



1
2
3
4
5

Document Number: DSP1120

Date: 2016-02-23

Version: 1.0.0b

6 **Network Management - Tunnel Management** 7 **Profile**

Information for Work-in-Progress version:

IMPORTANT: This document is not a standard. It does not necessarily reflect the views of the DMTF or all of its members. Because this document is a Work in Progress, it may still change, perhaps profoundly. This document is available for public review and comment until superseded.

Provide any comments through the DMTF Feedback Portal:

<http://www.dmtf.org/standards/feedback>

8 **Supersedes: 1.0.0a**
9 **Document Class: Normative**
10 **Document Status: Work In Progress**
11 **Document Language: en-US**

12 Copyright Notice

13 Copyright © 2013 - 2016 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

14 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
15 management and interoperability. Members and non-members may reproduce DMTF specifications and
16 documents, provided that correct attribution is given. As DMTF specifications may be revised from time to
17 time, the particular version and release date should always be noted.

18 Implementation of certain elements of this standard or proposed standard may be subject to third party
19 patent rights, including provisional patent rights (herein "patent rights"). DMTF makes no representations
20 to users of the standard as to the existence of such rights, and is not responsible to recognize, disclose,
21 or identify any or all such third party patent right, owners or claimants, nor for any incomplete or
22 inaccurate identification or disclosure of such rights, owners or claimants. DMTF shall have no liability to
23 any party, in any manner or circumstance, under any legal theory whatsoever, for failure to recognize,
24 disclose, or identify any such third party patent rights, or for such party's reliance on the standard or
25 incorporation thereof in its product, protocols or testing procedures. DMTF shall have no liability to any
26 party implementing such standard, whether such implementation is foreseeable or not, nor to any patent
27 owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is
28 withdrawn or modified after publication, and shall be indemnified and held harmless by any party
29 implementing the standard from any and all claims of infringement by a patent owner for such
30 implementations.

31 For information about patents held by third-parties which have notified the DMTF that, in their opinion,
32 such patent may relate to or impact implementations of DMTF standards, visit
33 <http://www.dmtf.org/about/policies/disclosures.php>.

34 This document's normative language is English. Translation into other languages is permitted.

35

CONTENTS

36	Foreword	5
37	Introduction.....	6
38	1 Scope	7
39	2 Normative references	7
40	3 Terms and definitions	8
41	4 Symbols and abbreviated terms.....	9
42	5 Synopsis	10
43	6 Description	10
44	6.1 Class diagram	10
45	6.2 Tunneling Protocol Variants.....	11
46	6.3 IIPEncapsulationInterface	11
47	6.4 CIM_EncapsulationMappingSettingData	11
48	6.5 CIM_IPEncapsulationManagementService	12
49	7 Implementation	12
50	7.1 Representing IP encapsulation management services	12
51	7.1.1 CIM_IPEncapsulationManagementService	12
52	7.2 CIM_IPEncapsulationInterface	12
53	7.3 Representing the Protocol Endpoints of an Encapsulation gateway.....	12
54	7.4 Representing an encapsulation mapping table	13
55	7.4.1 CIM_EncapsulationMappingSettingData	13
56	8 Methods.....	13
57	8.1 Extrinsic Methods.....	13
58	8.1.1 Job parameter.....	14
59	8.1.2 CIM_IPConfigurationService.AddIIPEncapsulationInterface()	14
60	8.1.3 CIM_IPConfigurationService.AddIIPEncapsulationMappings().....	15
61	8.1.4 CIM_IPConfigurationService.RemoveIIPEncapsulationInterface()	16
62	8.1.5 CIM_IPConfigurationService.RemoveIIPEncapsulationMappings().....	16
63	8.2 Profile conventions for operations	17
64	8.3 CIM_GatewayEndpoint.....	17
65	8.4 CIM_BindsTo	18
66	8.5 CIM_HostedService	18
67	8.6 CIM_HostedIPInterface.....	18
68	8.7 CIM_IPEncapsulationManagementService	19
69	8.8 IIPEncapsulationInterface	19
70	9 Use cases.....	20
71	9.1 Profile Registration.....	20
72	9.2 L2 NVGRE Tunnel Gateway	21
73	9.3 Routed NVGRE Tunnel Gateway	22
74	10 CIM Elements	24
75	10.1 CIM_BindsToLANEndpoint.....	25
76	10.2 CIM_BindsTo	25
77	10.3 CIM_EncapsulationMappingSettingData	25
78	10.4 CIM_HostedService	26
79	10.5 CIM_IPEncapsulationManagementService	26
80	10.6 CIM_IPProtocolEndpoint	26
81	10.7 CIM_IPEncapsulationInterface	27
82	10.8 CIM_RegisteredProfile.....	28
83	ANNEX A (informative) Change log.....	29

84

85 **Figures**

86		
87	Figure 1 – Network Management - Tunnel Management Profile	11
88	Figure 2 – Registered profile	20
89	Figure 3 – NVGRE Tunnel Gateway	22
90	Figure 4 – NVGRE Routed Tunnel Gateway	24
91		

92 **Tables**

93		
94	Table 1 – Referenced profiles	10
95	Table 2 – AddIPEncapsulationInterface () Method: Parameters	14
96	Table 3 – AddIPEncapsulationMappings () Method: Parameters	15
97	Table 4 – RemoveIPEncapsulationInterface() Method: Parameters	16
98	Table 5 – RemoveIPEncapsulationMappings() Method: Parameters	17
99	Table 6 – Operations: CIM_GatewayEndpoint	17
100	Table 7 – Operations: CIM_BindsToLANEndpoint	18
101	Table 8 – Operations: CIM_HostedService	18
102	Table 9 – Operations: CIM_HostedIPInterface	18
103	Table 10 – CIM Elements: Network Management – Tunnel Management Profile	24
104	Table 11 – Class: CIM_BindsToLANEndpoint	25
105	Table 12 – Class: CIM_BindsTo	25
106	Table 13 – Class: CIM_EncapsulationMappingSettingData	25
107	Table 14 – Class: CIM_HostedService	26
108	Table 15 – Class: CIM_IPEncapsulationManagmentService	26
109	Table 16 – Class: CIM_IPProtocolEndpoint	26
110	Table 17 – Class: CIM_IPEncapsulationInterface	27
111	Table 18 – Class: CIM_RegisteredProfile	28
112		

113

Foreword

114 The *Network Management* - Tunnel Management Profile (DSP 1120) was prepared by the Network
115 Services Management Working Group of the DMTF.

116 DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems
117 management and interoperability.

118 Acknowledgments

119 The DMTF acknowledges the following individuals for their contributions to this document:

120 Editors:

- 121 • John Parchem – DMTF Fellow
- 122 • Bhumip Khasnabish - ZTE Corporation

123 Contributors:

- 124 • John Crandall – Brocade Communications System
- 125 • Bhumip Khasnabish - ZTE Corporation
- 126 • Lawrence Lamers – VMware
- 127 • John Parchem – DMTF Fellow
- 128 • Shishir Pardikar – Citrix
- 129 • Hemal Shah – Broadcom Corporation
- 130 • Eric Wells – Hitachi
- 131 • Alex Zhdankin – Cisco Systems

132

133

Introduction

134 The information in this specification should be sufficient for a provider or consumer of this data to identify
135 unambiguously the classes, properties, methods, and values that shall be instantiated and manipulated to
136 represent and manage Network Services and the associated configuration information. The target
137 audience for this specification is implementers who are writing CIM-based providers or consumers of
138 management interfaces that represent the component described in this document.

139 Document conventions

140 Typographical conventions

141 The following typographical conventions are used in this document:

- 142 • Document titles are marked in *italics*.
- 143 • ABNF rules are in `monospaced font`.

