



1
2
3
4

Document Number: DSP0819

Date: 2009-07-14

Version: 1.0.0

5 **DNS Client Profile SM CLP Command Mapping**
6 **Specification**

7 **Document Type: Specification**
8 **Document Status: DMTF Standard**
9 **Document Language: E**

10

11 Copyright notice

12 Copyright © 2006, 2009 Distributed Management Task Force, Inc. (DMTF). All rights reserved.

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

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27 withdrawn or modified after publication, and shall be indemnified and held harmless by any party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

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

33

34

CONTENTS

35	Foreword	5
36	Introduction	6
37	1 Scope	7
38	2 Normative References.....	7
39	2.1 Approved References	7
40	2.2 Other References.....	7
41	3 Terms and Definitions.....	7
42	4 Symbols and Abbreviated Terms.....	8
43	5 Recipes.....	9
44	6 Mappings.....	9
45	6.1 CIM_DNSGeneralSettingData	9
46	6.2 CIM_DNSSettingData	13
47	6.3 CIM_DNSProtocolEndpoint	16
48	6.4 CIM_ElementSettingData	19
49	6.5 CIM_SAPSAPDependency.....	25
50	6.6 CIM_HostedAccessPoint	27
51	6.7 CIM_RemoteAccessAvailableToElement.....	30
52	6.8 CIM_RemoteServiceAccessPoint.....	33
53	ANNEX A (informative) Change Log	36

54

55 Tables

56	Table 1 – Command Verb Requirements for CIM_DNSGeneralSettingData	9
57	Table 2 – Command Verb Requirements for CIM_DNSSettingData	13
58	Table 3 – Command Verb Requirements for CIM_DNSProtocolEndpoint	16
59	Table 4 – Command Verb Requirements for CIM_ElementSettingData	19
60	Table 5 – Command Verb Requirements for CIM_SAPSAPDependency.....	25
61	Table 6 – Command Verb Requirements for CIM_HostedAccessPoint	27
62	Table 7 – Command Verb Requirements for CIM_RemoteAccessAvailableToElement.....	31
63	Table 8 – Command Verb Requirements for CIM_RemoteServiceAccessPoint.....	33

64

66

Foreword

67 The *DNS Client Profile SM CLP Command Mapping Specification* (DSP0819) was prepared by the Server
68 Management Working Group.

69 **Conventions**

70 The pseudo-code conventions utilized in this document are the Recipe Conventions as defined in the
71 [SMI-S 1.1.0](#), section 7.6.

72 **Acknowledgements**

73 The authors wish to acknowledge the following participants from the DTMF Server Management Working
74 Group:

- 75 • Aaron Merkin – IBM
- 76 • Jon Hass – Dell
- 77 • Khachatur Papanyan – Dell
- 78 • Enoch Suen – Dell
- 79 • Jeff Hilland – HP
- 80 • Christina Shaw – HP
- 81 • Perry Vincent – Intel
- 82 • John Leung – Intel

83

84

Introduction

85 This document defines the SM CLP mapping for CIM elements described in the [DNS Client Profile](#). The
86 information in this specification, combined with the [SM CLP-to-CIM Common Mapping Specification 1.0](#),
87 is intended to be sufficient to implement SM CLP commands relevant to the classes, properties, and
88 methods described in the [DNS Client Profile](#) using CIM operations.

89 The target audience for this specification is implementers of the SM CLP support for the [DNS Client](#)
90 [Profile](#).

91 DNS Client Profile SM CLP Command Mapping Specification

92 1 Scope

93 This specification contains the requirements for an implementation of the SM CLP to provide access to,
94 and implement the behaviors of, the [DNS Client Profile](#).

95 2 Normative References

96 The following referenced documents are indispensable for the application of this document. For dated
97 references, only the edition cited applies. For undated references, the latest edition of the referenced
98 document (including any amendments) applies.

99 2.1 Approved References

100 DMTF DSP0216, *SM CLP-to-CIM Common Mapping Specification 1.0*,
101 http://www.dmtf.org/standards/published_documents/DSP0216_1.0.pdf

102 DMTF DSP1038, *DNS Client Profile 1.0*,
103 http://www.dmtf.org/standards/published_documents/DSP1038_1.0.pdf

104 SNIA, *Storage Management Initiative Specification (SMI-S) 1.1.0*,
105 http://www.snia.org/tech_activities/standards/curr_standards/smi

106 2.2 Other References

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

109 3 Terms and Definitions

110 For the purposes of this document, the following terms and definitions apply.

111 3.1

112 **can**

113 used for statements of possibility and capability, whether material, physical, or causal

114 3.2

115 **cannot**

116 used for statements of possibility and capability, whether material, physical or causal

117 3.3

118 **conditional**

119 indicates requirements to be followed strictly in order to conform to the document when the specified
120 conditions are met

121 3.4

122 **mandatory**

123 indicates requirements to be followed strictly in order to conform to the document and from which no
124 deviation is permitted

- 125 **3.5**
126 **may**
127 indicates a course of action permissible within the limits of the document
- 128 **3.6**
129 **need not**
130 indicates a course of action permissible within the limits of the document
- 131 **3.7**
132 **optional**
133 indicates a course of action permissible within the limits of the document
- 134 **3.8**
135 **shall**
136 indicates requirements to be followed strictly in order to conform to the document and from which no
137 deviation is permitted
- 138 **3.9**
139 **shall not**
140 indicates requirements to be followed strictly in order to conform to the document and from which no
141 deviation is permitted
- 142 **3.10**
143 **should**
144 indicates that among several possibilities, one is recommended as particularly suitable, without
145 mentioning or excluding others, or that a certain course of action is preferred but not necessarily required
- 146 **3.11**
147 **should not**
148 indicates that a certain possibility or course of action is deprecated but not prohibited

149 **4 Symbols and Abbreviated Terms**

150 The following symbols and abbreviations are used in this document.

- 151 **4.1**
152 **CIM**
153 Common Information Model
- 154 **4.2**
155 **CLP**
156 Command Line Protocol
- 157 **4.3**
158 **DMTF**
159 Distributed Management Task Force
- 160 **4.4**
161 **IETF**
162 Internet Engineering Task Force

163 **4.5**
 164 **SM**
 165 Server Management

166 **4.6**
 167 **SMI-S**
 168 Storage Management Initiative Specification

169 **4.7**
 170 **SNIA**
 171 Storage Networking Industry Association

172 **4.8**
 173 **UFsT**
 174 User Friendly selection Tag

175 **5 Recipes**

176 The following is a list of the common recipes used by the mappings in this specification. For a definition of
 177 each recipe, see *SM CLP-to-CIM Common Mapping Specification 1.0* ([DSP0216](#)).

