

5.5.3.4 void nlAddr_n2a (const unsigned char * *addr*, int *alen*, int *type*, char * *buf*, int *blen*)

Function to convert a L2 address from network representation to string representation (user friendly).

Parameters:

addr [in] L2 address to convert.

alen [in] L2 address length.

type [in] link type.

buf [out] converted address.

blen [in] converted address maximum length.

Definition at line 1892 of file OSBase_Netlink.c.

5.5.3.5 int nlAddRouteToList (const struct nlmsghdr * *rcvH*, struct nlRouteInfoList ** *nlRtInfoList*, struct nlRouteInfo * *nlRtInfo*)

This function is used to add a route to specified routes list.

Parameters:

rcvH [in] route message.

nlRtInfoList [out] list where route will be added.

nlRtInfo [in] route info.

Returns:

0=succesful | 1=fail

Definition at line 659 of file OSBase_Netlink.c.

5.5.3.6 int nlCloseSocket ()

Used to close previously open Netlink socket.

Returns:

0=succesful | 1=fail

Definition at line 124 of file OSBase_Netlink.c.

5.5.3.7 int nlCreateDefaultLinkInfo (struct nlLinkInfo * *nlLinkInfo*)

Used to create a nlLinkInfo structure with default values.

Parameters:

nlLinkInfo [out] link structure.

Returns:

0=succesful | 1=fail

Definition at line 1855 of file OSBase_Netlink.c.

5.5.3.8 int nlCreateDefaultRtInfo (struct nlRouteInfo * *nlRtInfo*)

Used to create a nlRouteInfo structure with default values.

Parameters:

nlRtInfo [out] route structure.

Returns:

0=succesful | 1=fail

Definition at line 1069 of file OSBase_Netlink.c.

5.5.3.9 int nlGenLinkFilter (const struct nlLinkInfo * *nlLinkInfo*)

Used to generate a filter to be applied to a list of links.

Parameters:

nlLinkInfo [in] link info.

Returns:

0=succesful | 1=fail

Definition at line 1700 of file OSBase_Netlink.c.

5.5.3.10 int nlGenRouteFilter (struct nlRouteInfo * *nlRtInfo*)

Used to generate a filter to be applied to a list of routes.

Parameters:

nlRtInfo [in] route info.

Returns:

0=succesful | 1=fail

Definition at line 851 of file OSBase_Netlink.c.

5.5.3.11 int nlGetLinks (struct nlLinkInfoList ** *nlLinkInfoList*, const struct nlLinkInfo * *nlLinkInfo*)

This function is used to get links using nlLinkInfo as filter.

Parameters:

nlLinkInfoList [out] links list (one or more entries).

nlLinkInfo [in] link info.

Returns:

0=succesful | 1=fail

Definition at line 1110 of file OSBase_Netlink.c.

5.5.3.12 unsigned short nlGetLinkTypePos (unsigned short *devType*)

Lookup the position of the passed type into linkType array.

Parameters:

devType [in] type to find.

Returns:

position into linkType array.

Definition at line 1929 of file OSBase_Netlink.c.

5.5.3.13 int nlGetRoutes (struct nlRouteInfoList ** *nlRtInfoList*, struct nlRouteInfo * *nlRtInfo*)

This function is used to get routes using nlRtInfo as filter.

Parameters:

nlRtInfoList [out] routes list (one or more entries).

nlRtInfo [in] route info.

Returns:

0=succesful | 1=fail

Definition at line 209 of file OSBase_Netlink.c.

5.5.3.14 int nlModifyLink (struct nlLinkInfo * *nlLinkInfo*, int *hType*, unsigned int *hFlags*)

This function is used to ADD/DELETE/MODIFY a link.

Parameters:

nlLinkInfo [in] link info.

hType [in] type of message.

hFlags [in] message flags.

Returns:

0=succesful | 1=fail

Definition at line 1279 of file OSBase_Netlink.c.

5.5.3.15 int nlModifyRoute (struct nlRouteInfo * *nlRtInfo*, int *hType*, unsigned int *hFlags*)

This function is used to ADD/DELETE/MODIFY a route.

Parameters:

nlRtInfo [in] route info.
hType [in] type of message.
hFlags [in] message flags.

Returns:

0=succesful | 1=fail

Definition at line 384 of file OSBase_Netlink.c.

5.5.3.16 int nlOpenSocket ()

Used to open a Netlink socket.

Returns:

0=succesful | 1=fail

Definition at line 48 of file OSBase_Netlink.c.

5.5.3.17 void nlResetLinkFilter ()

Used to reset link filter handler.

Definition at line 1825 of file OSBase_Netlink.c.

5.5.3.18 void nlResetRouteFilter ()

Used to reset route filter handler.

Definition at line 1042 of file OSBase_Netlink.c.

5.5.4 Variable Documentation

5.5.4.1 unsigned int address

Definition at line 290 of file OSBase_Netlink.h.

5.5.4.2 unsigned int broadcast

Definition at line 291 of file OSBase_Netlink.h.

5.5.4.3 unsigned int change

Definition at line 289 of file OSBase_Netlink.h.

5.5.4.4 unsigned int dstAddr

Definition at line 157 of file OSBase_Netlink.h.

5.5.4.5 unsigned int dstLen

Definition at line 155 of file OSBase_Netlink.h.

5.5.4.6 unsigned int family

Definition at line 150 of file OSBase_Netlink.h.

5.5.4.7 unsigned int flags

Definition at line 288 of file OSBase_Netlink.h.

5.5.4.8 unsigned int gw

Definition at line 161 of file OSBase_Netlink.h.

5.5.4.9 unsigned int ifname

Definition at line 292 of file OSBase_Netlink.h.

5.5.4.10 unsigned int index

Definition at line 287 of file OSBase_Netlink.h.

5.5.4.11 unsigned int inputIf

Definition at line 159 of file OSBase_Netlink.h.

5.5.4.12 unsigned int link

Definition at line 294 of file OSBase_Netlink.h.

5.5.4.13 struct { ... } linkFlt

Struct used as link filter. If a field is deactivated (= 0), it will be skiped.

5.5.4.14 unsigned int linkmode

Definition at line 306 of file OSBase_Netlink.h.

5.5.4.15 const unsigned short linkType[ ] [static]

Initial value:

```
{
        ARPHRD_NETROM, ARPHRD_ETHER, ARPHRD_EETHER, ARPHRD_AX25,
        ARPHRD_PRONET, ARPHRD_CHAOS, ARPHRD_IEEE802, ARPHRD_ARCNET,
        ARPHRD_APPLETLK, ARPHRD_DLCI, ARPHRD_ATM, ARPHRD_METRICOM,
        ARPHRD_IEEE1394, ARPHRD_EUI64, ARPHRD_INFINIBAND, ARPHRD_SLIP,
        ARPHRD_CSLIP, ARPHRD_SLIP6, ARPHRD_CSLIP6, ARPHRD_RSRVD,
        ARPHRD_ADAPT, ARPHRD_ROSE, ARPHRD_X25, ARPHRD_HWX25, ARPHRD_CAN,
        ARPHRD_PPP, ARPHRD_CISCO, ARPHRD_HDLC, ARPHRD_LAPB, ARPHRD_DDCMP,
        ARPHRD_RAWHDLC, ARPHRD_TUNNEL, ARPHRD_TUNNEL6, ARPHRD_FRAD,
        ARPHRD_SKIP, ARPHRD_LOOPBACK, ARPHRD_LOCALTLK, ARPHRD_FDDI,
        ARPHRD_BIF, ARPHRD_SIT, ARPHRD_IPDDP, ARPHRD_IPGRE,
        ARPHRD_PIMREG, ARPHRD_HIPPI, ARPHRD_ASH, ARPHRD_ECONET,
        ARPHRD_IRDA, ARPHRD_FCPP, ARPHRD_FCAL, ARPHRD_FCPL,
        ARPHRD_FCFABRIC, ARPHRD_IEEE802_TR, ARPHRD_IEEE80211,
        ARPHRD_IEEE80211_PRISM, ARPHRD_IEEE80211_RADIOTAP, ARPHRD_VOID,
        ARPHRD_NONE
    }
```

The list of link types. (taked from kernel/net/core/dev.c)

Definition at line 318 of file OSBase_Netlink.h.

5.5.4.16 const char* linkTypeName[ ] [static]

Initial value:

```
{
        "NETROM", "ETHER", "EETHER", "AX25",
        "PRONET", "CHAOS", "IEEE802", "ARCNET",
        "APPLETLK", "DLCI", "ATM", "METRICOM",
        "IEEE1394", "EUI64", "INFINIBAND", "SLIP",
        "CSLIP", "SLIP6", "CSLIP6", "RSRVD",
        "ADAPT", "ROSE", "X25", "HWX25", "CAN"
        "PPP", "CISCO", "HDLC", "LAPB", "DDCMP",
        "RAWHDLC", "TUNNEL", "TUNNEL6", "FRAD",
        "SKIP", "LOOPBACK", "LOCALTLK", "FDDI",
        "BIF", "SIT", "IPDDP", "IPGRE",
        "PIMREG", "HIPPI", "ASH", "ECONET",
        "IRDA", "FCPP", "FCAL", "FCPL",
        "FCFABRIC", "IEEE802_TR", "IEEE80211",
        "IEEE80211_PRISM", "IEEE80211_RADIOTAP",
        "VOID", "NONE"
    }
```

The list of link types string representations. (taked from kernel/net/core/dev.c)

Definition at line 340 of file OSBase_Netlink.h.