144

145 Network Management - Tunnel Management Profile

146 1 Scope

147 The *Network Management - Tunnel Management Profile* is a profile that will specify the CIM schema and
148 use cases associated with the general and common aspects of tunneling management. In general, the
149 tunnel interface includes switch virtual interface and loopback interface.

150 2 Normative references

151 The following referenced documents are indispensable for the application of this document. For dated or
152 versioned references, only the edition cited (including any corrigenda or DMTF update versions) applies.
153 For references without a date or version, the latest published edition of the referenced document
154 (including any corrigenda or DMTF update versions) applies.

155 DMTF DSP0004, *CIM Infrastructure Specification 2.6*,
156 http://www.dmtf.org/standards/published_documents/DSP0004_2.6.pdf

157 DMTF DSP0200, *CIM Operations over HTTP 1.3*,
158 http://www.dmtf.org/standards/published_documents/DSP0200_1.3.pdf

159 DMTF DSP0223, *Generic Operations 1.0*,
160 http://www.dmtf.org/standards/published_documents/DSP0223_1.0.pdf

161 DMTF DSP1001, *Management Profile Specification Usage Guide 1.0*,
162 http://www.dmtf.org/standards/published_documents/DSP1001_1.0.pdf

163 DMTF DSP1033, *Profile Registration Profile 1.0*,
164 http://www.dmtf.org/standards/published_documents/DSP1033_1.0.pdf

165 DMTF DSP1097, *Virtual Ethernet Switch Profile 1.1*,
166 http://dmtf.org/sites/default/files/standards/documents/DSP1097_1.1.0.pdf

167 DMTF DSP1036 *IP Interface Profile 1.1.1*,
168 http://www.dmtf.org/sites/default/files/standards/documents/DSP1036_1.1.1.pdf

169 IETF WG, Network Virtualization Overlays (NVO3), Sept. 2011,
170 <https://datatracker.ietf.org/wg/nvo3/charter/>

171 IETF Draft, A Stateless Transport Tunneling (STT) Protocol, April 2014,
172 <http://tools.ietf.org/html/draft-davie-stt-06/>

173 IETF Draft, Network Virtualization using Generic Routing Encapsulation (NVGRE), July 2014,
174 <http://tools.ietf.org/html/draft-sridharan-virtualization-nvgre-05/>

175 IETF RFC 7348, Virtual eXtensible Local Area Network (VXLAN), August 2014,
176 <http://tools.ietf.org/html/rfc7348/>

177 IETF Draft, Generic Network Virtualization Encapsulation (Geneve), August 2014,
178 <http://tools.ietf.org/html/draft-gross-geneve-01/>

179 ISO/IEC Directives, Part 2, *Rules for the structure and drafting of International Standards*,
180 <http://isotc.iso.org/livelink/livelink.exe?func=ll&objId=4230456&objAction=browse&sort=subtype>

181 3 Terms and definitions

182 In this document, some terms have a specific meaning beyond the normal English meaning. Those terms
183 are defined in this clause.

184 The terms "shall" ("required"), "shall not", "should" ("recommended"), "should not" ("not recommended"),
185 "may," "need not" ("not required"), "can" and "cannot" in this document are to be interpreted as described
186 in [ISO/IEC Directives, Part 2](#), Annex H. The terms in parenthesis are alternatives for the preceding term,
187 for use in exceptional cases when the preceding term cannot be used for linguistic reasons. Note that
188 [ISO/IEC Directives, Part 2](#), Annex H specifies additional alternatives. Occurrences of such additional
189 alternatives shall be interpreted in their normal English meaning.

190 The terms "clause", "subclause", "paragraph", and "annex" in this document are to be interpreted as
191 described in [ISO/IEC Directives, Part 2](#), Clause 5.

192 The terms "normative" and "informative" in this document are to be interpreted as described in [ISO/IEC](#)
193 [Directives, Part 2](#), Clause 3. In this document, clauses, subclauses, or annexes labeled "(informative)" do
194 not contain normative content. Notes and examples are always informative elements.

195 The terms defined in [DSP0004](#), [DSP0223](#), and [DSP1001](#) apply to this document. The following additional
196 terms are used in this document.

197 3.1

198 **conditional**

199 indicates requirements to be followed strictly to conform to the document when the specified conditions
200 are met

201 3.2

202 **mandatory**

203 indicates requirements to be followed strictly to conform to the document and from which no deviation is
204 permitted

205 3.3

206 **optional**

207 indicates a course of action permissible within the limits of the document

208 3.4

209 **pending configuration**

210 indicates the configuration that will be applied to an IP network connection the next time the IP network
211 connection accepts a configuration

212 3.5

213 **referencing profile**

214 indicates a profile that owns the definition of this class and can include a reference to this profile in its
215 "Referenced Profiles" table

216 3.6

217 **unspecified**

218 indicates that this profile does not define any constraints for the referenced CIM element or operation
219

220 4 Symbols and abbreviated terms

221 The abbreviations defined in [DSP0004](#), [DSP0223](#), and [DSP1001](#) apply to this document. The following
222 additional abbreviations are used in this document.

223 4.1

224 Customer Network

225 The customer network in an overlay is the network as seen by a virtual machines or a physical server. In
226 an overlay network the customer network traffic is the encapsulated payload in a packet on the provider
227 network.

228 4.2

229 IP

230 Internet Protocol

231 4.3

232 Gateway

233 Interconnects networks with different network protocol technologies or separate IP address networks
234 by performing the required protocol or IP address mapping conversions.

235 4.4

236 IP Gateway

237 Uses the IP protocol as an underlay protocol to tunnel non-routable network segments, allowing the
238 traffic to be routed on an IP network.

239 4.5

240 Tunnel

241 A Tunnel is a path across non-routable network segments. These can be segments on different IP
242 networks or networks using different protocols.

243 4.6

244 IP Encapsulation

245 A method of creating IP packets in which logically separate networks are abstracted from their underlying
246 structures by inclusion of inside network packet within higher level network packet.

247 4.7

248 Overlay Network\Customer

249 A network containing non-encapsulated network traffic from the perspective of the encapsulation
250 gateways specified in this profile.

251 4.8

252 Provider Network

253 The underlay network of a tunnel. The network containing the encapsulated network traffic from the
254 perspective of the encapsulation gateways specified in this profile.

255 4.9

256 Network Overlay/Underlay

257 Network overlay/underlay allows encapsulation of one packet into another using "packet-in-a-packet"
258 technique. The encapsulated packet is forwarded to an endpoint where it is decapsulated. Network
259 overlay/underlay is commonly used to (a) support secure multi-tenancy and (b) extend one network
260 across another.

261

262 5 Synopsis

263 **Profile name:** Network Management - Tunnel Management Profile

264 **Version:** 1.0.0b

265 **Organization:** DMTF

266 **CIM Schema version:** 2.45

267 **Central class:** CIM_IPEncapsulationManagementSevice

268 **Scoping class:** CIM_ComputerSystem

269 The *Network Management - Tunnel Management Profile* is a profile that specifies the CIM schema and
 270 use cases associated with Tunneling Management where a layer 2 or layer 3 overlay network is carried
 271 over a tunnel where layer 3 is used as the tunnel underlay. This profile includes a specification of the
 272 IIPEncapsulationInterface and the associated setting data among others.