- 178 • smShowInstance()
- 179 • smShowInstances()
- 180 • smSetInstance()
- 181 • smShowAssociationInstances()
- 182 • smShowAssociationInstance()

183 This mapping does not define any recipes for local reuse.

184 **6 Mappings**

185 The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in
 186 the [DNS Client Profile](#). Requirements specified here related to support for a CLP verb for a particular
 187 class are solely within the context of this profile.

188 **6.1 CIM_DNSGeneralSettingData**

189 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

190 Table 1 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 191 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 192 verb and target. Table 1 is for informational purposes only; in case of a conflict between Table 1 and
 193 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 194 information in Table 1.

195 **Table 1 – Command Verb Requirements for CIM_DNSGeneralSettingData**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	

Command Verb	Requirement	Comments
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.1.2.
show	Shall	See 6.1.3.
start	Not supported	
stop	Not supported	

196 No mapping is defined for the following verbs for the specified target: create, delete, dump, load,
197 reset, start, and stop.

198 6.1.1 Ordering of Results

199 When results are returned for multiple instances of CIM_DNSGeneralSettingData, implementations shall
200 utilize the following algorithm to produce the natural (that is, default) ordering:

- 201 • Results for CIM_DNSGeneralSettingData are unordered; therefore, no algorithm is defined.

202 6.1.2 Set

203 This section describes how to implement the `set` verb when it is applied to an instance of
204 CIM_DNSGeneralSettingData. Implementations may support the use of the `set` verb with
205 CIM_DNSGeneralSettingData.

206 The `set` verb is used to modify the properties of the CIM_DNSGeneralSettingData instance.

207 6.1.2.1 General Usage of Set for a Single Property

208 This command form corresponds to the general usage of the `set` verb to modify a single property of a
209 target instance. This is the most common case.

210 The requirement for supporting modification of a property using this command form shall be equivalent to
211 the requirement for supporting modification of the property using the ModifyInstance operation as defined
212 in the [DNS Client Profile](#).

213 6.1.2.1.1 Command Form

```
214 set <CIM_DNSGeneralSettingData single instance> <propertyname>=<propertyvalue>
```

215 6.1.2.1.2 CIM Requirements

216 See CIM_DNSGeneralSettingData in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
217 modifiable properties.

218 6.1.2.1.3 Behavior Requirements

```
219 $instance=<CIM_DNSGeneralSettingData single instance>
220 #propertyName[] = {<propertyname>};
221 #propertyValues[] = {<propertyvalue>};
222 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
223 &smEnd;
```

224 6.1.2.2 General Usage of Set for Multiple Properties

225 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
226 target instance where there is not an explicit relationship between the properties. This is the most
227 common case.

228 The requirement for supporting modification of a property using this command form shall be equivalent to
229 the requirement for supporting modification of the property using the `ModifyInstance` operation as defined
230 in the [DNS Client Profile](#).

231 6.1.2.2.1 Command Form

```
232 set <CIM_DNSGeneralSettingData single instance> <propertyname1>=<propertyvalue1>  
233 <propertynamen>=<propertyvaluen>
```

234 6.1.2.2.2 CIM Requirements

235 See `CIM_DNSGeneralSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
236 mandatory properties.

237 6.1.2.2.3 Behavior Requirements

```
238 $instance=<CIM_DNSGeneralSettingData single instance>  
239 #propertyNames[] = {<propertyname>};  
240 for #i < n  
241 {  
242     #propertyNames[#i] = <propertyname#i>  
243     #propertyValues[#i] = <propertyvalue#i>  
244 }  
245 &smSetInstance ( $instance, #propertyNames[], #propertyValues[] );  
246 &smEnd;
```

247 6.1.3 Show

248 This section describes how to implement the `show` verb when applied to an instance of
249 `CIM_DNSGeneralSettingData`. Implementations shall support the use of the `show` verb with
250 `CIM_DNSGeneralSettingData`.

251 The `show` verb is used to display information about `CIM_DNSGeneralSettingData`.

252 6.1.3.1 Show a Single Instance

253 This command form is for the `show` verb applied to a single instance of `CIM_DNSGeneralSettingData`.

254 6.1.3.1.1 Command Form

```
255 show <CIM_DNSGeneralSettingData single instance>
```

256 6.1.3.1.2 CIM Requirements

257 See `CIM_DNSGeneralSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
258 mandatory properties.

259 6.1.3.1.3 Behavior Requirements

260 6.1.3.1.3.1 Preconditions

261 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

262 **6.1.3.1.3.2 Pseudo Code**

```

263 $instance=<CIM_DNSGeneralSettingData single instance>
264 #propertylist[] = NULL;
265 if (false == #all)
266     {
267         #propertylist[] = { //all mandatory non-key properties }
268     }
269 &smShowInstance ( $instance.getInstancePath(), #propertylist[] );
270 &smEnd;

```

271 **6.1.3.2 Show Multiple Instances in Settings Collection**

272 This command form is for the `show` verb applied to multiple instances of `CIM_DNSGeneralSettingData`.
 273 This command form corresponds to UFsT-based selection within a scoping system.

274 **6.1.3.2.1 Command Form**

```

275 show <CIM_DNSGeneralSettingData multiple instances>

```

276 **6.1.3.2.2 CIM Requirements**

277 See `CIM_DNSGeneralSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
 278 mandatory properties.

279 **6.1.3.2.3 Behavior Requirements**280 **6.1.3.2.3.1 Preconditions**

281 `$containerInstance` contains the instance of `CIM_ConcreteCollection` for which related
 282 `CIM_DNSGeneralSettingData` instances are displayed. The `CIM_DNSGeneralSettingData` instance is
 283 associated with an instance of `CIM_ConcreteCollection` via an instance of the `CIM_MemberOfCollection`
 284 association.

285 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

286 **6.1.3.2.3.2 Pseudo Code**

```

287 #propertylist[] = NULL;
288 if (false == #all)
289     {
290         #propertylist[] = { //all mandatory non-key properties }
291     }
292 &smShowInstances ( "CIM_DNSGeneralSettingData", "CIM_MemberOfCollection",
293     $containerInstance.getInstancePath(), #propertylist[] );
294 &smEnd;

```

295 **6.1.3.3 Show Multiple Instances**

296 This command form is for the `show` verb applied to multiple instances of `CIM_DNSGeneralSettingData`.
 297 This command form corresponds to UFsT-based selection within a scoping system.