5.5.4.17 unsigned int map

Definition at line 303 of file OSBase_Netlink.h.

5.5.4.18 unsigned int metrics

Definition at line 164 of file OSBase_Netlink.h.

5.5.4.19 unsigned int mtu

Definition at line 293 of file OSBase_Netlink.h.

5.5.4.20 unsigned int operstate

Definition at line 305 of file OSBase_Netlink.h.

5.5.4.21 unsigned int outputIf

Definition at line 160 of file OSBase_Netlink.h.

5.5.4.22 unsigned int prefSrc

Definition at line 163 of file OSBase_Netlink.h.

5.5.4.23 unsigned int priority

Definition at line 162 of file OSBase_Netlink.h.

5.5.4.24 unsigned int protocol

Definition at line 152 of file OSBase_Netlink.h.

5.5.4.25 unsigned int qdisc

Definition at line 295 of file OSBase_Netlink.h.

5.5.4.26 struct { ... } rtFlt

Struct used as route filter. If a field is deactivated (= 0), it will be skiped.

5.5.4.27 unsigned int scope

Definition at line 153 of file OSBase_Netlink.h.

5.5.4.28 unsigned int srcAddr

Definition at line 158 of file OSBase_Netlink.h.

5.5.4.29 unsigned int srcLen

Definition at line 154 of file OSBase_Netlink.h.

5.5.4.30 unsigned int stats

Definition at line 296 of file OSBase_Netlink.h.

5.5.4.31 unsigned int table

Definition at line 165 of file OSBase_Netlink.h.

5.5.4.32 unsigned int tos

Definition at line 156 of file OSBase_Netlink.h.

5.5.4.33 unsigned int txqlen

Definition at line 302 of file OSBase_Netlink.h.

5.5.4.34 unsigned int type

Definition at line 151 of file OSBase_Netlink.h.

5.6 /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_-NextHopIPRoute.h File Reference

```
#include "OSBase_Netlink.h"
```

Data Structures

- struct nextHopIP
- struct nextHopIPList

Defines

- #define INSTANCEID_FORMAT "%s|%s|%d|%d|%d|%d|%s"
- #define INSTANCEID_FORMAT_PARSE "%[^|]|%[^|]|%d|%d|%d|%d|%[^|]"

Enumerations

- enum { NH_AT_UNKNOWN, NH_AT_IPV4, NH_AT_IPV6 }
- enum {
  NH_RTD_UNKNOWN, NH_RTD_OTHER, NH_RTD_CONNECTED, NH_RTD_USER_DEFINED,
  NH_RTD_IGRP, NH_RTD_EIGRP, NH_RTD_RIP, NH_RTD_HELLO,
  NH_RTD_EGP, NH_RTD_BGP, NH_RTD_ISIS, NH_RTD_OSPF }
- enum {
  NH_RTS_UNIVERSE = 0, NH_RTS_SITE = 200, NH_RTS_LINK = 253, NH_RTS_HOST = 254,
  NH_RTS_NOWHERE = 255 }
- enum {
  NH_RTT_UNSPEC = 0, NH_RTT_COMPAT = 252, NH_RTT_DEFAULT = 253, NH_RTT_MAIN = 254,
  NH_RTT_LOCAL = 255 }
- enum {
  NH_RTTY_UNSPEC, NH_RTTY_UNICAST, NH_RTTY_LOCAL, NH_RTTY_BROADCAST,
  NH_RTTY_ANYCAST, NH_RTTY_MULTICAST, NH_RTTY_BLACKHOLE, NH_RTTY_UNREACHABLE,
  NH_RTTY_PROHIBIT, NH_RTTY_THROW, NH_RTTY_NAT, NH_RTTY_XRESOLVE }
- enum { NH_TOR_ADMINISTRATOR = 2, NH_TOR_COMPUTED, NH_TOR_ACTUAL }

Functions

- int getIPRoutes (struct nextHopIPList **nHopIPList, struct nlRouteInfo *nlRtInfo)
- int getAllIPRoutes (struct nextHopIPList **nHopIPList)
- int getIPRouteIId (struct nextHopIPList **nHopIPList, char *instanceId)
- int addIPRoute (struct nextHopIP *nHopIP)

- int delIPRoute (struct nextHopIP *nHopIP)
- int nlListTOnhList (struct nlRouteInfoList **nlRtInfoList, struct nextHopIPList **nHopIPList)
- int nhTOnlInfo (struct nextHopIP *nHopIP, struct nlRouteInfo *nlRtInfo)
- int nlInfoTOnh (struct nlRouteInfo *nlRtInfo, struct nextHopIP *nHopIP)
- int getAddrType (const char *addr)
- int getAddrTypeStr (int addressType, char *addrTypeStr)
- void freeNextHopIPList (struct nextHopIPList *lptr)
- void freeNextHopIP (struct nextHopIP *sptr)

5.6.1 Detailed Description

OSBase_NextHopIPRoute.h

THIS FILE IS PROVIDED UNDER THE TERMS OF THE ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THIS FILE CONSTITUTES RECIPIENTS ACCEPTANCE OF THE AGREEMENT.

You can obtain a current copy of the Eclipse Public License from http://www.opensource.org/licenses/eclipse-1.0.php

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This file defines the interfaces for the resource access implementation of the CIM class Linux_NextHopIPRoute.

Definition in file OSBase_NextHopIPRoute.h.

5.6.2 Define Documentation

5.6.2.1 #define INSTANCEID_FORMAT "%s|%s|%d|%d|%d|%d|%s"

Maximum hostName length. InstanceID format. It is supposed that it will be not changed. hostName|dstAddr|dstLen|table|outputIf|scope|gw

Definition at line 44 of file OSBase_NextHopIPRoute.h.

5.6.2.2 #define INSTANCEID_FORMAT_PARSE "%[^|]|%[^|]|%d|%d|%d|%d|%[^|]"

InstanceID format for parse. It is supposed that it will be not changed.

Definition at line 49 of file OSBase_NextHopIPRoute.h.

5.6.3 Enumeration Type Documentation

5.6.3.1 anonymous enum

CIM - Describes the format of the address properties.

Enumerator:

NH_AT_UNKNOWN Unidentified.
NH_AT_IPV4 IP version 4.
NH_AT_IPV6 IP version 6.

Definition at line 54 of file OSBase_NextHopIPRoute.h.

5.6.3.2 anonymous enum

CIM - How the route was derived.

Enumerator:

NH_RTD_UNKNOWN
NH_RTD_OTHER
NH_RTD_CONNECTED
NH_RTD_USER_DEFINED
NH_RTD_IGRP
NH_RTD_EIGRP
NH_RTD_RIP
NH_RTD_HELLO
NH_RTD_EGP
NH_RTD_BGP
NH_RTD_ISIS
NH_RTD_OSPF

Definition at line 63 of file OSBase_NextHopIPRoute.h.

5.6.3.3 anonymous enum

CIM - Route scope.

Enumerator:

NH_RTS_UNIVERSE
NH_RTS_SITE
NH_RTS_LINK
NH_RTS_HOST
NH_RTS_NOWHERE

Definition at line 81 of file OSBase_NextHopIPRoute.h.

5.6.3.4 anonymous enum

CIM - Routing table.

Enumerator:

NH_RTT_UNSPEC

NH_RTT_COMPAT

NH_RTT_DEFAULT

NH_RTT_MAIN Main kernel table.

NH_RTT_LOCAL

Definition at line 92 of file OSBase_NextHopIPRoute.h.

5.6.3.5 anonymous enum

CIM - Route Type.

Enumerator:

NH_RTTY_UNSPEC

NH_RTTY_UNICAST Gateway or direct route.

NH_RTTY_LOCAL Accept locally.

NH_RTTY_BROADCAST Accept locally as broadcast, send as broadcast.

NH_RTTY_ANYCAST Accept locally as broadcast, but send as unicast.

NH_RTTY_MULTICAST Multicast route.

NH_RTTY_BLACKHOLE Drop.

NH_RTTY_UNREACHABLE Destination is unreachable.

NH_RTTY_PROHIBIT Administratively prohibited.

NH_RTTY_THROW Not in this table.

NH_RTTY_NAT Translate this address.

NH_RTTY_XRESOLVE Use external resolver.

Definition at line 103 of file OSBase_NextHopIPRoute.h.

5.6.3.6 anonymous enum

CIM - Type of route.

Enumerator:

NH_TOR_ADMINISTRATOR

NH_TOR_COMPUTED

NH_TOR_ACTUAL

Definition at line 123 of file OSBase_NextHopIPRoute.h.

5.6.4 Function Documentation

5.6.4.1 int addIPRoute (struct nextHopIP * *nHopIP*)

This function is used to add a new route to FIB.

Parameters:

nHopIP [in] route to be added.

Returns:

0=succesful | 1=fail

Definition at line 143 of file OSBase_NextHopIPRoute.c.

5.6.4.2 int delIPRoute (struct nextHopIP * *nHopIP*)

This function is used to delete a route from FIB.

Parameters:

nHopIP [in] route to be deleted.

Returns:

0=succesful | 1=fail

Definition at line 186 of file OSBase_NextHopIPRoute.c.