273 Table 1 identifies profiles on which this profile has a dependency.

274 **Table 1 – Referenced profiles**

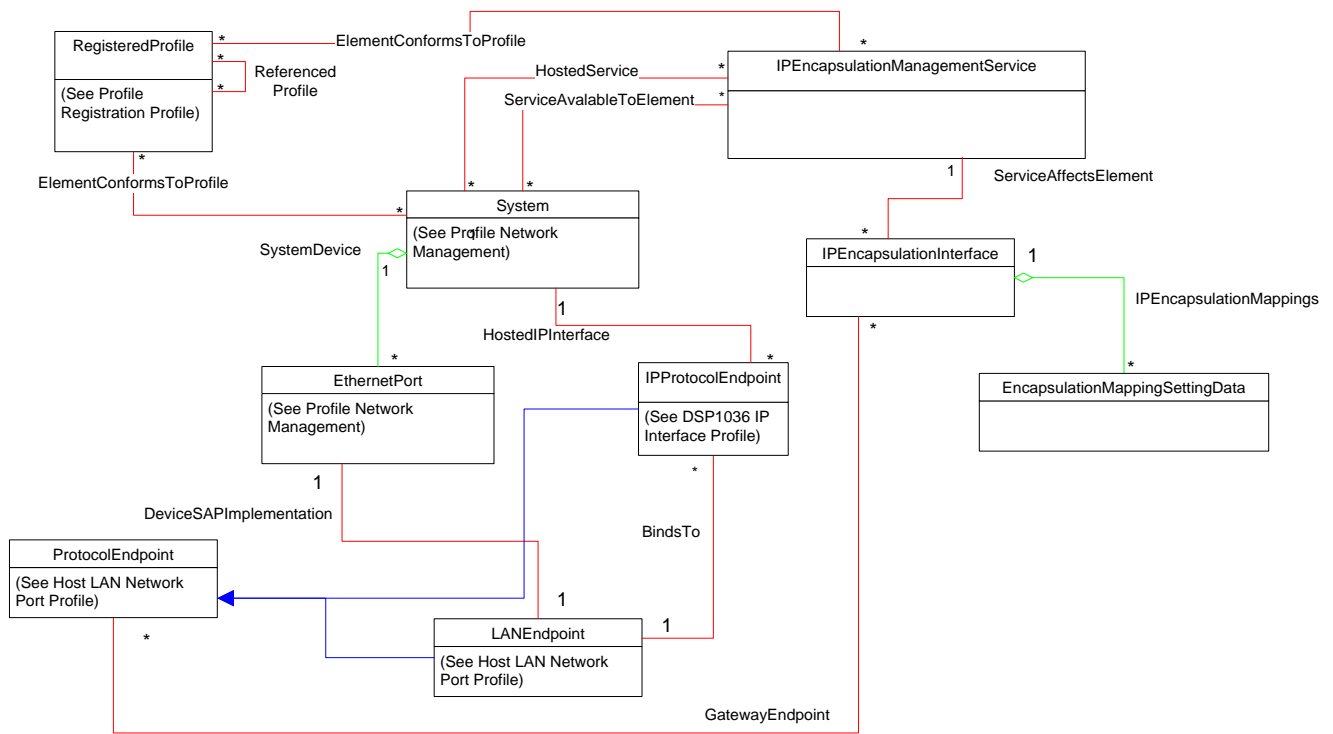
Profile Name	Organization	Version	Requirement	Description
Profile Registration	DMTF	1.0	Mandatory	None
Network Management	DMTF	1.0	Mandatory	None

275 6 Description

276 The *Network Management - Tunnel Management Profile* is a profile that will specify the CIM schema and
 277 use cases associated with the general and common aspects of Tunneling Management. This profile
 278 includes a specification of the CIM_IPEncapsulationManagementService and a set of associated CIM
 279 classes to configure and manage a Tunnel gateway.
 280

281 6.1 Class diagram

282 Figure 1 represents the class schema for the *Network Management - Tunnel Management Profile*. For
 283 simplicity, the CIM_ prefix has been removed from the names of the classes.
 284



285
286

287

Figure 1 – Network Management - Tunnel Management Profile

288

289 6.2 Tunneling Protocol Variants

290 A variety of tunneling protocols can be utilized in (virtual) overlay networks in order to extend disparate
 291 network segments between hosts (servers) for multi-tenant data center networks. These include Virtual
 292 Extensible LAN (VXLAN), Network Virtualization using Generic Encapsulation (NVGRE), state-less
 293 transport tunneling (STT) and IETF Network Virtualization Overlays 3 (NVO3).

294 Currently this profile supports VXLAN and NVGRE although the profile generically supports other L2 and
 295 L3 tunneling protocols such as STT, LISP, L2TPv3, MPLS, GRE, VXLAN-GPE, GENEVE and GUE.

296 6.3 IIPEncapsulationInterface

297 In this profile the IIPEncapsulationInterface is used to provide the configuration for an instance of an
 298 encapsulation service. This includes the ability to configure the endpoints that are on the provider network
 299 and the customer network and the virtual routing and forwarding tables that are to be used to determine
 300 the next hop routes required to route traffic between the two networks. An instance of
 301 IIPEncapsulationInterface is associated with the collection of CIM_EncapsulationMappingSettingData that
 302 is used to describe the policy to map each tunnel.

303 6.4 CIM_EncapsulationMappingSettingData

304 CIM_EncapsulationSettingData represents a lookup record contained in a mapping table, represented by
 305 the aggregation IPEncapsulationMappings. This table of records provide the required information
 306 generally indexed off of the target IP address of an incoming packet from the provider network. It provides
 307 the information required to construct the encapsulation header for the underlay network.

308 **6.5 CIM_IPEncapsulationManagementService**

309 The CIM_IPEncapsulationManagementService is the central class of this profile. The service has a set of
310 extrinsic methods to control the creation and removal of the instances required to create an IP
311 encapsulation gateway.

312 **7 Implementation**

313 This clause details the requirements related to the arrangement of instances and the properties of
314 instances for implementations of this profile

315 **7.1 Representing IP encapsulation management services**

316 **7.1.1 CIM_IPEncapsulationManagementService**

317 One or more instances of CIM_IPEncapsulationManagementService shall be instantiated.

318 These instances of CIM_IPEncapsulationManagementService shall be associated with an instance of the
319 scoping CIM_System class through an instance of CIM_HostedService.

320 The instances of the CIM_IPEncapsulationManagementService class shall also be associated to each
321 CIM_System subclass instance that may be used as the TargetInterface parameter of its
322 AddIPEncapsulationInterface () method through an instance of CIM_ServiceAvailableToElement.

323 IPEncapsulationInterface instances managed by or created through the use of an instance of
324 CIM_IPEncapsulationManagementService shall be associated to the
325 CIM_IPEncapsulationManagementService instance through an instance of CIM_ServiceAffectsElement.

326 **7.2 CIM_IPEncapsulationInterface**

327 Instances of CIM_IPEncapsulationInterface created as part of the execution of
328 AddIPEncapsulationInterface() method shall be associated with the instance of
329 CIM_IPEncapsulationManagementService from which the method call was made through an instance of
330 CIM_ServiceAffectsElement association.

331 An instance of CIM_IPEncapsulationInterface directly associated to an instance CIM_ProtocolEndpoint
332 as configured through the CASourceEndpoint or the PASourceEndpoint property in the representative
333 CIM_IPEncapsulationInterface instance shall be associated through an instance of a
334 CIM_GatewayEndpoint association.

335 Instances of CIM_IPProtocolEndpoint created as part of the execution of AddIPEncapsulationInterface()
336 method shall be associated to the instantiated instance of CIM_IPEncapsulationInterface from the same
337 method call through an instance of the CIM_GatewayEndpoint association.

338 **7.3 Representing the Protocol Endpoints of an Encapsulation gateway.**

339 Each instance of CIM_EncapsulationGateway shall have two associated instances of
340 CIM_ProtocolEndpoint or a subclass of CIM_ProtocolEndpoint one representing an endpoint that is a part
341 of the customer network and another representing an endpoint that is a part of the provider network.
342 These instances are associated through the CIM_GatewayConnection association instances as
343 described in 7.2. The association is made to these endpoints after the successful completion of a
344 CIM_IPEncapsulationManagementService.AddIPEncapsulationInterface() method.