298 **6.1.3.3.1 Command Form**

```

299 show <CIM_DNSGeneralSettingData multiple instances>

```

300 **6.1.3.3.2 CIM Requirements**

301 See CIM_DNSGeneralSettingData in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
 302 mandatory properties.

303 **6.1.3.3.3 Behavior Requirements**

304 **6.1.3.3.3.1 Preconditions**

305 \$containerInstance contains the instance of CIM_ConcreteCollection for which we are displaying
 306 related CIM_DNSGeneralSettingData instances. SM ME Addressing requires that the
 307 CIM_DNSGeneralSettingData instance be associated with an instance of CIM_ConcreteCollection via an
 308 instance of the CIM_MemberOfCollection association.

309 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

310 **6.1.3.3.3.2 Pseudo Code**

```

311 #propertylist[] = NULL;
312 if (false == #all)
313     {
314         #propertylist[] = { //all mandatory non-key properties }
315     }
316 &smShowInstances ( "CIM_DNSGeneralSettingData", "CIM_MemberOfCollection",
317     $containerInstance.getInstancePath(), #propertylist[] );
318 &smEnd;
    
```

319 **6.2 CIM_DNSSettingData**

320 The cd and help verbs shall be supported as described in [DSP0216](#).

321 Table 2 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 322 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 323 verb and target. Table 2 is for informational purposes only; in case of a conflict between Table 2 and
 324 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 325 information in Table 2.

326 **Table 2 – Command Verb Requirements for CIM_DNSSettingData**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.2.2.
show	Shall	See 6.2.3.
start	Not supported	
stop	Not supported	

327 No mapping is defined for the following verbs for the specified target: create, delete, dump, load,
 328 reset, start, and stop.

329 6.2.1 Ordering of Results

330 When results are returned for multiple instances of CIM_DNSSettingData, implementations shall utilize
331 the following algorithm to produce the natural (that is, default) ordering:

- 332 • Results for CIM_DNSSettingData are unordered; therefore, no algorithm is defined.

333 6.2.2 Set

334 This section describes how to implement the `set` verb when it is applied to an instance of
335 CIM_DNSSettingData. Implementations may support the use of the `set` verb with CIM_DNSSettingData.

336 The `set` verb is used to modify the properties of the CIM_DNSSettingData instance.

337 6.2.2.1 General Usage of Set for a Single Property

338 This command form corresponds to the general usage of the `set` verb to modify a single property of a
339 target instance. This is the most common case.

340 The requirement for supporting modification of a property using this command form shall be equivalent to
341 the requirement for supporting modification of the property using the ModifyInstance operation as defined
342 in the [DNS Client Profile](#).

343 6.2.2.1.1 Command Form

```
344 set <CIM_DNSSettingData single instance> <propertyname>=<propertyvalue>
```

345 6.2.2.1.2 CIM Requirements

346 See CIM_DNSSettingData in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
347 modifiable properties.

348 6.2.2.1.3 Behavior Requirements

```
349 $instance=<CIM_DNSSettingData single instance>  
350 #propertyName[] = {<propertyname>};  
351 #propertyValues[] = {<propertyvalue>};  
352 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );  
353 &smEnd;
```

354 6.2.2.2 General Usage of Set for Multiple Properties

355 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
356 target instance where there is not an explicit relationship between the properties. This is the most
357 common case.

358 The requirement for supporting modification of a property using this command form shall be equivalent to
359 the requirement for supporting modification of the property using the ModifyInstance operation as defined
360 in the [DNS Client Profile](#).

361 6.2.2.2.1 Command Form

```
362 set <CIM_DNSSettingData single instance> <propertyname1>=<propertyvalue1>  
363 <propertynamen>=<propertyvaluen>
```

364 6.2.2.2.2 CIM Requirements

365 See CIM_DNSSettingData in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
366 mandatory properties.

367 6.2.2.2.3 Behavior Requirements

```

368 $instance=<CIM_DNSSettingData single instance>
369 #propertyName[] = {<propertyname>};
370 for #i < n
371 {
372     #propertyName[#i] = <propertyname#i>
373     #propertyValue[#i] = <propertyvalue#i>
374 }
375 &smSetInstance ( $instance, #propertyName[], #propertyValue[] );
376 &smEnd;

```

377 6.2.3 Show

378 This section describes how to implement the `show` verb when applied to an instance of
 379 `CIM_DNSSettingData`. Implementations shall support the use of the `show` verb with
 380 `CIM_DNSSettingData`.

381 The `show` verb is used to display information about `CIM_DNSSettingData`.

382 6.2.3.1 Show a Single Instance

383 This command form is for the `show` verb applied to a single instance of `CIM_DNSSettingData`.

384 6.2.3.1.1 Command Form

```

385 show <CIM_DNSSettingData single instance>

```

386 6.2.3.1.2 CIM Requirements

387 See `CIM_DNSSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
 388 mandatory properties.

389 6.2.3.1.3 Behavior Requirements

390 6.2.3.1.3.1 Preconditions

391 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

392 6.2.3.1.3.2 Pseudo Code

```

393 $instance=<CIM_DNSSettingData single instance>
394 #propertylist[] = NULL;
395 if (false == #all)
396 {
397     #propertylist[] = { //all mandatory non-key properties }
398 }
399 &smShowInstance ( $instance.getInstancePath(), #propertylist[] );
400 &smEnd;

```

401 6.2.3.2 Show Multiple Instances

402 This command form is for the `show` verb applied to multiple instances of `CIM_DNSSettingData`. This
 403 command form corresponds to UFsT-based selection within a scoping system.

404 **6.2.3.2.1 Command Form**405 `show <CIM_DNSSettingData multiple instances>`406 **6.2.3.2.2 CIM Requirements**407 See CIM_DNSSettingData in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
408 mandatory properties.409 **6.2.3.2.3 Behavior Requirements**410 **6.2.3.2.3.1 Preconditions**411 \$containerInstance contains the instance of CIM_IPAssignmentSettingData for which related
412 CIM_DNSSettingData instances are displayed. The CIM_DNSSettingData instance is associated with an
413 instance of CIM_IPAssignmentSettingData via an instance of the CIM_OrderedComponent association.