5.6.4.3 void freeNextHopIP (struct nextHopIP * *sptr*)

This function is used to clean a nextHopIP structure.

Parameters:

sptr [in] structure to be cleaned.

Definition at line 583 of file OSBase_NextHopIPRoute.c.

5.6.4.4 void freeNextHopIPList (struct nextHopIPList * *lptr*)

This function is used to clean a nextHopIP list.

Parameters:

lptr [in] list to be cleaned.

Definition at line 560 of file OSBase_NextHopIPRoute.c.

5.6.4.5 int getAddrType (const char * *addr*)

Used to get CIM address type from specified route address.

Parameters:

addr [int] route address.

Returns:

CIM address type.

Definition at line 521 of file OSBase_NextHopIPRoute.c.

5.6.4.6 int getAddrTypeStr (int *addressType*, char * *addrTypeStr*)

Used to get CIM address type in string format.

Parameters:

addressType [in] address type identifier.
addrTypeStr [out] address type string.

Returns:

0=succesful | 1=fail

Definition at line 540 of file OSBase_NextHopIPRoute.c.

5.6.4.7 int getAllIPRoutes (struct nextHopIPList ** *nHopIPList*)

This functions is used to get all ipv4 and ipv6 routes from fib.

Parameters:

nHopIPList [out] routes struct to be filled.

Returns:

0=succesful | 1=fail

Definition at line 77 of file OSBase_NextHopIPRoute.c.

5.6.4.8 int getIPRouteIId (struct nextHopIPList ** *nHopIPList*, char * *instanceId*)

This functions is used to get a route base on specified InstanceID.

Parameters:

nHopIPList [out] routes struct to be filled.
instanceId [in] route InstanceID.

Returns:

0=succesful | 1=fail

Definition at line 110 of file OSBase_NextHopIPRoute.c.

5.6.4.9 int getIPRoutes (struct nextHopIPList ** *nHopIPList*, struct nlRouteInfo * *nlRtInfo*)

This functions is used to get a list of routes based on nlRtInfo filter.

Parameters:

nHopIPList [out] routes struct to be filled.

nlRtInfo [in] route info.

Returns:

0=succesful | 1=fail

Definition at line 31 of file OSBase_NextHopIPRoute.c.

5.6.4.10 int nhTOnlInfo (struct nextHopIP * *nHopIP*, struct nlRouteInfo * *nlRtInfo*)

Converts a nextHopIP structure to nlRouteInfo structure.

Parameters:

nHopIP [in] structure to be converted.

nlRtInfo [out] converted structure.

Returns:

0=succesful | 1=fail

Definition at line 273 of file OSBase_NextHopIPRoute.c.

5.6.4.11 int nlInfoTOnh (struct nlRouteInfo * *nlRtInfo*, struct nextHopIP * *nHopIP*)

Converts a nlRouteInfo structure to nextHopIP structure.

Parameters:

nlRtInfo [in] structure to be converted.

nHopIP [out] converted structure.

Returns:

0=succesful | 1=fail

Definition at line 381 of file OSBase_NextHopIPRoute.c.

5.6.4.12 int nlListTOnhList (struct nlRouteInfoList ** *nlRtInfoList*, struct nextHopIPList ** *nHopIPList*)

Converts a nlRouteInfoList structure to nextHopIPList strucutre.

Parameters:

nlRtInfoList [in] list to be converted.

nHopIPList [out] converted list.

Returns:

0=succesful | 1=fail

Definition at line 233 of file OSBase_NextHopIPRoute.c.

5.7 /mnt/storage/TESIS/sblim/cmpi-router/include/OSBase_-Zebra.h File Reference

```
#include <sys/socket.h>
```

5.7.1 Detailed Description

OSBase_Zebra.h

THIS FILE IS PROVIDED UNDER THE TERMS OF THE ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THIS FILE CONSTITUTES RECIPIENTS ACCEPTANCE OF THE AGREEMENT.

You can obtain a current copy of the Eclipse Public License from http://www.opensource.org/licenses/eclipse-1.0.php

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

Todo

WILL BE USED IN THE FUTURE TO OBTAIN INFO FROM ZEBRA/QUAGGA.

Definition in file OSBase_Zebra.h.

5.8 /mnt/storage/TESIS/sblim/cmpi-router/src/cmpiOSBase_CSHostedRouteProvider.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "cmpidt.h"
#include "cmpift.h"
#include "cmpimacs.h"
#include "OSBase_Common.h"
#include "cmpiOSBase_Common.h"
```

Functions

- CMPIStatus OSBase_CSHostedRouteProviderCleanup (CMPIInstanceMI *mi, const CMPIContext *ctx, CMPIBoolean terminating)
- CMPIStatus OSBase_CSHostedRouteProviderEnumInstanceNames (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref)
- CMPIStatus OSBase_CSHostedRouteProviderEnumInstances (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char **properties)
- CMPIStatus OSBase_CSHostedRouteProviderGetInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char **properties)
- CMPIStatus OSBase_CSHostedRouteProviderCreateInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const CMPIInstance *ci)
- CMPIStatus OSBase_CSHostedRouteProviderSetInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const CMPIInstance *ci, const char **properties)
- CMPIStatus OSBase_CSHostedRouteProviderDeleteInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop)
- CMPIStatus OSBase_CSHostedRouteProviderExecQuery (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char *lang, const char *query)
- CMPIStatus OSBase_CSHostedRouteProviderAssociationCleanup (CMPIAssociationMI *mi, const CMPIContext *ctx, CMPIBoolean terminating)
- CMPIStatus OSBase_CSHostedRouteProviderAssociators (CMPIAssociationMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char *assocClass, const char *resultClass, const char *role, const char *resultRole, const char **propertyList)
- CMPIStatus OSBase_CSHostedRouteProviderAssociatorNames (CMPIAssociationMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char *assocClass, const char *resultClass, const char *role, const char *resultRole)
- CMPIStatus OSBase_CSHostedRouteProviderReferences (CMPIAssociationMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char *assocClass, const char *role, const char **propertyList)
- CMPIStatus OSBase_CSHostedRouteProviderReferenceNames (CMPIAssociationMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char *assocClass, const char *role)

- CMInstanceMIStub (OSBase_CSHostedRouteProvider, OSBase_CSHostedRouteProvider, _broker, CMNoHook)
- CMAssociationMIStub (OSBase_CSHostedRouteProvider, OSBase_CSHostedRouteProvider, _broker, CMNoHook)

Variables

- static const CMPIBroker * _broker
- static char * _ClassName = "Linux_CSHostedRoute"
- static char * _RefLeft = "Antecedent"
- static char * _RefRight = "Dependent"
- static char * _RefLeftClass = "Linux_ComputerSystem"
- static char * _RefRightClass = "CIM_NextHopRoute"

5.8.1 Detailed Description

cmpiOSBase_CSHostedRouteProvider.c

THIS FILE IS PROVIDED UNDER THE TERMS OF THE ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THIS FILE CONSTITUTES RECIPIENTS ACCEPTANCE OF THE AGREEMENT.

You can obtain a current copy of the Eclipse Public License from http://www.opensource.org/licenses/eclipse-1.0.php

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

Linux_CSHostedRoute association class provider implementation. The following CMPI instance methods are supported:

- Cleanup
- EnumerateInstanceNames
- EnumerateInstances
- GetInstance
- AssociationCleanup
- Associators
- AssociatorNames
- References
- ReferenceNames

Interface Type : Common Manageability Programming Interface (CMPI)

Definition in file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2 Function Documentation

5.8.2.1 CMAssociationMIStub (OSBase_CSHostedRouteProvider, OSBase_CSHostedRouteProvider, _broker, CMNoHook)

5.8.2.2 CMInstanceMIStub (OSBase_CSHostedRouteProvider, OSBase_CSHostedRouteProvider, _broker, CMNoHook)

5.8.2.3 CMPIStatus OSBase_CSHostedRouteProviderAssociationCleanup (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, CMPIBoolean *terminating*)

Definition at line 273 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.4 CMPIStatus OSBase_CSHostedRouteProviderAssociatorNames (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char * *assocClass*, const char * *resultClass*, const char * *role*, const char * *resultRole*)

Definition at line 378 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.5 CMPIStatus OSBase_CSHostedRouteProviderAssociators (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char * *assocClass*, const char * *resultClass*, const char * *role*, const char * *resultRole*, const char ** *propertyList*)

Definition at line 286 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.6 CMPIStatus OSBase_CSHostedRouteProviderCleanup (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, CMPIBoolean *terminating*)

Definition at line 65 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.7 CMPIStatus OSBase_CSHostedRouteProviderCreateInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const CMPIInstance * *ci*)

Definition at line 189 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.8 CMPIStatus OSBase_CSHostedRouteProviderDeleteInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*)

Definition at line 229 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.9 CMPIStatus OSBase_CSHostedRouteProviderEnumInstanceNames (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*)

Definition at line 78 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.10 CMPIStatus OSBase_CSHostedRouteProviderEnumInstances (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char ** *properties*)

Definition at line 115 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.11 CMPIStatus OSBase_CSHostedRouteProviderExecQuery (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char * *lang*, const char * *query*)

Definition at line 247 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.12 CMPIStatus OSBase_CSHostedRouteProviderGetInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char ** *properties*)

Definition at line 152 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.13 CMPIStatus OSBase_CSHostedRouteProviderReferenceNames (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char * *assocClass*, const char * *role*)

Definition at line 534 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.14 CMPIStatus OSBase_CSHostedRouteProviderReferences (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char * *assocClass*, const char * *role*, const char ** *propertyList*)

Definition at line 469 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.2.15 CMPIStatus OSBase_CSHostedRouteProviderSetInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const CMPIInstance * *ci*, const char ** *properties*)

Definition at line 208 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.3 Variable Documentation

5.8.3.1 const CMPIBroker* _broker `[static]`

Definition at line 45 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.3.2 char* _ClassName = "Linux_CSHostedRoute" [static]

Definition at line 51 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.3.3 char* _RefLeft = "Antecedent" [static]

Definition at line 52 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.3.4 char* _RefLeftClass = "Linux_ComputerSystem" [static]

Definition at line 54 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.3.5 char* _RefRight = "Dependent" [static]

Definition at line 53 of file cmpiOSBase_CSHostedRouteProvider.c.

5.8.3.6 char* _RefRightClass = "CIM_NextHopRoute" [static]

Definition at line 55 of file cmpiOSBase_CSHostedRouteProvider.c.

5.9 /mnt/storage/TESIS/sblim/cmpi-router/src/cmpiOSBase_-LANEndpoint.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "cmpidt.h"
#include "cmpimacs.h"
#include "cmpiOSBase_Common.h"
#include "OSBase_Common.h"
#include "cmpiOSBase_LANEndpoint.h"
#include "OSBase_LANEndpoint.h"
```