345 If an instance of CIM_IPProtocolEndpoint is instantiated through a successful completion of a
346 CIM_IPEncapsulationManagementService.AddIPEncapsulationInterface() method where either a
347 CAIPEndpoint or an PAIPEndpoint parameter was populated, that instance shall be associated through

348 an instance of CIM_HostedIPInterface to the instance CIM_System that was specified in the
349 TargetSystem parameter of the method call.

350 7.4 Representing an encapsulation mapping table

351 7.4.1 CIM_EncapsulationMappingSettingData

352 Instances of CIM_EncapsulationMappingSettingData created as a result of the
353 AddIPEncapsulationInterface() or AddEncapsulationMappingSettingData () method shall be associated
354 to the instance of CIM_IPEncapsulationInterface contained in the EncapsulationGateway parameter of
355 the respective method through an aggregation instance of CIM_IPEncapsulationMappings.

356 8 Methods

357 This clause details the requirements for supporting intrinsic operations and extrinsic methods for the CIM
358 elements defined by this profile.

359 8.1 Extrinsic Methods

360 If synchronous execution of a method succeeds, the implementation shall set a return value of
361 0 (Completed with No Error).

362 If synchronous execution of a method fails, the implementation shall set a return value of 2 (Failed) or a
363 more specific return code as specified with the respective method.

364 If a method is executed as an asynchronous task, the implementation shall perform all of the following ac-
365 tions:

- 366 • Set a return value of 4096 (Job Started).
- 367 • Set the value of the Job output parameter to refer to an instance of the CIM_ConcreteJob class
368 that represents the asynchronous task.
- 369 • Set the values of the JobState and TimeOfLastStateChange properties in that instance to repre-
370 sent the state and last state change time of the asynchronous task.

371 In addition, the implementation may present state change indications as task state changes occur.

372 If the method execution as an asynchronous task succeeds, the implementation shall perform all of the
373 following actions:

- 374 • Set the value of the JobState property to 7 (Completed).
- 375 • Provide an instance of the CIM_AffectedJobEntity association with property values set as fol-
376 lows:
 - 377 – The value of the AffectedElement property shall refer to the object that represents the top-
378 level entity that was created or modified by the asynchronous task. For example, for the
379 CIM_IPConfigurationService. AddIPProtocolEndpoint() method, this is an instance of the
380 CIM_IPProtocolEndpoint class
 - 381 – The value of the AffectingElement property shall refer to the instance of the
382 CIM_ConcreteJob class that represents the completed asynchronous task.
 - 383 – The value of the first element in the ElementEffects[] array property (ElementEffects[0])
384 shall be set to 5 (Create) for the CIM_IPConfigurationService. AddIPProtocolEndpoint()
385 method. Otherwise, this value shall be 0 (Unknown).

386 If the method execution as an asynchronous task fails, the implementation shall set the value of the
387 JobState property to 9 (Killed) or 10 (Exception).

388 **8.1.1 Job parameter**

389 The implementation shall set the value of the Job parameter as a result of an asynchronous execution of
390 a method of the CIM_IPConfigurationService as follows:

- 391 • If the method execution is performed synchronously, the implementation shall set the value to
392 NULL.
- 393 • If the method execution is performed asynchronously, the implementation shall set the value to
394 refer to the instance of the CIM_ConcreteJob class that represents the asynchronous task.

395 **8.1.2 CIM_IPConfigurationService.AddIIPEncapsulationInterface()**

396 The implementation of the AddIIPEncapsulationInterface () method is optional, the provisions in this sub
397 clause apply in addition to behavior applicable to all extrinsic methods as specified in 8.1.

398 The successful execution of the AddIIPEncapsulationInterface () method shall create an instance of the
399 CIM_IPEncapsulationInterface class or a subclass of CIM_IPEncapsulationInterface and any required
400 associations as described in the sub clauses 7.2 required to instantiate a complete encapsulation
401 interface. In addition if the optional method parameter EncapsulationMappings is populated with
402 corresponding instances of the embedded CIM_EncapsulationMappingSettingData classes the
403 instantiated instances should be associated with the newly instantiated IIPEncapsulationInterface through
404 an instance of CIM_IPEncapsulationMappings. If the optional method parameters PAIPEndpoint or
405 CAIPEndpoint are populated with corresponding instances of the embedded CIM_IPProtocolEndpoint
406 classes the instantiated instances shall be associated with the newly instantiated
407 IIPEncapsulationInterface through an instance of CIM_GatewayEndpoint.

408 Table 2 contains requirements for parameters of this method.

409 **Table 2 – AddIIPEncapsulationInterface () Method: Parameters**

Qualifiers	Name	Type	Description/Values
IN	TargetSystem	CIM_System REF	See 8.1.2.1
IN	EncapsulationGateway	String	See 8.1.2.2
IN	PAIPEndpoint	String	See 8.1.2.2
IN	CAIPEndpoint	String	See 8.1.2.4
IN	EncapsulationMappings	String[]	See 8.1.2.5
OUT	ResultingGateway	IIPEncapsulationInterface REF	See 8.1.2.5
OUT	Job	CIM_ConcreteJob REF	See 8.1.1

410 **8.1.2.1 TargetSystem**

411 A required reference to a system or network. The supported target interfaces for a
412 CIM_IPEncapsulationInterface class or subclass shall be as described in the sub clauses of 7.2.

413 **8.1.2.2 EncapsulationGateway**

414 A required string containing an embedded instance of the class or subclass of
415 CIM_IPEncapsulationInterface describes the initial configuration of the resulting
416 CIM_IPEncapsulationInterface instance. The populated properties of the embedded instance should not
417 contain key properties, and any key property values may be ignored.

418 **8.1.2.3 PAIPEndpoint**

419 An optional string containing an embedded instance of the class or subclass of CIM_IPProtocolEndpoint
 420 that describes the initial configuration of a CIM_IPProtocolEndpoint that is on the provider network. The
 421 populated properties of the embedded instance should not contain key properties, and any key property
 422 values may be ignored.

423 **8.1.2.4 CAIPEndpoint**

424 An optional string containing an embedded instances of the class or subclass of CIM_IPProtocolEndpoint
 425 that describes the initial configuration of an CIM_IPProtocolEndpoint that is on the customer network. The
 426 populated properties of the embedded instance should not contain key properties, and any key property
 427 values may be ignored.

428 **8.1.2.5 EncapsulationMapping[]**

429 An optional array of strings containing embedded instances of the class or subclass of
 430 CIM_EncapsulationMappingSettingData that describes entries in a mapping table used by the
 431 encapsulation gateway to provide the encapsulation header information used by the gateway. The
 432 populated properties of the embedded CIM_EncapsulationMappingSettingData instances should not
 433 contain key properties, and any key property values may be ignored.

434 **8.1.2.6 ResultingInterface**

435 If the creation of the CIM_IPEncapsulationInterface is successful, the instance of the class
 436 CIM_IPEncapsulationInterface that represents the instantiated instance of CIM_IPEncapsulationInteface
 437 is returned.

438 **8.1.2.7 Job**

439 See 8.1.1

440 **8.1.3 CIM_IPConfigurationService.AddIPEncapsulationMappings()**

441 The implementation of the AddIPEncapsulationMappings () method is optional, the provisions in this sub
 442 clause apply in addition to behavior applicable to all extrinsic methods as specified in 8.1.

443 The successful execution of the AddIPEncapsulationMappings () method shall create or add to an array
 444 of instances of the CIM_EncapsulationMappingSettingData. The added instances of
 445 CIM_EncapsulationMappingSettingData shall be associated to the target IPEncapsulationInterface
 446 through an instance of CIM_IPEncapsulationMappings.