414 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

415 **6.2.3.2.3.2 Pseudo Code**416

```
#propertylist[] = NULL;
417 if (false == #all)
418 {
419     #propertylist[] = { //all mandatory non-key properties }
420 }
421 &smShowInstances ( "CIM_DNSSettingData", "CIM_OrderedComponent",
422     $containerInstance.getInstancePath(), #propertylist[] );
423 &smEnd;
```

424 **6.3 CIM_DNSProtocolEndpoint**425 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).426 Table 3 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
427 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
428 verb and target. Table 3 is for informational purposes only; in case of a conflict between Table 3 and
429 requirements detailed in the following sections, the text detailed in the following sections supersedes the
430 information in Table 3.431 **Table 3 – Command Verb Requirements for CIM_DNSProtocolEndpoint**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.3.2.
show	Shall	See 6.3.3.
start	Not supported	
stop	Not supported	

432 No mapping is defined for the following verbs for the specified target: create, delete, dump, load,
433 reset, start, and stop.

434 6.3.1 Ordering of Results

435 When results are returned for multiple instances of CIM_DNSProtocolEndpoint, implementations shall
436 utilize the following algorithm to produce the natural (that is, default) ordering:

- 437 • Results for CIM_DNSProtocolEndpoint are unordered; therefore, no algorithm is defined.

438 6.3.2 Set

439 This section describes how to implement the `set` verb when it is applied to an instance of
440 CIM_DNSProtocolEndpoint. Implementations may support the use of the `set` verb with
441 CIM_DNSProtocolEndpoint.

442 The `set` verb is used to modify descriptive properties of the CIM_DNSProtocolEndpoint instance.

443 6.3.2.1 General Usage of Set for a Single Property

444 This command form corresponds to the general usage of the `set` verb to modify a single property of a
445 target instance. This is the most common case.

446 The requirement for supporting modification of a property using this command form shall be equivalent to
447 the requirement for supporting modification of the property using the ModifyInstance operation as defined
448 in the [DNS Client Profile](#).

449 6.3.2.1.1 Command Form

```
450 set <CIM_DNSProtocolEndpoint single instance> <propertyname>=<propertyvalue>
```

451 6.3.2.1.2 CIM Requirements

452 See CIM_DNSProtocolEndpoint in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
453 modifiable properties.

454 6.3.2.1.3 Behavior Requirements

```
455 $instance=<CIM_DNSProtocolEndpoint single instance>
456 #propertyName[] = {<propertyname>};
457 #propertyValues[] = {<propertyvalue>};
458 &smSetInstance ( $instance, #propertyName[], #propertyValues[] );
459 &smEnd;
```

460 6.3.2.2 General Usage of Set for Multiple Properties

461 This command form corresponds to the general usage of the `set` verb to modify multiple properties of a
462 target instance where there is not an explicit relationship between the properties. This is the most
463 common case.

464 The requirement for supporting modification of a property using this command form shall be equivalent to
465 the requirement for supporting modification of the property using the ModifyInstance operation as defined
466 in the [DNS Client Profile](#).

467 6.3.2.2.1 Command Form

```
468 set <CIM_DNSProtocolEndpoint single instance> <propertyname1>=<propertyvalue1>
469 <propertynamen>=<propertyvaluen>
```

470 6.3.2.2.2 CIM Requirements

471 See CIM_DNSProtocolEndpoint in the "CIM Elements" section of the [DNS Client Profile](#) for the list of
472 mandatory properties.

473 6.3.2.2.3 Behavior Requirements

```
474 $instance=<CIM_DNSProtocolEndpoint single instance>
475 #propertyName[] = {<propertyname>};
476 for #i < n
477 {
478     #propertyName[#i] = <propertyname#i>
479     #propertyValue[#i] = <propertyvalue#i>
480 }
481 &smSetInstance ( $instance, #propertyName[], #propertyValue[] );
482 &smEnd;
```

483 6.3.3 Show

484 This section describes how to implement the `show` verb when applied to an instance of
485 CIM_DNSProtocolEndpoint. Implementations shall support the use of the `show` verb with
486 CIM_DNSProtocolEndpoint.

487 The `show` verb is used to display information about the IP interface.

488 6.3.3.1 Show a Single Instance

489 This command form is for the `show` verb applied to a single instance of CIM_DNSProtocolEndpoint.

490 6.3.3.1.1 Command Form

```
491 show <CIM_DNSProtocolEndpoint single instance>
```

492 6.3.3.1.2 CIM Requirements

493 See CIM_DNSProtocolEndpoint in the "CIM Elements" section of the [DNS Client Profile](#) for the list of
494 mandatory properties.

495 6.3.3.1.3 Behavior Requirements

496 6.3.3.1.3.1 Preconditions

497 #all is true if the "-all" option was specified with the command; otherwise, #all is false.

498 6.3.3.1.3.2 Pseudo Code

```
499 $instance=<CIM_DNSProtocolEndpoint single instance>
500 #propertylist[] = NULL;
501 if (false == #all)
502 {
503     #propertylist[] = { //all mandatory non-key properties };
504 }
505 &smShowInstance ( $instance.getInstancePath(), #propertylist[] );
506 &smEnd;
```

507 **6.3.3.2 Show Multiple Instances**

508 This command form is for the `show` verb applied to multiple instances of `CIM_DNSProtocolEndpoint`. This
 509 command form corresponds to UFT-based selection within a scoping system.

510 **6.3.3.2.1 Command Form**

511 `show <CIM_DNSProtocolEndpoint multiple instances>`

512 **6.3.3.2.2 CIM Requirements**

513 See `CIM_DNSProtocolEndpoint` in the "CIM Elements" section of the [DNS Client Profile](#) for the list of
 514 mandatory properties.

515 **6.3.3.2.3 Behavior Requirements**

516 **6.3.3.2.3.1 Preconditions**

517 `$containerInstance` contains the instance of `CIM_ComputerSystem` for which we are displaying
 518 scoped `CIM_DNSProtocolEndpoint` instances. The [DNS Client Profile](#) requires that the
 519 `CIM_DNSProtocolEndpoint` instance be associated with its scoping system via an instance of the
 520 `CIM_HostedAccessPoint` association.

521 `#all` is true if the "-all" option was specified with the command; otherwise, `#all` is false.

522 **6.3.3.2.3.2 Pseudo Code**

```
523 #propertylist[] = NULL;
524 if (false == #all)
525     {
526         #propertylist[] = { //all mandatory non-key properties };
527     }
528 &smShowInstances ( "CIM_DNSProtocolEndpoint", "CIM_HostedAccessPoint",
529     $containerInstance.getInstancePath(), #propertylist[] );
530 &smEnd;
```

531 **6.4 CIM_ElementSettingData**

532 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

533 Table 4 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 534 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 535 verb and target. Table 4 is for informational purposes only; in case of a conflict between Table 4 and
 536 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 537 information in Table 4.

538 **Table 4 – Command Verb Requirements for CIM_ElementSettingData**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	

Command Verb	Requirement	Comments
set	Not supported	
show	Shall	See 6.4.2.
start	Not supported	
stop	Not supported	

539 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
540 `reset`, `set`, `start`, and `stop`.