Functions

- CMPIObjectPath * _makePath_LANEndpoint (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIObjectPath *cop, const struct LANEndpoint *pLANEP, CMPIStatus *rc)
- CMPIInstance * _makeInst_LANEndpoint (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIObjectPath *cop, const char **properties, const struct LANEndpoint *pLANEP, CMPIStatus *rc)

5.9.1 Detailed Description

cmpiOSBase_LANEndpoint.c

THIS FILE IS PROVIDED UNDER THE TERMS OF THE ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THIS FILE CONSTITUTES RECIPIENTS ACCEPTANCE OF THE AGREEMENT.

You can obtain a current copy of the Eclipse Public License from http://www.opensource.org/licenses/eclipse-1.0.php

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This is the factory implementation for creating instances of CIM class Linux_LANEndpoint.

Interface Type : Common Magabeablity Programming Interface (CMPI)

Definition in file cmpiOSBase_LANEndpoint.c.

5.9.2 Function Documentation

5.9.2.1 CMPIInstance* _makeInst_LANEndpoint (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIObjectPath * *cop*, const char ** *properties*, const struct LANEndpoint * *pLANEP*, CMPIStatus * *rc*)

Method to create a CMPIInstance of this class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] Context object.

cop [in] Source objectPath.

properties [in]

pLANEP [in] LANEndpoint instance.

rc [in] cim status.

Returns:

created LANEndpoint instance.

Definition at line 85 of file cmpiOSBase_LANEndpoint.c.

5.9.2.2 CMPIObjectPath* _makePath_LANEndpoint (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIObjectPath * *cop*, const struct LANEndpoint * *pLANEP*, CMPIStatus * *rc*)

Method to create a CMPIObjectPath of this class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] Context object.

cop [in] Source objectPath.

pLANEP [in] LANEndpoint instance.

rc [in] cim status.

Returns:

created LANEndpoint ObjectPath.

Definition at line 41 of file cmpiOSBase_LANEndpoint.c.

5.10 /mnt/storage/TESIS/sblim/cmpi-router/src/cmpiOSBase_-LANEndpointProvider.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "cmpidt.h"
#include "cmpift.h"
#include "cmpimacs.h"
#include "OSBase_Common.h"
#include "cmpiOSBase_Common.h"
#include "cmpiOSBase_LANEndpoint.h"
```

Functions

- CMPIStatus OSBase_LANEndpointProviderCleanup (CMPIInstanceMI *mi, const CMPIContext *ctx, CMPIBoolean terminating)
- CMPIStatus OSBase_LANEndpointProviderEnumInstanceNames (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref)
- CMPIStatus OSBase_LANEndpointProviderEnumInstances (CMPIInstanceMI *mi, const CMPI-Context *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char **properties)
- CMPIStatus OSBase_LANEndpointProviderGetInstance (CMPIInstanceMI *mi, const CMPICon-text *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char **properties)
- CMPIStatus OSBase_LANEndpointProviderCreateInstance (CMPIInstanceMI *mi, const CMPI-Context *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const CMPIInstance *ci)
- CMPIStatus OSBase_LANEndpointProviderSetInstance (CMPIInstanceMI *mi, const CMPICon-text *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const CMPIInstance *ci, const char **properties)
- CMPIStatus OSBase_LANEndpointProviderDeleteInstance (CMPIInstanceMI *mi, const CMPI-Context *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop)
- CMPIStatus OSBase_LANEndpointProviderExecQuery (CMPIInstanceMI *mi, const CMPICon-text *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char *lang, const char *query)
- CMPIStatus OSBase_LANEndpointProviderMethodCleanup (CMPIMethodMI *mi, const CMPI-Context *ctx, CMPIBoolean terminating)
- CMPIStatus OSBase_LANEndpointProviderInvokeMethod (CMPIMethodMI *mi, const CMPI-Context *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char *methodName, const CMPIArgs *in, CMPIArgs *out)
- CMInstanceMIStub (OSBase_LANEndpointProvider, OSBase_LANEndpointProvider, _broker, CMNoHook)
- CMMethodMIStub (OSBase_LANEndpointProvider, OSBase_LANEndpointProvider, _broker, CMNoHook)

Variables

- static const CMPIBroker * _broker

5.10.1 Detailed Description

cmpiOSBase_LANEndpointProvider.c

THIS FILE IS PROVIDED UNDER THE TERMS OF THE ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THIS FILE CONSTITUTES RECIPIENTS ACCEPTANCE OF THE AGREEMENT.

You can obtain a current copy of the Eclipse Public License from http://www.opensource.org/licenses/eclipse-1.0.php

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

Linux_LANEndpoint class provider implementation. The following CMPI instance methods are supported:

- Cleanup
- EnumerateInstanceNames
- EnumerateInstances
- GetInstance
- MethodCleanup
- InvokeMethod

Interface Type : Common Manageability Programming Interface (CMPI)

Definition in file cmpiOSBase_LANEndpointProvider.c.

5.10.2 Function Documentation

5.10.2.1 CMInstanceMIStub (OSBase_LANEndpointProvider, OSBase_LANEndpointProvider, _broker, CMNoHook)

5.10.2.2 CMMethodMIStub (OSBase_LANEndpointProvider, OSBase_LANEndpointProvider, _broker, CMNoHook)

5.10.2.3 CMPIStatus OSBase_LANEndpointProviderCleanup (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, CMPIBoolean *terminating*)

Definition at line 55 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.4 CMPIStatus OSBase_LANEndpointProviderCreateInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const CMPIInstance * *ci*)

Definition at line 264 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.5 CMPIStatus OSBase_LANEndpointProviderDeleteInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*)

Definition at line 303 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.6 CMPIStatus OSBase_LANEndpointProviderEnumInstanceNames (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*)

Definition at line 68 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.7 CMPIStatus OSBase_LANEndpointProviderEnumInstances (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char ** *properties*)

Definition at line 132 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.8 CMPIStatus OSBase_LANEndpointProviderExecQuery (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char * *lang*, const char * *query*)

Definition at line 321 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.9 CMPIStatus OSBase_LANEndpointProviderGetInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char ** *properties*)

Definition at line 198 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.10 CMPIStatus OSBase_LANEndpointProviderInvokeMethod (CMPIMethodMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char * *methodName*, const CMPIArgs * *in*, CMPIArgs * *out*)

Definition at line 360 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.11 CMPIStatus OSBase_LANEndpointProviderMethodCleanup (CMPIMethodMI * *mi*, const CMPIContext * *ctx*, CMPIBoolean *terminating*)

Definition at line 347 of file cmpiOSBase_LANEndpointProvider.c.

5.10.2.12 CMPIStatus OSBase_LANEndpointProviderSetInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const CMPIInstance * *ci*, const char ** *properties*)

Definition at line 283 of file cmpiOSBase_LANEndpointProvider.c.

5.10.3 Variable Documentation

5.10.3.1 const CMPIBroker* _broker [static]

Definition at line 41 of file cmpiOSBase_LANEndpointProvider.c.

5.11 /mnt/storage/TESIS/sblim/cmpi-router/src/cmpiOSBase_-NextHopIPRoute.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "cmpidt.h"
#include "cmpimacs.h"
#include "cmpiOSBase_Common.h"
#include "OSBase_Common.h"
#include "cmpiOSBase_NextHopIPRoute.h"
#include "OSBase_NextHopIPRoute.h"
```

Functions

- CMPIObjectPath * _makePath_NextHopIPRoute (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIObjectPath *cop, const struct nextHopIP *pNHop, CMPIStatus *rc)
- CMPIInstance * _makeInst_NextHopIPRoute (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIObjectPath *cop, const char **properties, const struct nextHopIP *pNHop, CMPIStatus *rc)

5.11.1 Detailed Description

cmpiOSBase_NextHopIPRoute.c

THIS FILE IS PROVIDED UNDER THE TERMS OF THE ECLIPSE PUBLIC LICENSE ("AGREEMENT"). ANY USE, REPRODUCTION OR DISTRIBUTION OF THIS FILE CONSTITUTES RECIPIENTS ACCEPTANCE OF THE AGREEMENT.