447 Table 2 contains requirements for parameters of this method.

448 **Table 3 – AddIPEncapsulationMappings () Method: Parameters**

Qualifiers	Name	Type	Description/Values
IN	TargetInterface	CIM_IPEncapsulationInterface REF	See 8.1.3.1
IN	EncapsulationMappings	String[]	See 8.1.3.2
OUT	Job	CIM_ConcreteJob REF	See 8.1.3.3

449 **8.1.3.1 TargetInterface**

450 A required reference to an instance of IPEncapsulationInterface class or subclass.

451 **8.1.3.2 EncapsulationMapping[]**

452 An optional array of strings containing embedded instances of the class or subclass of
 453 CIM_EncapsulationMappingSettingData that describes entries in a mapping table used by the
 454 encapsulation interface to provide the encapsulation header for the resultant IIPEncapsulationInterface.
 455 The populated properties of the embedded CIM_EncapsulationMappingSettingData instances should not
 456 contain key properties, and any key property values may be ignored. The resulting
 457 CIM_EncapsulationMappingSettingData instance shall be associated with the target instance of
 458 IIPEncapsulationInterface configured in the EncapsulationGateway parameter through an instance of
 459 CIM_IPEncapsulationMapping.

460 **8.1.3.3 Job**

461 See 8.1.1

462 **8.1.4 CIM_IPConfigurationService.RemoveIIPEncapsulationInterface()**

463 The implementation of the RemoveIIPEncapsulationInterface() method is optional, the provisions in this
 464 sub clause apply in addition to behavior applicable to all extrinsic methods as specified in 8.1.

465 The successful execution of the RemoveIIPEncapsulationInterface() method shall remove the instances
 466 referenced in the methods Gateway parameter and shall remove any associated CIM_SettingData
 467 instances.

468 Table 2 contains requirements for parameters of this method.

469 **Table 4 – RemoveIIPEncapsulationInterface() Method: Parameters**

Qualifiers	Name	Type	Description/Values
IN	Gateway	CIM_IPIIPEncapsulationInterface REF[]	See 8.1.4.1
OUT	Job	CIM_ConcreteJob REF	See 8.1.1

470 **8.1.4.1 Endpoint**

471 An array of references to the pair of the class CIM_IPIIPEncapsulationInterface instances that shall be
 472 removed.

473 **8.1.4.2 Job**

474 See 8.1.1

475 **8.1.5 CIM_IPConfigurationService.RemoveIIPEncapsulationMappings()**

476 The implementation of the RemoveIIPEncapsulationMappings() method is optional, the provisions in this
 477 sub clause apply in addition to behavior applicable to all extrinsic methods as specified in 8.1.

478 The successful execution of the RemoveIIPEncapsulationMappings() method shall remove the instances
 479 referenced in the methods EncapsulationMapping parameter.

480 Table 2 contains requirements for parameters of this method.

481 **Table 5 – RemoveIPEncapsulationMappings() Method: Parameters**

Qualifiers	Name	Type	Description/Values
IN	Encapsulation Mappings	CIM_EncapsulationMappingSettingData REF[]	See 8.1.5.1
OUT	Job	CIM_ConcreteJob REF	See 8.1.5.2

482 **8.1.5.1 EncapsulationMappings**

483 An array of references to the pair of the CIM_EncapsulationMappingSettingData instances that shall be
 484 removed.

485 **8.1.5.2 Job**

486 See 8.1.1

487 **8.2 Profile conventions for operations**

488 For each profile class (including associations), the implementation requirements for operations, including
 489 those in the following default list, are specified in class-specific subclauses of this clause.

490 The default list of operations is as follows:

- 491 • GetInstance
- 492 • EnumerateInstances
- 493 • EnumerateInstanceNames
- 494 • Associators
- 495 • AssociatorNames
- 496 • References
- 497 • ReferenceNames

498 **8.3 CIM_GatewayEndpoint**

499 Table 6 lists implementation requirements for operations. If implemented, these operations shall be
 500 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 6, all operations in
 501 the default list in 8.2 shall be implemented as defined in [DSP0200](#).

502 NOTE: Related profiles may define additional requirements on operations for the profile class.

503 **Table 6 – Operations: CIM_GatewayEndpoint**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

504 **8.4 CIM_BindsTo**

505 Table 6 lists implementation requirements for operations. If implemented, these operations shall be
 506 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 6, all operations in
 507 the default list in 8.2 shall be implemented as defined in [DSP0200](#).

508 NOTE Related profiles may define additional requirements on operations for the profile class.

509 **Table 7 – Operations: CIM_BindsToLANEndpoint**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

510 **8.5 CIM_HostedService**

511 Table 8 lists implementation requirements for operations. If implemented, these operations shall be
 512 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 8, all operations in
 513 the default list in 8.2 shall be implemented as defined in [DSP0200](#).

514 NOTE Related profiles may define additional requirements on operations for the profile class.

515 **Table 8 – Operations: CIM_HostedService**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

516 **8.6 CIM_HostedIPInterface**

517 Table 9 lists implementation requirements for operations. If implemented, these operations shall be
 518 implemented as defined in [DSP0200](#). In addition, and unless otherwise stated in Table 9, all operations in
 519 the default list in 8.2 shall be implemented as defined in [DSP0200](#).

520 NOTE Related profiles may define additional requirements on operations for the profile class.

521 **Table 9 – Operations: CIM_HostedIPInterface**

Operation	Requirement	Messages
Associators	Unspecified	None
AssociatorNames	Unspecified	None
References	Unspecified	None
ReferenceNames	Unspecified	None

522

523 8.7 CIM_IPEncapsulationManagementService

524 All operations in the default list in 8.2 shall be implemented as defined in [DSP0200](#).

525 8.8 IIPEncapsulationInterface

526 All operations in the default list in 8.2 shall be implemented as defined in [DSP0200](#).

527 NOTE Related profiles may define additional requirements on operations for the profile class.

528

529 **9 Use cases**

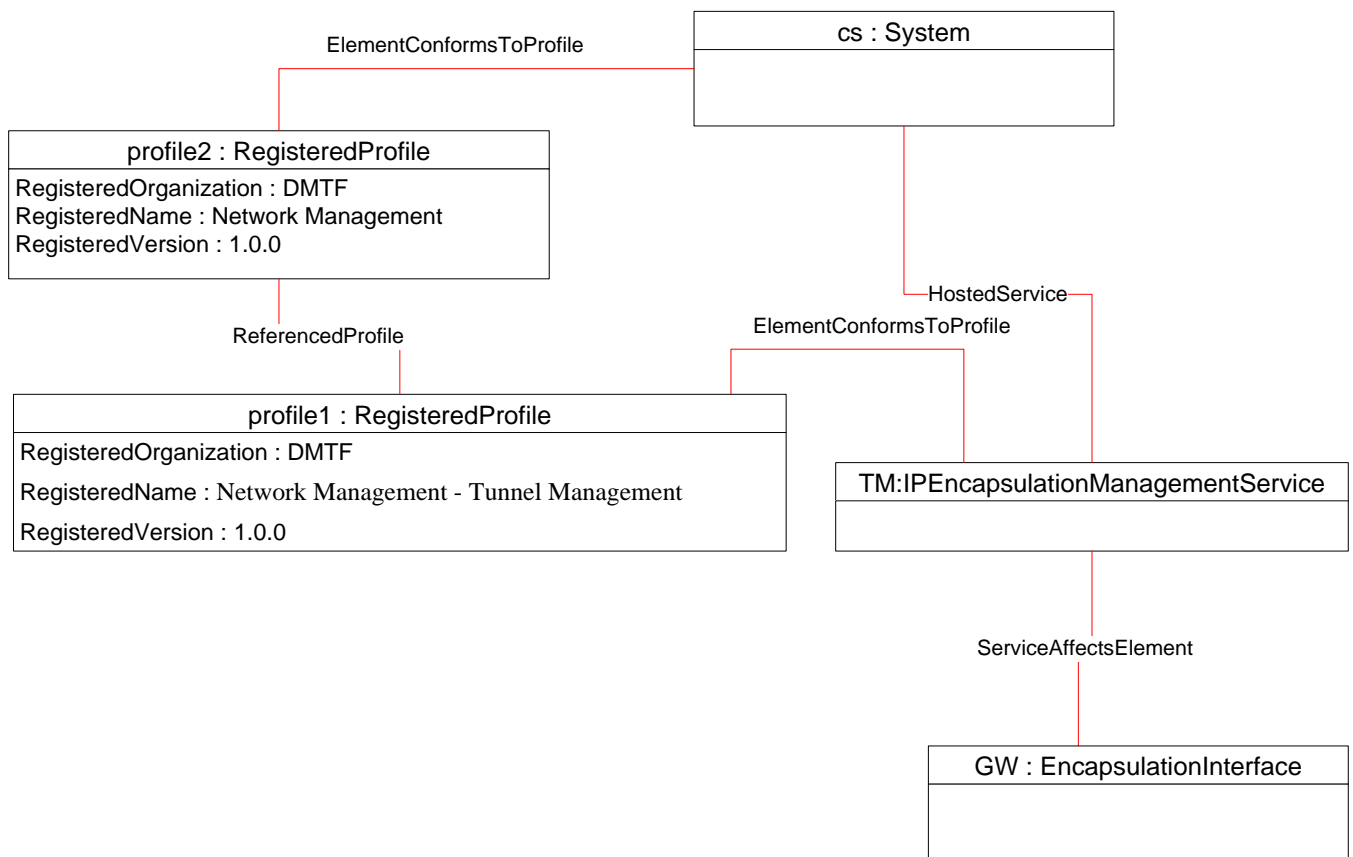
530 This clause contains object diagrams and use cases for the *Network Management - Tunnel Management*
 531 *Profile*.