541 **6.4.1 Ordering of Results**

542 When results are returned for multiple instances of `CIM_ElementSettingData`, implementations shall
543 utilize the following algorithm to produce the natural (that is, default) ordering:

- 544 • Results for `CIM_ElementSettingData` are unordered; therefore, no algorithm is defined.

545 **6.4.2 Show**

546 This section describes how to implement the `show` verb when applied to an instance of
547 `CIM_ElementSettingData`. Implementations shall support the use of the `show` verb with
548 `CIM_ElementSettingData`.

549 The `show` command is used to display information about the `CIM_ElementSettingData` instance or
550 instances.

551 **6.4.2.1 Show Multiple Instances – CIM_DNSProtocolEndpoint Reference**

552 This command form is for the `show` verb applied to multiple instances. This command form corresponds
553 to a `show` command issued against `CIM_ElementSettingData` where only one reference is specified and
554 the reference is to an instance of `CIM_DNSProtocolEndpoint`.

555 **6.4.2.1.1 Command Form**

```
556 show <CIM_ElementSettingData multiple instances>
```

557 **6.4.2.1.2 CIM Requirements**

558 See `CIM_ElementSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
559 mandatory properties.

560 **6.4.2.1.3 Behavior Requirements**

561 **6.4.2.1.3.1 Preconditions**

562 `$.instance` contains the instance of `CIM_DNSProtocolEndpoint` which is referenced by
563 `CIM_ElementSettingData`.

564 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

565 6.4.2.1.3.2 Pseudo Code

```

566 #propertylist[] = NULL;
567 if (false == #all)
568     {
569         #propertylist[] = { //all mandatory non-key properties };
570     }
571 &smShowAssociationInstances ( "CIM_ElementSettingData", $instance.getInstancePath(),
572     #propertylist[] );
573 &smEnd;

```

574 6.4.2.2 Show Multiple Instances – CIM_ComputerSystem Reference

575 This command form is for the `show` verb applied to multiple instances. This command form corresponds
 576 to a `show` command issued against `CIM_ElementSettingData` where only one reference is specified and
 577 the reference is to an instance of `CIM_ComputerSystem`.

578 6.4.2.2.1 Command Form

```
579 show <CIM_ElementSettingData multiple instances>
```

580 6.4.2.2.2 CIM Requirements

581 See `CIM_ElementSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
 582 mandatory properties.

583 6.4.2.2.3 Behavior Requirements

584 6.4.2.2.3.1 Preconditions

585 `$instance` contains the instance of `CIM_ComputerSystem` which is referenced by
 586 `CIM_ElementSettingData`.

587 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

588 6.4.2.2.3.2 Pseudo Code

```

589 #propertylist[] = NULL;
590 if (false == #all)
591     {
592         #propertylist[] = { //all mandatory non-key properties };
593     }
594 &smShowAssociationInstances ( "CIM_ElementSettingData", $instance.getInstancePath(),
595     #propertylist[] );
596 &smEnd;

```

597 6.4.2.3 Show Multiple Instances – CIM_DNSSettingData Reference

598 This command form is for the `show` verb applied to multiple instances. This command form corresponds
 599 to a `show` command issued against `CIM_ElementSettingData` where only one reference is specified and
 600 the reference is to an instance of `CIM_DNSSettingData`.

601 6.4.2.3.1 Command Form

```
602 show <CIM_ElementSettingData multiple instances>
```

603 **6.4.2.3.2 CIM Requirements**

604 See CIM_ElementSettingData in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
605 mandatory properties.

606 **6.4.2.3.3 Behavior Requirements**607 **6.4.2.3.3.1 Preconditions**

608 \$instance contains the instance of CIM_DNSSettingData which is referenced by
609 CIM_ElementSettingData.

610 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

611 **6.4.2.3.3.2 Pseudo Code**

```
612 #propertylist[] = NULL;
613 if (false == #all)
614     {
615         #propertylist[] = { //all mandatory non-key properties };
616     }
617 &smShowAssociationInstances ( "CIM_ElementSettingData", $instance.getInstancePath(),
618     #propertylist[] );
619 &smEnd;
```

620 **6.4.2.4 Show Multiple Instances – CIM_DNSGeneralSettingData Reference**

621 This command form is for the show verb applied to multiple instances. This command form corresponds
622 to a show command issued against CIM_ElementSettingData where only one reference is specified and
623 the reference is to an instance of CIM_DNSGeneralSettingData.

624 **6.4.2.4.1 Command Form**

```
625 show <CIM_ElementSettingData multiple instances>
```

626 **6.4.2.4.2 CIM Requirements**

627 See CIM_ElementSettingData in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
628 mandatory properties.

629 **6.4.2.4.3 Behavior Requirements**630 **6.4.2.4.3.1 Preconditions**

631 \$instance contains the instance of CIM_DNSGeneralSettingData which is referenced by
632 CIM_ElementSettingData.

633 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

634 **6.4.2.4.3.2 Pseudo Code**

```
635 #propertylist[] = NULL;
636 if (false == #all)
637     {
638         #propertylist[] = { //all mandatory non-key properties };
639     }
640 &smShowAssociationInstances ( "CIM_ElementSettingData", $instance.getInstancePath(),
641     #propertylist[] );
642 &smEnd;
```

643 **6.4.2.5 Show a Single Instance – CIM_DNSGeneralSettingData and CIM_DNSProtocolEndpoint**
 644 **References**

645 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 646 a `show` command issued against `CIM_ElementSettingData` where both references are specified and
 647 therefore the desired instance is unambiguously identified.

648 **6.4.2.5.1 Command Form**

```
649 show <CIM_ElementSettingData single instance>
```

650 **6.4.2.5.2 CIM Requirements**

651 See `CIM_ElementSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
 652 mandatory properties.

653 **6.4.2.5.3 Behavior Requirements**

654 **6.4.2.5.3.1 Preconditions**

655 `$instanceA` contains the instance of `CIM_DNSProtocolEndpoint` which is referenced by
 656 `CIM_ElementSettingData`.

657 `$instanceB` contains the instance of `CIM_DNSGeneralSettingData` which is referenced by
 658 `CIM_ElementSettingData`.