You can obtain a current copy of the Eclipse Public License from http://www.opensource.org/licenses/eclipse-1.0.php

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This is the factory implementation for creating instances of CIM class Linux_NextHopIPRoute.

Interface Type : Common Magabeablity Programming Interface (CMPI)

Definition in file cmpiOSBase_NextHopIPRoute.c.

5.11.2 Function Documentation

5.11.2.1 CMPIInstance* _makeInst_NextHopIPRoute (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIObjectPath * *cop*, const char ** *properties*, const struct nextHopIP * *pNHop*, CMPIStatus * *rc*)

Method to create a CMPIInstance of this class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] Context object.

cop [in] Source objectPath.

properties [in]

pNHop [in] nextHopIP instance.

rc [in] cim status.

Returns:

created NextHopIPRoute instance.

Definition at line 82 of file cmpiOSBase_NextHopIPRoute.c.

5.11.2.2 CMPIObjectPath* _makePath_NextHopIPRoute (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIObjectPath * *cop*, const struct nextHopIP * *pNHop*, CMPIStatus * *rc*)

Method to create a CMPIObjectPath of this class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] Context object.

cop [in] Source objectPath.

pNHop [in] nextHopIP instance.

rc [in] cim status.

Returns:

created NextHopIPRoute ObjectPath.

Definition at line 42 of file cmpiOSBase_NextHopIPRoute.c.

5.12 /mnt/storage/TESIS/sblim/cmpi-router/src/cmpiOSBase_-NextHopIPRouteProvider.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "cmpidt.h"
#include "cmpift.h"
#include "cmpimacs.h"
#include "OSBase_Common.h"
#include "cmpiOSBase_Common.h"
#include "cmpiOSBase_NextHopIPRoute.h"
```

Functions

- CMPIStatus getNextHopIPParams (const CMPIInstance *ci, struct nextHopIP *nHopIP, struct nextHopIP *nHopIPPrev)
- CMPIStatus OSBase_NextHopIPRouteProviderCleanup (CMPIInstanceMI *mi, const CMPIContext *ctx, CMPIBoolean terminating)
- CMPIStatus OSBase_NextHopIPRouteProviderEnumInstanceNames (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref)
- CMPIStatus OSBase_NextHopIPRouteProviderEnumInstances (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char **properties)
- CMPIStatus OSBase_NextHopIPRouteProviderGetInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char **properties)
- CMPIStatus OSBase_NextHopIPRouteProviderCreateInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const CMPIInstance *ci)
- CMPIStatus OSBase_NextHopIPRouteProviderSetInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const CMPIInstance *ci, const char **properties)
- CMPIStatus OSBase_NextHopIPRouteProviderDeleteInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop)
- CMPIStatus OSBase_NextHopIPRouteProviderExecQuery (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char *lang, const char *query)
- CMPIStatus OSBase_NextHopIPRouteProviderMethodCleanup (CMPIMethodMI *mi, const CMPIContext *ctx, CMPIBoolean terminating)
- CMPIStatus OSBase_NextHopIPRouteProviderInvokeMethod (CMPIMethodMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char *methodName, const CMPIArgs *in, CMPIArgs *out)
- CMInstanceMIStub (OSBase_NextHopIPRouteProvider, OSBase_NextHopIPRouteProvider, _broker, CMNoHook)
- CMMethodMIStub (OSBase_NextHopIPRouteProvider, OSBase_NextHopIPRouteProvider, _broker, CMNoHook)

Variables

- static const CMPIBroker * _broker

5.12.1 Detailed Description

cmpiOSBase_NextHopIPRouteProvider.c

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

Linux_NextHopIPRoute class provider implementation. The following CMPI instance methods are supported:

- Cleanup
- EnumerateInstanceNames
- EnumerateInstances
- GetInstance
- CreateInstance
- SetInstance
- DeleteInstance

Interface Type : Common Manageability Programming Interface (CMPI)

Definition in file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2 Function Documentation

5.12.2.1 CMInstanceMIStub (OSBase_NextHopIPRouteProvider, OSBase_NextHopIPRouteProvider, _broker, CMNoHook)

5.12.2.2 CMMethodMIStub (OSBase_NextHopIPRouteProvider, OSBase_NextHopIPRouteProvider, _broker, CMNoHook)

5.12.2.3 CMPIStatus getNextHopIPParams (const CMPIInstance * *ci*, struct nextHopIP * *nHopIP*, struct nextHopIP * *nHopIPPrev*)

Get parameters from CMPIInstance.

Parameters:

ci [in] CMPIInstance.

nHopIP [out] structure to be filled.

nHopIPPrev [in] previous route info.

Returns:

0=succesful | 1=fail

Definition at line 550 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.4 CMPIStatus OSBase_NextHopIPRouteProviderCleanup (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, CMPIBoolean *terminating*)

Definition at line 66 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.5 CMPIStatus OSBase_NextHopIPRouteProviderCreateInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const CMPIInstance * *ci*)

Definition at line 278 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.6 CMPIStatus OSBase_NextHopIPRouteProviderDeleteInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*)

Definition at line 403 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.7 CMPIStatus OSBase_NextHopIPRouteProviderEnumInstanceNames (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*)

Definition at line 79 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.8 CMPIStatus OSBase_NextHopIPRouteProviderEnumInstances (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char ** *properties*)

Definition at line 144 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.9 CMPIStatus OSBase_NextHopIPRouteProviderExecQuery (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char * *lang*, const char * *query*)

Definition at line 454 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.10 CMPIStatus OSBase_NextHopIPRouteProviderGetInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char ** *properties*)

Definition at line 212 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.11 CMPIStatus OSBase_NextHopIPRouteProviderInvokeMethod (CMPIMethodMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char * *methodName*, const CMPIArgs * *in*, CMPIArgs * *out*)

Definition at line 493 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.12 CMPIStatus OSBase_NextHopIPRouteProviderMethodCleanup (CMPIMethodMI * *mi*, const CMPIContext * *ctx*, CMPIBoolean *terminating*)

Definition at line 480 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.2.13 CMPIStatus OSBase_NextHopIPRouteProviderSetInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const CMPIInstance * *ci*, const char ** *properties*)

Definition at line 324 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.12.3 Variable Documentation

5.12.3.1 const CMPIBroker* _broker `[static]`

Definition at line 42 of file cmpiOSBase_NextHopIPRouteProvider.c.

5.13 /mnt/storage/TESIS/sblim/cmpi-router/src/cmpiOSBase_-RouteUsesEndpoint.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "cmpidt.h"
#include "cmpimacs.h"
#include "cmpiOSBase_Common.h"
#include "OSBase_Common.h"
#include "OSBase_Netlink.h"
#include "cmpiOSBase_RouteUsesEndpoint.h"
```

Functions

- CMPIObjectPath * _makePath_RouteUsesEndpoint (const CMPIBroker *_broker, const CMPIObjectPath *ops, const CMPIObjectPath *opt, CMPIStatus *rc)
- CMPIInstance * _makeInst_RouteUsesEndpoint (const CMPIBroker *_broker, const CMPIObjectPath *ops, const CMPIObjectPath *opt, CMPIStatus *rc)
- CMPIStatus * _assoc_get_NextHopRoute_insts (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const CMPIInstance *sourceInst, const char *targetClass, const int assocType, CMPIStatus *rc)
- CMPIStatus * _assoc_get_ProtocolEndpoint_insts (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const CMPIInstance *sourceInst, const char *targetClass, const int assocType, CMPIStatus *rc)
- CMPIStatus _assoc_RouteUsesEndpoint (const CMPIBroker *_broker, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char *targetClass, const int assocType, CMPIStatus *rc)

5.13.1 Detailed Description

cmpiOSBase_RouteUsesEndpoint.c

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This is the factory implementation for creating instances of CIM class Linux_RouteUsesEndpoint.

Interface Type : Common Magabeablity Programming Interface (CMPI)

Definition in file cmpiOSBase_RouteUsesEndpoint.c.

5.13.2 Function Documentation

5.13.2.1 CMPIStatus* _assoc_get_NextHopRoute_insts (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const CMPIInstance * *sourceInst*, const char * *targetClass*, int *assocType*, CMPIStatus * *rc*)

Retrieve a list of instances from target class (CIM_NextHopRoute subclass), associated to source class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] context object.

rslt [in] result.

ref [in] source objectPath.

sourceInst [in] source class instance.

targetClass [in] target class name.

assocType [in] association type.

rc [in] cim status.

Returns:

cim status.

Definition at line 120 of file cmpiOSBase_RouteUsesEndpoint.c.

5.13.2.2 CMPIStatus* _assoc_get_ProtocolEndpoint_insts (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const CMPIInstance * *sourceInst*, const char * *targetClass*, int *assocType*, CMPIStatus * *rc*)

Retrieve a list of instances from target class (CIM_ProtocolEndpoint subclass), associated to source class.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] context object.

rslt [in] result.

ref [in] source objectPath.

sourceInst [in] source class instance.

targetClass [in] target class name.

assocType [in] association type.

rc [in] cim status.