532

533 **9.1 Profile Registration**

534 The object diagram in Figure 2 shows one possible method for advertising profile conformance. The
 535 instances of CIM_RegisteredProfile are used to identify the version of the Network Management - Tunnel
 536 Management Profile with which an instance of CIM_IPEncapsulationManagementService is conformant.
 537 An instance of CIM_RegisteredProfile exists for each profile that is instrumented in the system. One
 538 instance of CIM_RegisteredProfile identifies the System conforming to the Network Management Profile.
 539 The other instance identifies an instance of CIM_IPEncapsulationManagementService. The
 540 CIM_IPEncapsulationManagementService instance is scoped to an instance of CIM_System. This
 541 instance of CIM_System is conformant with the DMTF Network Management Profile version 1.0.0 as
 542 indicated by the CIM_ElementConformsToProfile association to the CIM_RegisteredProfile instance.

543



544

545

546

547

Figure 2 – Registered profile

548

549 9.2 L2 NVGRE Tunnel Gateway

550 The object diagram shown in Figure 3 contains the basic elements used to model a simple L2 NVGRE
551 tunnel gateway. The gateway as shown could be an example of an NVGRE tunnel gateway as part of a
552 virtual Ethernet switch. CIM_EthernetPort: E0/4 is a port available to a virtual computer system and
553 CIM_EthernetPort: E1/1 is the uplink to the physical Ethernet switch. In this example the virtual system
554 port (E0/4) would be on the customer network and the uplink port (E1/1) would be on the provider
555 network.

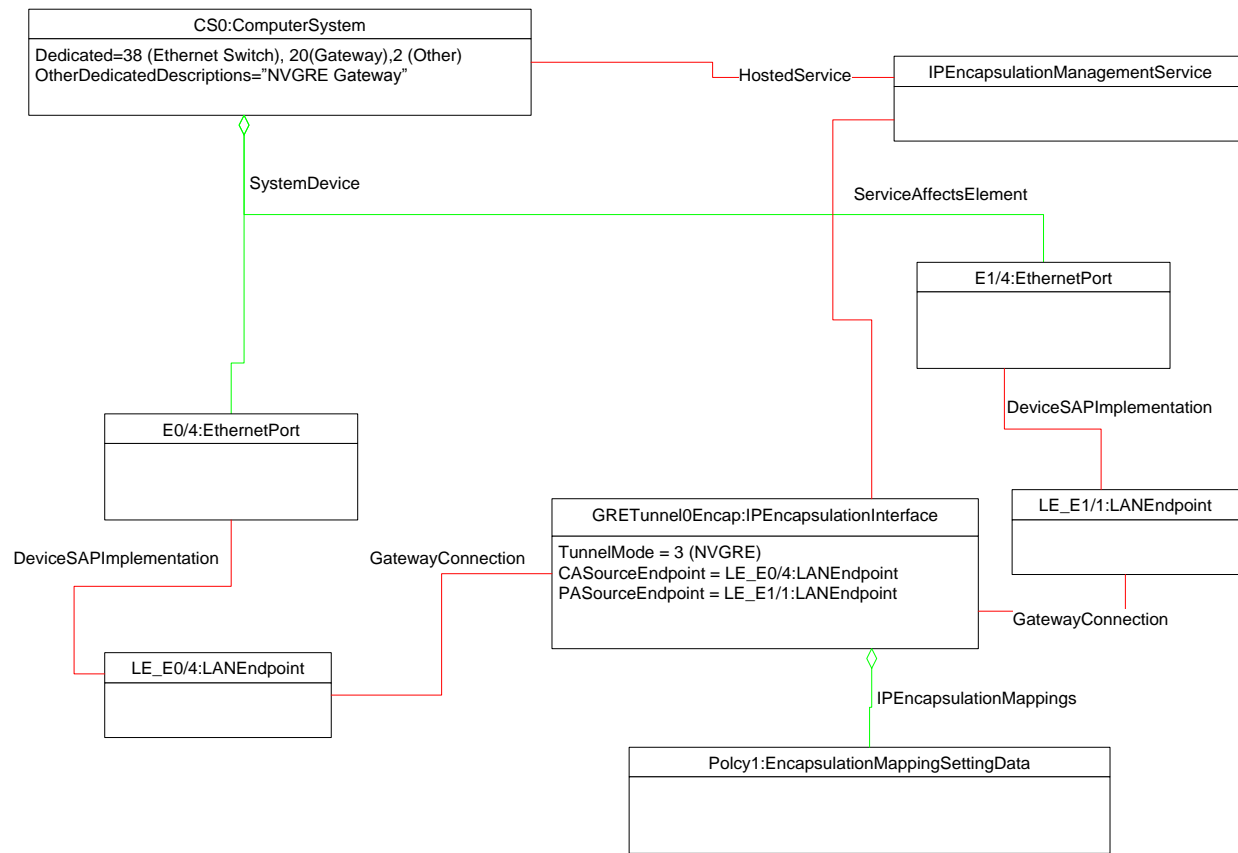
556 This gateway takes all network traffic from CIM_EthernetPort: E0/4 and encapsulates it based on the
557 mappings found in the instances of CIM_EncapsulationMappingSettingData and sends the encapsulated
558 traffic out on the provider network through CIM_EthernetPort: E1/1. Any traffic from the provider network
559 is de-capsulated and sent to the virtual machine through CIM_EthernetPort: E0/4.

560 This is a very simple instance diagram, not shown are many of the required properties of the relative
561 profiles for the objects shown.

562 The IIPEncapsulationInterface was created with a
563 CIM_IPConfigurationService.AddEncapsulationGateway() method with the following parameters.