659 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

660 **6.4.2.5.3.2 Pseudo Code**

```
661 #propertylist[] = NULL;
662 if (false == #all)
663 {
664     #propertylist[] = { //all mandatory non-key properties };
665 }
666 &smShowAssociationInstance ( "CIM_ElementSettingData", $instanceA.getInstancePath(),
667     $instanceB.getInstancePath(), #propertylist[] );
668 &smEnd;
```

669 **6.4.2.6 Show a Single Instance – CIM_DNSGeneralSettingData and CIM_ComputerSystem**
 670 **References**

671 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 672 a `show` command issued against `CIM_ElementSettingData` where both references are specified and
 673 therefore the desired instance is unambiguously identified.

674 **6.4.2.6.1 Command Form**

```
675 show <CIM_ElementSettingData single instance>
```

676 **6.4.2.6.2 CIM Requirements**

677 See `CIM_ElementSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
 678 mandatory properties.

679 **6.4.2.6.3 Behavior Requirements**680 **6.4.2.6.3.1 Preconditions**

681 \$instanceA contains the instance of CIM_ComputerSystem which is referenced by
682 CIM_ElementSettingData.

683 \$instanceB contains the instance of CIM_DNSGeneralSettingData which is referenced by
684 CIM_ElementSettingData.

685 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

686 **6.4.2.6.3.2 Pseudo Code**

```
687 #propertylist[] = NULL;
688 if (false == #all)
689     {
690         #propertylist[] = { //all mandatory non-key properties };
691     }
692 &smShowAssociationInstance ( "CIM_ElementSettingData", $instanceA.getInstancePath(),
693     $instanceB.getInstancePath(), #propertylist[] );
694 &smEnd;
```

695 **6.4.2.7 Show a Single Instance – CIM_DNSSettingData and CIM_DNSProtocolEndpoint Reference**
696

697 This command form is for the `show` verb applied to a single instance. This command form corresponds to
698 a `show` command issued against `CIM_ElementSettingData` where both references are specified and
699 therefore the desired instance is unambiguously identified.

700 **6.4.2.7.1 Command Form**

```
701 show <CIM_ElementSettingData single instance>
```

702 **6.4.2.7.2 CIM Requirements**

703 See `CIM_ElementSettingData` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
704 mandatory properties.

705 **6.4.2.7.3 Behavior Requirements**706 **6.4.2.7.3.1 Preconditions**

707 \$instanceA contains the instance of CIM_DNSProtocolEndpoint which is referenced by
708 CIM_ElementSettingData.

709 \$instanceB contains the instance of CIM_DNSSettingData which is referenced by
710 CIM_ElementSettingData.

711 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

712 **6.4.2.7.3.2 Pseudo Code**

```

713 #propertylist[] = NULL;
714 if (false == #all)
715     {
716         #propertylist[] = { //all mandatory non-key properties };
717     }
718 &smShowAssociationInstance ( "CIM_ElementSettingData", $instanceA.getInstancePath(),
719     $instanceB.getInstancePath(), #propertylist[] );
720 &smEnd;

```

721 **6.5 CIM_SAPSAPDependency**

722 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

723 Table 5 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 724 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 725 verb and target. Table 5 is for informational purposes only; in case of a conflict between Table 5 and
 726 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 727 information in Table 5.

728 **Table 5 – Command Verb Requirements for CIM_SAPSAPDependency**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.5.2.
start	Not supported	
stop	Not supported	

729 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 730 `reset`, `set`, `start`, and `stop`.

731 **6.5.1 Ordering of Results**

732 When results are returned for multiple instances of `CIM_SAPSAPDependency`, implementations shall
 733 utilize the following algorithm to produce the natural (that is, default) ordering:

- 734 • Results for `CIM_SAPSAPDependency` are unordered; therefore, no algorithm is defined.

735 **6.5.2 Show**

736 This section describes how to implement the `show` verb when applied to an instance of
 737 `CIM_SAPSAPDependency`. Implementations shall support the use of the `show` verb with
 738 `CIM_SAPSAPDependency`.

739 The `show` command is used to display information about the CIM_SAPSAPDependency instance or
740 instances.

741 **6.5.2.1 Show a Single Instance – CIM_IPProtocolEndpoint Reference**

742 This command form is for the `show` verb applied to a single instance. This command form corresponds to
743 a `show` command issued against CIM_SAPSAPDependency where only one reference is specified and
744 the reference is to an instance of CIM_IPProtocolEndpoint.

745 **6.5.2.1.1 Command Form**

```
746 show <CIM_SAPSAPDependency single instance>
```

747 **6.5.2.1.2 CIM Requirements**

748 See CIM_SAPSAPDependency in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
749 mandatory properties.

750 **6.5.2.1.3 Behavior Requirements**

751 **6.5.2.1.3.1 Preconditions**

752 `$instance` contains the instance of CIM_IPProtocolEndpoint which is referenced by
753 CIM_SAPSAPDependency.

754 **6.5.2.1.3.2 Pseudo Code**

```
755 &smShowAssociationInstances ( "CIM_SAPSAPDependency", $instance.getInstancePath() );  
756 &smEnd;
```

757 **6.5.2.2 Show a Single Instance – CIM_DNSProtocolEndpoint Reference**

758 This command form is for the `show` verb applied to a single instance. This command form corresponds to
759 a `show` command issued against CIM_SAPSAPDependency where the reference specified is to an
760 instance of CIM_DNSProtocolEndpoint. An instance of CIM_DNSProtocolEndpoint is referenced by
761 exactly one instance of CIM_SAPSAPDependency. Therefore, a single instance will be returned.

762 **6.5.2.2.1 Command Form**

```
763 show <CIM_SAPSAPDependency single instance>
```

764 **6.5.2.2.2 CIM Requirements**

765 See CIM_SAPSAPDependency in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
766 mandatory properties.

767 **6.5.2.2.3 Behavior Requirements**

768 **6.5.2.2.3.1 Preconditions**

769 `$instance` contains the instance of CIM_DNSProtocolEndpoint which is referenced by
770 CIM_SAPSAPDependency.

771 **6.5.2.2.3.2 Pseudo Code**

```
772 &smShowAssociationInstances ( "CIM_SAPSAPDependency", $instance.getInstancePath() );  
773 &smEnd;
```

774 **6.5.2.3 Show a Single Instance – Both References**

775 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 776 a `show` command issued against `CIM_SAPSAPDependency` where both references are specified and
 777 therefore the desired instance is unambiguously identified.

778 **6.5.2.3.1 Command Form**

```
779 show <CIM_SAPSAPDependency single instance>
```

780 **6.5.2.3.2 CIM Requirements**

781 See `CIM_SAPSAPDependency` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
 782 mandatory properties.

783 **6.5.2.3.3 Behavior Requirements**

784 **6.5.2.3.3.1 Preconditions**

785 `$instanceA` contains the instance of `CIM_IPProtocolEndpoint` which is referenced by
 786 `CIM_SAPSAPDependency`.