Returns:

cim status.

Definition at line 242 of file cmpiOSBase_RouteUsesEndpoint.c.

5.13.2.3 CMPIStatus _assoc_RouteUsesEndpoint (const CMPIBroker * *_broker*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char * *targetClass*, int *assocType*, CMPIStatus * *rc*)

Create CMPIInstances of association.

Parameters:

_broker [in] CIM Object Manager.

ctx [in] context object.

rslt [in] result.

ref [in] source objectPath.

targetClass [in] target class name.

assocType [in] association type.

rc [in] cim status.

Returns:

cim status.

Definition at line 368 of file cmpiOSBase_RouteUsesEndpoint.c.

5.13.2.4 CMPIInstance* _makeInst_RouteUsesEndpoint (const CMPIBroker * *_broker*, const CMPIObjectPath * *ops*, const CMPIObjectPath * *opt*, CMPIStatus * *rc*)

Method to create a CMPIInstance of this class.

Parameters:

_broker [in] CIM Object Manager.

ops [in] Source objectPath.

opt [in] Target objectPath.

rc [in] cim status.

Returns:

created Instance.

Definition at line 79 of file cmpiOSBase_RouteUsesEndpoint.c.

5.13.2.5 CMPIObjectPath* _makePath_RouteUsesEndpoint (const CMPIBroker * *_broker*, const CMPIObjectPath * *ops*, const CMPIObjectPath * *opt*, CMPIStatus * *rc*)

Method to create a CMPIObjectPath of this class.

Parameters:

_broker [in] CIM Object Manager.

ops [in] Source objectPath.

opt [in] Target objectPath.

rc [in] cim status.

Returns:

created ObjectPath.

Definition at line 41 of file cmpiOSBase_RouteUsesEndpoint.c.

5.14 /mnt/storage/TESIS/sblim/cmpi-router/src/cmpiOSBase_-RouteUsesEndpointProvider.c File Reference

```
#include <stdlib.h>
#include <stdio.h>
#include <string.h>
#include "cmpidt.h"
#include "cmpift.h"
#include "cmpimacs.h"
#include "OSBase_Common.h"
#include "cmpiOSBase_Common.h"
#include "cmpiOSBase_RouteUsesEndpoint.h"
```

Functions

- CMPIStatus OSBase_RouteUsesEndpointProviderCleanup (CMPIInstanceMI *mi, const CMPI-Context *ctx, CMPIBoolean terminating)
- CMPIStatus OSBase_RouteUsesEndpointProviderEnumInstanceNames (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref)
- CMPIStatus OSBase_RouteUsesEndpointProviderEnumInstances (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char **properties)
- CMPIStatus OSBase_RouteUsesEndpointProviderGetInstance (CMPIInstanceMI *mi, const CMPI-Context *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char **properties)
- CMPIStatus OSBase_RouteUsesEndpointProviderCreateInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const CMPIInstance *ci)
- CMPIStatus OSBase_RouteUsesEndpointProviderSetInstance (CMPIInstanceMI *mi, const CMPI-Context *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const CMPIInstance *ci, const char **properties)
- CMPIStatus OSBase_RouteUsesEndpointProviderDeleteInstance (CMPIInstanceMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop)
- CMPIStatus OSBase_RouteUsesEndpointProviderExecQuery (CMPIInstanceMI *mi, const CMPI-Context *ctx, const CMPIResult *rslt, const CMPIObjectPath *ref, const char *lang, const char *query)
- CMPIStatus OSBase_RouteUsesEndpointProviderAssociationCleanup (CMPIAssociationMI *mi, const CMPIContext *ctx, CMPIBoolean terminating)
- CMPIStatus OSBase_RouteUsesEndpointProviderAssociators (CMPIAssociationMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char *assocClass, const char *resultClass, const char *role, const char *resultRole, const char **propertyList)
- CMPIStatus OSBase_RouteUsesEndpointProviderAssociatorNames (CMPIAssociationMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char *assocClass, const char *resultClass, const char *role, const char *resultRole)
- CMPIStatus OSBase_RouteUsesEndpointProviderReferences (CMPIAssociationMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char *assocClass, const char *role, const char **propertyList)

- CMPIStatus OSBase_RouteUsesEndpointProviderReferenceNames (CMPIAssociationMI *mi, const CMPIContext *ctx, const CMPIResult *rslt, const CMPIObjectPath *cop, const char *assocClass, const char *role)
- CMInstanceMIStub (OSBase_RouteUsesEndpointProvider, OSBase_RouteUsesEndpointProvider, _broker, CMNoHook)
- CMAssociationMIStub (OSBase_RouteUsesEndpointProvider, OSBase_RouteUsesEndpointProvider, _broker, CMNoHook)

Variables

- static const CMPIBroker * _broker

5.14.1 Detailed Description

cmpiOSBase_RouteUsesEndpointProvider.c

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

Linux_RouteUsesEndpointProvider association provider implementation. The following CMPI instance methods are supported:

- Cleanup
- EnumerateInstanceNames
- EnumerateInstances
- GetInstance
- AssociationCleanup
- Associators
- AssociatorNames
- References
- ReferenceNames

Interface Type : Common Manageability Programming Interface (CMPI)

Definition in file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2 Function Documentation

5.14.2.1 CMAssociationMIStub (OSBase_RouteUsesEndpointProvider, OSBase_RouteUsesEndpointProvider, _broker, CMNoHook)

5.14.2.2 CMInstanceMIStub (OSBase_RouteUsesEndpointProvider, OSBase_RouteUsesEndpointProvider, _broker, CMNoHook)

5.14.2.3 CMPIStatus OSBase_RouteUsesEndpointProviderAssociationCleanup (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, CMPIBoolean *terminating*)

Definition at line 267 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.4 CMPIStatus OSBase_RouteUsesEndpointProviderAssociatorNames (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char * *assocClass*, const char * *resultClass*, const char * *role*, const char * *resultRole*)

Definition at line 371 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.5 CMPIStatus OSBase_RouteUsesEndpointProviderAssociators (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char * *assocClass*, const char * *resultClass*, const char * *role*, const char * *resultRole*, const char ** *propertyList*)

Definition at line 280 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.6 CMPIStatus OSBase_RouteUsesEndpointProviderCleanup (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, CMPIBoolean *terminating*)

Definition at line 59 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.7 CMPIStatus OSBase_RouteUsesEndpointProviderCreateInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const CMPIInstance * *ci*)

Definition at line 183 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.8 CMPIStatus OSBase_RouteUsesEndpointProviderDeleteInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*)

Definition at line 223 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.9 CMPIStatus OSBase_RouteUsesEndpointProviderEnumInstanceNames (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*)

Definition at line 72 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.10 CMPIStatus OSBase_RouteUsesEndpointProviderEnumInstances (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char ** *properties*)

Definition at line 109 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.11 CMPIStatus OSBase_RouteUsesEndpointProviderExecQuery (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *ref*, const char * *lang*, const char * *query*)

Definition at line 241 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.12 CMPIStatus OSBase_RouteUsesEndpointProviderGetInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char ** *properties*)

Definition at line 146 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.13 CMPIStatus OSBase_RouteUsesEndpointProviderReferenceNames (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char * *assocClass*, const char * *role*)

Definition at line 524 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.14 CMPIStatus OSBase_RouteUsesEndpointProviderReferences (CMPIAssociationMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const char * *assocClass*, const char * *role*, const char ** *propertyList*)

Definition at line 460 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.2.15 CMPIStatus OSBase_RouteUsesEndpointProviderSetInstance (CMPIInstanceMI * *mi*, const CMPIContext * *ctx*, const CMPIResult * *rslt*, const CMPIObjectPath * *cop*, const CMPIInstance * *ci*, const char ** *properties*)

Definition at line 202 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.14.3 Variable Documentation

5.14.3.1 const CMPIBroker* _broker [static]

Definition at line 45 of file cmpiOSBase_RouteUsesEndpointProvider.c.

5.15 /mnt/storage/TESIS/sblim/cmpi-router/src/OSBase_-LANEndpoint.c File Reference

```
#include "OSBase_Common.h"
#include "OSBase_LANEndpoint.h"
#include <unistd.h>
#include <linux/rtnetlink.h>
#include <linux/if.h>
#include <sys/types.h>
#include <signal.h>
```

Functions

- unsigned long datetime_str_interval_to_ms (const char *pDatetimeString)
- int getLANEndpoints (struct LANEndpointList **lanEPList, const struct nlLinkInfo *nlLinkInfo)
- int getALLLANEndpoints (struct LANEndpointList **list)
- int getLANEndpoint (struct LANEndpointList **lanEPList, const char *linkName)
- unsigned short changeLinkOPState (const char *linkName, unsigned short enabledState, unsigned long timeoutPeriod)
- int nlListTOLanEPList (struct nlLinkInfoList **nlLinkInfoList, struct LANEndpointList **lanEPList)
- int nlInfoTOLanEP (struct nlLinkInfo *nlLinkInfo, struct LANEndpoint *lanEP)
- void freeLANEndpointList (struct LANEndpointList *lptr)
- void freeLANEndpoint (struct LANEndpoint *sptr)

5.15.1 Detailed Description

OSBase_LANEndpoint.c

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This shared library provides resource access functionality for the class Linux_LANEndpoint. It is independent from any specific CIM technology.