- 564 • TargetInterface – WBEM URI reference to CIM_ComputerSystem:CS0
- 565 • EncapsulationGateway
 - 566 ○ Embedded Instance of IIPEncapsulationInterface {
 - 567 TunnelMode=4 (NVGRE)
 - 568 CASourceInterface = WBEM URI reference to CIM_EthernetPort: E0/4
 - 569 PASourceInterface = WBEM URI reference to CIM_EthernetPort: E1/1 }
 - 570
- 571 • EncapsulationMappings
 - 572 ○ Embedded Instance of EncapsulationMappingSettingData {
 - 573 CustomerAddress 10.1.0.125
 - 574 ProviderAddress 198.168.56.255
 - 575 MACAddress = 01:23:45:67:89:ab
 - 576 VSID = GUID for Virtual Subnet ID.
 - 577

578



579

580

Figure 3 – NVGRE Tunnel Gateway

581

582 9.3 Routed NVGRE Tunnel Gateway

583 The instance diagram shown below contains the basic elements used to model a routed NVGRE tunnel
 584 gateway. The gateway as shown could be an example of an NVGRE tunnel gateway as part of a network
 585 router. While this is a simplified instance diagram, the example shows a port E0/4 that is on the customer
 586 network and is configured with an instance of CIM_SwitchVirtualInterface an IP endpoint that is
 587 configured through VLAN encapsulation to accept traffic tagged with a specific VALN ID. Another port
 588 E1/4 is an IP enabled port on the provider network. The customer network and the information required to
 589 determine the next hop routes are represented in the Virtual Routing and Forwarding table
 590 VRF1_CA:VirtualRoutingAndForwarding. The provider network and the information required to determine
 591 the next hop routes for the provider network are represented in the Virtual Routing and Forwarding table
 592 VRF0_PA:VirtualRoutingAndForwarding.

593 LE_E0/4:SwitchedVirtualInterface, an IP protocol endpoint for port E0/1, is a member of the
 594 VRF1_CA:VirtualRoutingAndForwarding table. IPE_E1/4:IPProtocolEndpoint is a member of the
 595 VRF0_PA:VirtualRoutingAndForwarding table.

596 The router has an IPEncapsulationManagementService, ServiceNVGRE. This example shows the result
 597 of a AddEncapsulationGateway() method call on that service that creates a IIPEncapsulationInterface
 598 with IPProtocolEndpoint instances for both the customer and the provider networks respectively
 599 GRE Tunnel0CA and GRE Tunnel0PA. As shown in Figure 4 these IP protocol endpoints are also
 600 members of their respective virtual routing and forwarding tables.

601 This example shows the method call used to add a NVGRE IIPEncapsulationInterface.

602

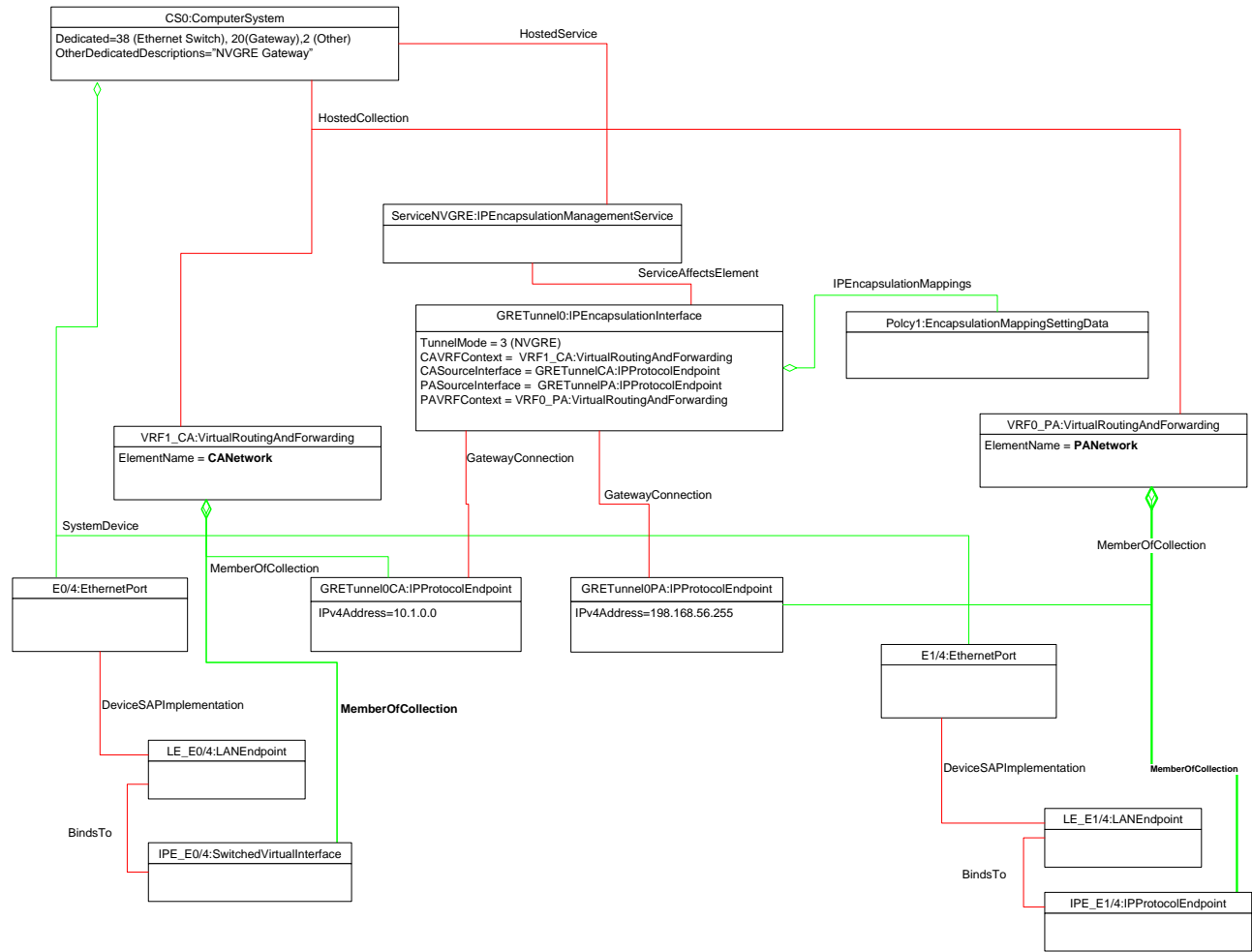
603 The IIPEncapsulationInterface was created with a

604 CIM_IPConfigurationService.AddEncapsulationGateway() method with the following parameters.

- 605 • TargetInterface – WBEM URI reference to CIM_ComputerSystem:CS0
- 606 • EncapsulationGateway
 - 607 ○ Embedded Instance of IIPEncapsulationInterface {
 - 608 TunnelMode=4 (NVGRE)
 - 609 CASourceInterface = null
 - 610 PASourceInterface = null
 - 611 CAVRFContext = WBEM URI reference to VRF1_CA:VirtualRoutingAndForwarding
 - 612 PAVRFContext = WBEM URI reference to VRF0_PA:VirtualRoutingAndForwarding
 - 613
- 614 • CAEndpoint
 - 615 ○ Embedded Instance of CIM_IPProtocolEndpoint {
 - 616 IPv4Address=10.1.0.0
 - 617 ProtocolIIFType=4060}
 - 618
- 619 • PAEndpoint
 - 620 ○ Embedded Instance of CIM_IPProtocolEndpoint {
 - 621 IPv4Address=198.168.0.0
 - 622 ProtocolIIFType=4060}
 - 623
 - 624
 - 625
- 626 • EncapsulationMappings
 - 627 ○ CustomerAddress 10.1.0.125
 - 628 ProviderAddress 198.168.56.255
 - 629 MACAddress = 01:23:45:67:89:ab
 - 630 VSID = GUID for Virtual Subnet ID.