787 `$instanceB` contains the instance of `CIM_DNSProtocolEndpoint` which is referenced by
 788 `CIM_SAPSAPDependency`.

789 **6.5.2.3.3.2 Pseudo Code**

```
790 &smShowAssociationInstance ( "CIM_SAPSAPDependency", $instanceA.getInstancePath(),  
791     $instanceB.getInstancePath() );  
792 &smEnd;
```

793 **6.6 CIM_HostedAccessPoint**

794 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

795 Table 6 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 796 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 797 verb and target. Table 6 is for informational purposes only; in case of a conflict between Table 6 and
 798 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 799 information in Table 6.

800 **Table 6 – Command Verb Requirements for CIM_HostedAccessPoint**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.6.2.
start	Not supported	
stop	Not supported	

801 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 802 `reset`, `set`, `start`, and `stop`.

803 6.6.1 Ordering of Results

804 When results are returned for multiple instances of CIM_HostedAccessPoint, implementations shall utilize
805 the following algorithm to produce the natural (that is, default) ordering:

- 806 • Results for CIM_HostedAccessPoint are unordered; therefore, no algorithm is defined.

807 6.6.2 Show

808 This section describes how to implement the `show` verb when applied to an instance of
809 CIM_HostedAccessPoint. Implementations shall support the use of the `show` verb with
810 CIM_HostedAccessPoint.

811 The `show` command is used to display information about the CIM_HostedAccessPoint instance or
812 instances.

813 6.6.2.1 Show Multiple Instances – CIM_ComputerSystem

814 This command form is for the `show` verb applied to multiple instances. This command form corresponds
815 to a `show` command issued against CIM_HostedAccessPoint where only one reference is specified and
816 the reference is to an instance of CIM_ComputerSystem.

817 6.6.2.1.1 Command Form

```
818 show <CIM_HostedAccessPoint multiple instances>
```

819 6.6.2.1.2 CIM Requirements

820 See CIM_HostedAccessPoint in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
821 mandatory properties.

822 6.6.2.1.3 Behavior Requirements

823 6.6.2.1.3.1 Preconditions

824 `$instance` contains the instance of CIM_ComputerSystem which is referenced by
825 CIM_HostedAccessPoint.

826 6.6.2.1.3.2 Pseudo Code

```
827 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.getInstancePath() );  
828 &smEnd;
```

829 6.6.2.2 Show a Single Instance – CIM_DNSProtocolEndpoint Reference

830 This command form is for the `show` verb applied to a single instance. This command form corresponds to
831 a `show` command issued against CIM_HostedAccessPoint where the reference specified is to an
832 instance of CIM_DNSProtocolEndpoint. An instance of CIM_DNSProtocolEndpoint is referenced by
833 exactly one instance of CIM_HostedAccessPoint. Therefore, a single instance will be returned.

834 6.6.2.2.1 Command Form

```
835 show <CIM_HostedAccessPoint single instance>
```

836 6.6.2.2.2 CIM Requirements

837 See CIM_HostedAccessPoint in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
838 mandatory properties.

839 6.6.2.2.3 Behavior Requirements

840 6.6.2.2.3.1 Preconditions

841 \$instance contains the instance of CIM_DNSProtocolEndpoint which is referenced by
842 CIM_HostedAccessPoint.

843 6.6.2.2.3.2 Pseudo Code

```
844 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.GetInstancePath() );  
845 &smEnd;
```

846 6.6.2.3 Show a Single Instance – CIM_RemoteServiceAccessPoint Reference

847 This command form is for the `show` verb applied to a single instance. This command form corresponds to
848 a `show` command issued against `CIM_HostedAccessPoint` where the reference specified is to an
849 instance of `CIM_RemoteServiceAccessPoint`. An instance of `CIM_RemoteServiceAccessPoint` is
850 referenced by exactly one instance of `CIM_HostedAccessPoint`. Therefore, a single instance will be
851 returned.

852 6.6.2.3.1 Command Form

```
853 show <CIM_HostedAccessPoint single instance>
```

854 6.6.2.3.2 CIM Requirements

855 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
856 mandatory properties.

857 6.6.2.3.3 Behavior Requirements

858 6.6.2.3.3.1 Preconditions

859 \$instance contains the instance of `CIM_RemoteServiceAccessPoint` which is referenced by
860 `CIM_HostedAccessPoint`.

861 6.6.2.3.3.2 Pseudo Code

```
862 &smShowAssociationInstances ( "CIM_HostedAccessPoint", $instance.GetInstancePath() );  
863 &smEnd;
```

864 6.6.2.4 Show a Single Instance – Both References (CIM_DNSProtocolEndpoint)

865 This command form is for the `show` verb applied to a single instance. This command form corresponds to
866 a `show` command issued against `CIM_HostedAccessPoint` where both references are specified and
867 therefore the desired instance is unambiguously identified.

868 6.6.2.4.1 Command Form

```
869 show <CIM_HostedAccessPoint single instance>
```

870 6.6.2.4.2 CIM Requirements

871 See `CIM_HostedAccessPoint` in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
872 mandatory properties.

873 **6.6.2.4.3 Behavior Requirements**874 **6.6.2.4.3.1 Preconditions**

875 \$instanceA contains the instance of CIM_ComputerSystem which is referenced by
876 CIM_HostedAccessPoint.

877 \$instanceB contains the instance of CIM_DNSProtocolEndpoint which is referenced by
878 CIM_HostedAccessPoint.

879 **6.6.2.4.3.2 Pseudo Code**

```
880 &smShowAssociationInstance ( "CIM_HostedAccessPoint", $instanceA.getInstancePath(),
881     $instanceB.getInstancePath() );
882 &smEnd;
```

883 **6.6.2.5 Show a Single Instance – Both References (CIM_RemoteServiceAccessPoint)**

884 This command form is for the `show` verb applied to a single instance. This command form corresponds to
885 a `show` command issued against CIM_HostedAccessPoint where both references are specified and
886 therefore the desired instance is unambiguously identified.

887 **6.6.2.5.1 Command Form**

```
888 show <CIM_HostedAccessPoint single instance>
```

889 **6.6.2.5.2 CIM Requirements**

890 See CIM_HostedAccessPoint in the “CIM Elements” section of the [DNS Client Profile](#) for the list of
891 mandatory properties.

892 **6.6.2.5.3 Behavior Requirements**893 **6.6.2.5.3.1 Preconditions**

894 \$instanceA contains the instance of CIM_ComputerSystem which is referenced by
895 CIM_HostedAccessPoint.