Definition in file OSBase_LANEndpoint.c.

5.15.2 Function Documentation

5.15.2.1 unsigned short changeLinkOPState (const char * *linkName*, unsigned short *enabledState*, unsigned long *timeoutPeriod*)

Used to set device status.

Parameters:

linkName [in] link name.

enabledState [in] desired state for the device.

timeoutPeriod [in] maximum amount of time that the client expects the transition to the new state to take.

Returns:

0 = Completed with No Error 1 = Not Supported 2 = Unknown or Unspecified Error 3 = Cannot complete within Timeout Period 4 = Failed 5 = Invalid Parameter 6 = In Use 7..4095 = DMTF Reserved 4096 = Method Parameters Checked - Job Started 4097 = Invalid State Transition 4098 = Use of Timeout Parameter Not Supported 4099 = Busy 4100..32767 = Method Reserved 32768..65535 = Vendor Specific

Definition at line 157 of file OSBase_LANEndpoint.c.

5.15.2.2 unsigned long datetime_str_interval_to_ms (const char * *pDatetimeString*)

This function converts from datetime string interval to total number of milliseconds (10e-3).

Parameters:

pDatetimeString [in] datetime string interval to be converted.

Returns:

converted datetime string interval.

Todo

move this function to the proper file.

Definition at line 38 of file OSBase_LANEndpoint.c.

5.15.2.3 void freeLANEndpoint (struct LANEndpoint * *sptr*)

This function is used to clean a LANEndpoint structure.

Parameters:

sptr [in] structure to be cleaned.

Definition at line 501 of file OSBase_LANEndpoint.c.

5.15.2.4 void freeLANEndpointList (struct LANEndpointList * *lptr*)

This function is used to clean a LANEndpoint list.

Parameters:

lptr [in] list to be cleaned.

Definition at line 478 of file OSBase_LANEndpoint.c.

5.15.2.5 int getALLLANEndpoints (struct LANEndpointList ** *list*)

Definition at line 104 of file OSBase_LANEndpoint.c.

5.15.2.6 int getLANEndpoint (struct LANEndpointList ** *lanEPList*, const char * *linkName*)

This functions is used to get a route base on specified InstanceID.

Parameters:

lanEPList [out] LANEndpoint struct to be filled.

linkName [in] link ifname.

Returns:

0=succesful | 1=fail

Definition at line 130 of file OSBase_LANEndpoint.c.

5.15.2.7 int getLANEndpoints (struct LANEndpointList ** *lanEPList*, const struct nlLinkInfo * *nlLinkInfo*)

This functions is used to get a list of LANEndpoints based on nlLinkInfo filter.

Parameters:

lanEPList [out] LANEndpoint struct to be filled.

nlLinkInfo [in] link filter info.

Returns:

0=succesful | 1=fail

Definition at line 58 of file OSBase_LANEndpoint.c.

5.15.2.8 int nlInfoTOLanEP (struct nlLinkInfo * *nlLinkInfo*, struct LANEndpoint * *lanEP*)

Converts a nlLinkInfo structure to LANEndpoint structure.

Parameters:

nlLinkInfo [in] structure to be converted.

lanEP [out] converted structure.

Returns:

0=succesful | 1=fail

Definition at line 287 of file OSBase_LANEndpoint.c.

5.15.2.9 int nlListTOLanEPList (struct nlLinkInfoList ** *nlLinkInfoList*, struct LANEndpointList ** *lanEPList*)

Converts a nlLinkInfoList structure to LANEndpointList strucutre.

Parameters:

nlLinkInfoList [in] list to be converted.

lanEPList [out] converted list.

Returns:

0=succesful | 1=fail

Definition at line 244 of file OSBase_LANEndpoint.c.

5.16 /mnt/storage/TESIS/sblim/cmpi-router/src/OSBase_Netlink.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <sys/socket.h>
#include <linux/netlink.h>
#include <linux/rtnetlink.h>
#include <linux/if_arp.h>
#include <netinet/in.h>
#include <errno.h>
#include <time.h>
#include "OSBase_Common.h"
#include "OSBase_Netlink.h"
```

Functions

- int nlOpenSocket ()
- int nlCloseSocket ()
- int nlAddAttrToMsg (struct nlmsghdr *h, int maxlen, int attrType, const void *attrData, int attrBytelen)
- int nlAddAttrToMsg32 (struct nlmsghdr *h, int maxlen, int attrType, unsigned int attrData)
- int nlGetRoutes (struct nlRouteInfoList **nlRtInfoList, struct nlRouteInfo *nlRtInfo)
- int nlModifyRoute (struct nlRouteInfo *nlRtInfo, int hType, unsigned int hFlags)
- int nlAddRouteToList (const struct nlmsghdr *rcvH, struct nlRouteInfoList **nlRtInfoList, struct nlRouteInfo *nlRtInfo)
- int nlGenRouteFilter (struct nlRouteInfo *nlRtInfo)
- void nlResetRouteFilter ()
- int nlCreateDefaultRtInfo (struct nlRouteInfo *nlRtInfo)
- int nlGetLinks (struct nlLinkInfoList **nlLinkInfoList, const struct nlLinkInfo *nlLinkInfo)
- int nlModifyLink (struct nlLinkInfo *nlLinkInfo, int hType, unsigned int hFlags)
- int nlAddLinkToList (const struct nlmsghdr *rcvH, struct nlLinkInfoList **nlLinkInfoList, const struct nlLinkInfo *nlLinkInfo)
- int nlGenLinkFilter (const struct nlLinkInfo *nlLinkInfo)
- void nlResetLinkFilter ()
- int nlCreateDefaultLinkInfo (struct nlLinkInfo *nlLinkInfo)
- void nlAddr_n2a (const unsigned char *addr, int alen, int type, char *buf, int blen)
- unsigned short nlGetLinkTypePos (unsigned short devType)

Variables

- struct nlSockHandle nlSH = { .fd = -1 }

5.16.1 Detailed Description

OSBase_NetLink.c

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

Netlink support lib. Based on iproute2 package: (http://www.linuxfoundation.org/en/Net:Iproute2)

Definition in file OSBase_Netlink.c.

5.16.2 Function Documentation

5.16.2.1 int nlAddAttrToMsg (struct nlmsghdr * *h*, int *maxlen*, int *attrType*, const void * *attrData*, int *attrBytelen*)

This function is used to add an attribute to a netlink message.

Parameters:

h [out] message.

maxlen [in] message length.

attrType [in] attribute type.

attrData [in] attribute data.

attrBytelen [in] attribute length.

Returns:

0=succesful | 1=fail

Definition at line 147 of file OSBase_Netlink.c.

5.16.2.2 int nlAddAttrToMsg32 (struct nlmsghdr * *h*, int *maxlen*, int *attrType*, unsigned int *attrData*)

This function is used to add an attribute to a netlink message.

Parameters:

h [out] message.

maxlen [in] message length.

attrType [in] attribute type.

attrData [in] attribute data.

Returns:

0=succesful | 1=fail

Definition at line 175 of file OSBase_Netlink.c.

5.16.2.3 int nlAddLinkToList (const struct nlmsghdr * *rcvH*, struct nlLinkInfoList ** *nlLinkInfoList*, const struct nlLinkInfo * *nlLinkInfo*)

This function is used to add a link to specified links list.

Parameters:

rcvH [in] link message.

nlLinkInfoList [out] list where link will be added.

nlLinkInfo [in] link info.

Returns:

0=succesful | 1=fail

Definition at line 1490 of file OSBase_Netlink.c.

5.16.2.4 void nlAddr_n2a (const unsigned char * *addr*, int *alen*, int *type*, char * *buf*, int *blen*)

Function to convert a L2 address from network representation to string representation (user friendly).

Parameters:

addr [in] L2 address to convert.

alen [in] L2 address length.

type [in] link type.

buf [out] converted address.

blen [in] converted address maximum length.

Definition at line 1892 of file OSBase_Netlink.c.

5.16.2.5 int nlAddRouteToList (const struct nlmsghdr * *rcvH*, struct nlRouteInfoList ** *nlRtInfoList*, struct nlRouteInfo * *nlRtInfo*)

This function is used to add a route to specified routes list.

Parameters:

rcvH [in] route message.

nlRtInfoList [out] list where route will be added.

nlRtInfo [in] route info.

Returns:

0=succesful | 1=fail

Definition at line 659 of file OSBase_Netlink.c.

5.16.2.6 int nlCloseSocket ()

Used to close previously open Netlink socket.

Returns:

0=succesful | 1=fail

Definition at line 124 of file OSBase_Netlink.c.

5.16.2.7 int nlCreateDefaultLinkInfo (struct nlLinkInfo * *nlLinkInfo*)

Used to create a nlLinkInfo structure with default values.

Parameters:

nlLinkInfo [out] link structure.

Returns:

0=succesful | 1=fail

Definition at line 1855 of file OSBase_Netlink.c.

5.16.2.8 int nlCreateDefaultRtInfo (struct nlRouteInfo * *nlRtInfo*)

Used to create a nlRouteInfo structure with default values.

Parameters:

nlRtInfo [out] route structure.

Returns:

0=succesful | 1=fail

Definition at line 1069 of file OSBase_Netlink.c.

5.16.2.9 int nlGenLinkFilter (const struct nlLinkInfo * *nlLinkInfo*)

Used to generate a filter to be applied to a list of links.

Parameters:

nlLinkInfo [in] link info.

Returns:

0=succesful | 1=fail

Definition at line 1700 of file OSBase_Netlink.c.

5.16.2.10 int nlGenRouteFilter (struct nlRouteInfo * *nlRtInfo*)

Used to generate a filter to be applied to a list of routes.

Parameters:

nlRtInfo [in] route info.

Returns:

0=succesful | 1=fail

Definition at line 851 of file OSBase_Netlink.c.

5.16.2.11 int nlGetLinks (struct nlLinkInfoList ** *nlLinkInfoList*, const struct nlLinkInfo * *nlLinkInfo*)

This function is used to get links using nlLinkInfo as filter.

Parameters:

nlLinkInfoList [out] links list (one or more entries).

nlLinkInfo [in] link info.

Returns:

0=succesful | 1=fail

Definition at line 1110 of file OSBase_Netlink.c.

5.16.2.12 unsigned short nlGetLinkTypePos (unsigned short *devType*)

Lookup the position of the passed type into linkType array.

Parameters:

devType [in] type to find.

Returns:

position into linkType array.

Definition at line 1929 of file OSBase_Netlink.c.

5.16.2.13 int nlGetRoutes (struct nlRouteInfoList ** *nlRtInfoList*, struct nlRouteInfo * *nlRtInfo*)

This function is used to get routes using nlRtInfo as filter.

Parameters:

nlRtInfoList [out] routes list (one or more entries).
nlRtInfo [in] route info.

Returns:

0=succesful | 1=fail

Definition at line 209 of file OSBase_Netlink.c.

5.16.2.14 int nlModifyLink (struct nlLinkInfo * *nlLinkInfo*, int *hType*, unsigned int *hFlags*)

This function is used to ADD/DELETE/MODIFY a link.

Parameters:

nlLinkInfo [in] link info.
hType [in] type of message.
hFlags [in] message flags.

Returns:

0=succesful | 1=fail

Definition at line 1279 of file OSBase_Netlink.c.

5.16.2.15 int nlModifyRoute (struct nlRouteInfo * *nlRtInfo*, int *hType*, unsigned int *hFlags*)

This function is used to ADD/DELETE/MODIFY a route.

Parameters:

nlRtInfo [in] route info.
hType [in] type of message.
hFlags [in] message flags.

Returns:

0=succesful | 1=fail

Definition at line 384 of file OSBase_Netlink.c.

5.16.2.16 int nlOpenSocket ()

Used to open a Netlink socket.

Returns:

0=succesful | 1=fail

Definition at line 48 of file OSBase_Netlink.c.

5.16.2.17 void nlResetLinkFilter ()

Used to reset link filter handler.

Definition at line 1825 of file OSBase_Netlink.c.

5.16.2.18 void nlResetRouteFilter ()

Used to reset route filter handler.

Definition at line 1042 of file OSBase_Netlink.c.

5.16.3 Variable Documentation

5.16.3.1 struct nlSockHandle nlSH = { .fd = -1 }

Netlink socket handler instance.

Definition at line 44 of file OSBase_Netlink.c.

5.17 /mnt/storage/TESIS/sblim/cmpi-router/src/OSBase_-NextHopIPRoute.c File Reference