631

632



633

634

Figure 4 – NVGRE Routed Tunnel Gateway

635 **10 CIM Elements**

636 Table 10 shows the instances of CIM Elements for this profile. Instances of the CIM Elements shall be
 637 implemented as described in Table 10. Clauses 7 (“Implementation”) and 8 (“Methods”) may impose
 638 additional requirements on these elements.

639 **Table 10 – CIM Elements: Network Management – Tunnel Management Profile**

640

Element Name	Requirement	Description
Classes		
CIM_BindsTo	Optional	See DSP1036 IP Interface Profile 1.1.1
CIM_BindsToLANEndpoint	Optional	See DSP1036 IP Interface Profile 1.1.1
CIM_HostedService	Conditional	See DSP1036 IP Interface Profile 1.1.1
CIM_HostedIPInterface	Conditional	See DSP1036 IP Interface Profile 1.1.1
CIM_IPProtocolEndpoint	Conditional	See DSP1036 IP Interface Profile 1.1.1
CIM_IPEncapsulationManagementService	Mandatory	See 7.1

Element Name	Requirement	Description
CIM_RegisteredProfile	Optional	See DSP1036 IP Interface Profile 1.1.1
CIM_ServiceAffectsElement	Conditional	See DSP1036 IP Interface Profile 1.1.1
CIM_ServiceAvailableToElement	Conditional	See DSP1036 IP Interface Profile 1.1.1
IIPEncapsulationInterface	Required	See 7.2
Indications		
None defined in this profile		

641 **10.1 CIM_BindsToLANEndpoint**

642 CIM_BindsToLANEndpoint relates the CIM_IPProtocolEndpoint instance with the CIM_LANEndpoint
 643 instance on which it depends. Table 11 provides information about the properties of
 644 CIM_BindsToLANEndpoint.

645 **Table 11 – Class: CIM_BindsToLANEndpoint**

Elements	Requirement	Description
Antecedent	Mandatory	Key: This shall be a reference to an instance of CIM_LANEndpoint. Cardinality 0..1
Dependent	Mandatory	Key: This shall be a reference to the Central Instance. Cardinality 1

646 **10.2 CIM_BindsTo**

647 CIM_BindsTo relates two pairs of CIM_ProtocolEndpoints together. Table 12 provides information about
 648 the properties of CIM_BindsTo.

649 **Table 12 – Class: CIM_BindsTo**

Elements	Requirement	Description
Antecedent	Mandatory	Key: This shall be a reference to an instance of CIM_EncapsulationGateways. Cardinality 1
Dependent	Mandatory	Key: This shall be a reference to the paired CIM_EncapsulationGateways 1

650 **10.3 CIM_EncapsulationMappingSettingData**

651 Contains one mapping lookup record for an IP encapsulation gateway. Table 13 provides information
 652 about the properties of CIM_EncapsulationSettingData.

653 **Table 13 – Class: CIM_EncapsulationMappingSettingData**

Elements	Requirement	Description
InstanceID	Mandatory	Key

656 **10.4 CIM_HostedService**

657 CIM_HostedService relates the CIM_IPEncapsulationManagementService instance to its scoping
 658 CIM_System instance. Table 14 provides information about the properties of CIM_HostedService.

659 **Table 14 – Class: CIM_HostedService**

Elements	Requirement	Description
Antecedent	Mandatory	Key: This shall be a reference to the Central Instance. Cardinality 1
Dependent	Mandatory	Key: This shall be a reference to an instance of CIM_IPEncapsulationManagementService. Cardinality *

660 **10.5 CIM_IPEncapsulationManagementService**

661 CIM_IPEncapsulationManagementService provides the methods to create and delete an encapsulation
 662 gateway interface. Table 15 provides information about the properties of
 663 CIM_IPEncapsulationManagementService.

664 **Table 15 – Class: CIM_IPEncapsulationManagementService**

Elements	Requirement	Description
SystemCreationClassName	Mandatory	Key
CreationClassName	Mandatory	Key
SystemName	Mandatory	Key
Name	Mandatory	Key
ElementName	Mandatory	Pattern ".*"
AddIPEncapsulationInterface()	Optional	See 8.1.2.
RemoveIPEncapsulationInterface()	Optional	See 8.1.4
AddIPEncapsulationMappings()	Optional	See 8.1.3
RemoveIPEncapsulationMappings()	Optional	See 8.1.5

665 **10.6 CIM_IPProtocolEndpoint**

666 CIM_IPProtocolEndpoint represents an IP interface that is associated with an IP encapsulation gateway
 667 or an Ethernet interface. Table 16 provides information about the properties of CIM_IPProtocolEndpoint.

668 **Table 16 – Class: CIM_IPProtocolEndpoint**

Elements	Requirement	Description
SystemCreationClassName	Mandatory	Key
CreationClassName	Mandatory	Key
SystemName	Mandatory	Key
Name	Mandatory	Key

Elements	Requirement	Description
NameFormat	Mandatory	See DSP1036 IP Interface Profile 1.1.1
ProtocolIFTType	Mandatory	See DSP1036 IP Interface Profile 1.1.1
RequestedState	Mandatory	See DSP1036 IP Interface Profile 1.1.1
EnabledState	Mandatory	See DSP1036 IP Interface Profile 1.1.1
ElementName	Mandatory	See DSP1036 IP Interface Profile 1.1.1
RequestStateChange()	Conditional	See DSP1036 IP Interface Profile 1.1.1
IPv4Address	Conditional	See DSP1036 IP Interface Profile 1.1.1
SubnetMask	Conditional	See DSP1036 IP Interface Profile 1.1.1
AddressOrigin	Mandatory	See DSP1036 IP Interface Profile 1.1.1
IPv6Address	Conditional	See DSP1036 IP Interface Profile 1.1.1
IPv6AddressType	Conditional	See DSP1036 IP Interface Profile 1.1.1
IPv6SubnetPrefixLength	Conditional	See DSP1036 IP Interface Profile 1.1.1

669 **10.7 CIM_IPEncapsulationInterface**

670 IPEncapsulationInterface represents the either the encapsulation or the decapsulation IP encapsulation
 671 interface used to route to connect two disjointed IP networks. Table 17 provides information about the
 672 additional properties of IPEncapsulationInterface that are in addition to those in CIM_IPProtocolEndpoint
 673 10.5 Table 16.

674 **Table 17 – Class: CIM_IPEncapsulationInterface**

Elements	Requirement	Description
SystemCreationClassName	Mandatory	Key
CreationClassName	Mandatory	Key
SystemName	Mandatory	Key
Name	Mandatory	Key
ElementName	Mandatory	Pattern ".*"
TunnelMode	Mandatory	See 7.2.
CAVRFCContext	Optional	See 7.2.
CASourceInterface	Required	See 7.2.
PAVRFCContext	Optional	See 7.2.
PASourceInterface	Required	See 7.2.

675

676 **10.8 CIM_RegisteredProfile**

677 CIM_RegisteredProfile identifies the *Network Management - Tunnel Management Profile* in order for a
678 client to determine whether an instance of CIM_IPProtocolEndpoint is conformant with this profile. The
679 CIM_RegisteredProfile class is defined by the [Profile Registration Profile](#). With the exception of the
680 mandatory values specified for the properties in Table 18, the behavior of the CIM_RegisteredProfile
681 instance is in accordance with the [Profile Registration Profile](#).

682 **Table 18 – Class: CIM_RegisteredProfile**

Elements	Requirement	Description
RegisteredName	Mandatory	This property shall have a value of "Tunnel Management".
RegisteredVersion	Mandatory	This property shall have a value of "1.0.0b".
RegisteredOrganization	Mandatory	This property shall have a value of "DMTF".

683
684
685
686

ANNEX A (informative)

Change log

Version	Date	Description
1.0.0ba	2015-06-12	DMTF Work in Progress
1.0.0b	2016-02-23	DMTF Work in Progress
1.0.0c	2015-10-1	Removed specialized mappings

687
688