```
#include "OSBase_Common.h"
#include "OSBase_Netlink.h"
#include "OSBase_NextHopIPRoute.h"
#include <unistd.h>
#include <string.h>
#include <sys/param.h>
#include <linux/rtnetlink.h>
```

Functions

- int getIPRoutes (struct nextHopIPList **nHopIPList, struct nlRouteInfo *nlRtInfo)
- int getAllIPRoutes (struct nextHopIPList **nHopIPList)
- int getIPRouteIId (struct nextHopIPList **nHopIPList, char *instanceId)
- int addIPRoute (struct nextHopIP *nHopIP)
- int delIPRoute (struct nextHopIP *nHopIP)
- int nlListTOnhList (struct nlRouteInfoList **nlRtInfoList, struct nextHopIPList **nHopIPList)
- int nhTOnlInfo (struct nextHopIP *nHopIP, struct nlRouteInfo *nlRtInfo)
- int nlInfoTOnh (struct nlRouteInfo *nlRtInfo, struct nextHopIP *nHopIP)
- int getAddrType (const char *addr)
- int getAddrTypeStr (int addressType, char *addrTypeStr)
- void freeNextHopIPList (struct nextHopIPList *lptr)
- void freeNextHopIP (struct nextHopIP *sptr)

5.17.1 Detailed Description

OSBase_NextHopIPRoute.c

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

This shared library provides resource access functionality for the class Linux_NextHopIPRoute. It is independent from any specific CIM technology.

Definition in file OSBase_NextHopIPRoute.c.

5.17.2 Function Documentation

5.17.2.1 int addIPRoute (struct nextHopIP * *nHopIP*)

This function is used to add a new route to FIB.

Parameters:

nHopIP [in] route to be added.

Returns:

0=succesful | 1=fail

Definition at line 143 of file OSBase_NextHopIPRoute.c.

5.17.2.2 int delIPRoute (struct nextHopIP * *nHopIP*)

This function is used to delete a route from FIB.

Parameters:

nHopIP [in] route to be deleted.

Returns:

0=succesful | 1=fail

Definition at line 186 of file OSBase_NextHopIPRoute.c.

5.17.2.3 void freeNextHopIP (struct nextHopIP * *sptr*)

This function is used to clean a nextHopIP structure.

Parameters:

sptr [in] structure to be cleaned.

Definition at line 583 of file OSBase_NextHopIPRoute.c.

5.17.2.4 void freeNextHopIPList (struct nextHopIPList * *lptr*)

This function is used to clean a nextHopIP list.

Parameters:

lptr [in] list to be cleaned.

Definition at line 560 of file OSBase_NextHopIPRoute.c.

5.17.2.5 int getAddrType (const char * *addr*)

Used to get CIM address type from specified route address.

Parameters:

addr [int] route address.

Returns:

CIM address type.

Definition at line 521 of file OSBase_NextHopIPRoute.c.

5.17.2.6 int getAddrTypeStr (int *addressType*, char * *addrTypeStr*)

Used to get CIM address type in string format.

Parameters:

addressType [in] address type identifier.
addrTypeStr [out] address type string.

Returns:

0=succesful | 1=fail

Definition at line 540 of file OSBase_NextHopIPRoute.c.

5.17.2.7 int getAllIPRoutes (struct nextHopIPList ** *nHopIPList*)

This functions is used to get all ipv4 and ipv6 routes from fib.

Parameters:

nHopIPList [out] routes struct to be filled.

Returns:

0=succesful | 1=fail

Definition at line 77 of file OSBase_NextHopIPRoute.c.

5.17.2.8 int getIPRouteIId (struct nextHopIPList ** *nHopIPList*, char * *instanceId*)

This functions is used to get a route base on specified InstanceID.

Parameters:

nHopIPList [out] routes struct to be filled.
instanceId [in] route InstanceID.

Returns:

0=succesful | 1=fail

Definition at line 110 of file OSBase_NextHopIPRoute.c.

5.17.2.9 int getIPRoutes (struct nextHopIPList ** *nHopIPList*, struct nlRouteInfo * *nlRtInfo*)

This functions is used to get a list of routes based on nlRtInfo filter.

Parameters:

nHopIPList [out] routes struct to be filled.

nlRtInfo [in] route info.

Returns:

0=succesful | 1=fail

Definition at line 31 of file OSBase_NextHopIPRoute.c.

5.17.2.10 int nhTOnlInfo (struct nextHopIP * *nHopIP*, struct nlRouteInfo * *nlRtInfo*)

Converts a nextHopIP structure to nlRouteInfo structure.

Parameters:

nHopIP [in] structure to be converted.

nlRtInfo [out] converted structure.

Returns:

0=succesful | 1=fail

Definition at line 273 of file OSBase_NextHopIPRoute.c.

5.17.2.11 int nlInfoTOnh (struct nlRouteInfo * *nlRtInfo*, struct nextHopIP * *nHopIP*)

Converts a nlRouteInfo structure to nextHopIP structure.

Parameters:

nlRtInfo [in] structure to be converted.

nHopIP [out] converted structure.

Returns:

0=succesful | 1=fail

Definition at line 381 of file OSBase_NextHopIPRoute.c.

5.17.2.12 int nlListTOnhList (struct nlRouteInfoList ** *nlRtInfoList*, struct nextHopIPList ** *nHopIPList*)

Converts a nlRouteInfoList structure to nextHopIPList strucutre.

Parameters:

nlRtInfoList [in] list to be converted.

nHopIPList [out] converted list.

Returns:

0=succesful | 1=fail

Definition at line 233 of file OSBase_NextHopIPRoute.c.

5.18 /mnt/storage/TESIS/sblim/cmpi-router/src/OSBase_Zebra.c File Reference

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include "OSBase_Common.h"
#include "OSBase_Netlink.h"
```

5.18.1 Detailed Description

OSBase_NetLink.c

Author:

Federico Martin Casares (warptrosse@gmail.com)

Contributors:

Description:

Todo

WILL BE USED IN THE FUTURE TO OBTAIN INFO FROM ZEBRA/QUAGGA.

Definition in file OSBase_Zebra.c.

Index