896 \$instanceB contains the instance of CIM_RemoteServiceAccessPoint which is referenced by
897 CIM_HostedAccessPoint.

898 **6.6.2.5.3.2 Pseudo Code**

```
899 &smShowAssociationInstance ( "CIM_HostedAccessPoint", $instanceA.getInstancePath(),
900     $instanceB.getInstancePath() );
901 &smEnd;
```

902 **6.7 CIM_RemoteAccessAvailableToElement**

903 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

904 Table 7 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
905 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
906 verb and target. Table 7 is for informational purposes only; in case of a conflict between Table 7 and
907 requirements detailed in the following sections, the text detailed in the following sections supersedes the
908 information in Table 7.

909 **Table 7 – Command Verb Requirements for CIM_RemoteAccessAvailableToElement**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.7.2.
start	Not supported	
stop	Not supported	

910 No mappings are defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
911 `reset`, `set`, `start`, and `stop`.

912 **6.7.1 Ordering of Results**

913 When results are returned for multiple instances of `CIM_RemoteAccessAvailableToElement`,
914 implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- 915 • Results for `CIM_RemoteAccessAvailableToElement` are unordered; therefore, no algorithm is
916 defined.

917 **6.7.2 Show**

918 This section describes how to implement the `show` verb when applied to an instance of
919 `CIM_RemoteAccessAvailableToElement`. Implementations shall support the use of the `show` verb with
920 `CIM_RemoteAccessAvailableToElement`.

921 The `show` command is used to display information about the `CIM_RemoteAccessAvailableToElement`
922 instance or instances.

923 **6.7.2.1 Show Multiple Instances – CIM_RemoteServiceAccessPoint Reference**

924 This command form is for the `show` verb applied to multiple instances. This command form corresponds
925 to a `show` command issued against `CIM_RemoteAccessAvailableToElement` where only one reference is
926 specified and the reference is to an instance of `CIM_RemoteServiceAccessPoint`.

927 **6.7.2.1.1 Command Form**

```
928 show <CIM_RemoteAccessAvailableToElement multiple instances>
```

929 **6.7.2.1.2 CIM Requirements**

930 See `CIM_RemoteAccessAvailableToElement` in the “CIM Elements” section of the [DNS Client Profile](#) for
931 the list of mandatory properties.

932 **6.7.2.1.3 Behavior Requirements**933 **6.7.2.1.3.1 Preconditions**

934 `$instance` contains the instance of `CIM_RemoteServiceAccessPoint` which is referenced by
 935 `CIM_RemoteAccessAvailableToElement`.

936 **6.7.2.1.3.2 Pseudo Code**

```
937 &smShowAssociationInstances ( "CIM_RemoteAccessAvailableToElement",
938     $instance.getInstancePath() );
939 &smEnd;
```

940 **6.7.2.2 Show Multiple Instances – CIM_DNSProtocolEndpoint Reference**

941 This command form is for the `show` verb applied to multiple instances. This command form corresponds
 942 to a `show` command issued against `CIM_RemoteAccessAvailableToElement` where the reference
 943 specified is to an instance of `CIM_DNSProtocolEndpoint`. The [DNS Client Profile](#) allows the
 944 implementation to model the DNS servers discovered by the client in addition to the DNS Service that
 945 actually provides the configuration. Therefore, it is possible for there to be multiple
 946 `CIM_RemoteAccessAvailableToElement` associations that reference the `CIM_DNSProtocolEndpoint`
 947 instance.

948 **6.7.2.2.1 Command Form**

```
949 show <CIM_RemoteAccessAvailableToElement single instance>
```

950 **6.7.2.2.2 CIM Requirements**

951 See `CIM_RemoteAccessAvailableToElement` in the “CIM Elements” section of the [DNS Client Profile](#) for
 952 the list of mandatory properties.

953 **6.7.2.2.3 Behavior Requirements**954 **6.7.2.2.3.1 Preconditions**

955 `$instance` contains the instance of `CIM_DNSProtocolEndpoint` which is referenced by
 956 `CIM_RemoteAccessAvailableToElement`.

957 **6.7.2.2.3.2 Pseudo Code**

```
958 &smShowAssociationInstances ( "CIM_RemoteAccessAvailableToElement",
959     $instance.getInstancePath() );
960 &smEnd;
```

961 **6.7.2.3 Show a Single Instance – Both References**

962 This command form is for the `show` verb applied to a single instance. This command form corresponds to
 963 a `show` command issued against `CIM_RemoteAccessAvailableToElement` where both references are
 964 specified and therefore the desired instance is unambiguously identified.

965 **6.7.2.3.1 Command Form**

```
966 show <CIM_RemoteAccessAvailableToElement single instance>
```

967 **6.7.2.3.2 CIM Requirements**

968 See `CIM_RemoteAccessAvailableToElement` in the “CIM Elements” section of the [DNS Client Profile](#) for
 969 the list of mandatory properties.

970 **6.7.2.3.3 Behavior Requirements**

971 **6.7.2.3.3.1 Preconditions**

972 \$instanceA contains the instance of CIM_RemoteServiceAccessPoint which is referenced by
 973 CIM_RemoteAccessAvailableToElement.

974 \$instanceB contains the instance of CIM_DNSProtocolEndpoint which is referenced by
 975 CIM_RemoteAccessAvailableToElement.

976 **6.7.2.3.3.2 Pseudo Code**

```
977 &smShowAssociationInstance ( "CIM_RemoteAccessAvailableToElement",
978     $instanceA.GetInstancePath(), $instanceB.GetInstancePath() );
979 &smEnd;
```

980 **6.8 CIM_RemoteServiceAccessPoint**

981 The `cd` and `help` verbs shall be supported as described in [DSP0216](#).

982 Table 8 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of
 983 the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the
 984 verb and target. Table 8 is for informational purposes only; in case of a conflict between Table 8 and
 985 requirements detailed in the following sections, the text detailed in the following sections supersedes the
 986 information in Table 8.

987 **Table 8 – Command Verb Requirements for CIM_RemoteServiceAccessPoint**

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	Not supported	
show	Shall	See 6.8.2.
start	Not supported	
stop	Not supported	

988 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
 989 `reset`, `start`, and `stop`.

990 **6.8.1 Ordering of Results**

991 When results are returned for multiple instances of CIM_RemoteServiceAccessPoint, implementations
 992 shall utilize the following algorithm to produce the natural (that is, default) ordering:

- 993 • Results for CIM_RemoteServiceAccessPoint are unordered; therefore, no algorithm is defined.

994 **6.8.2 Show**

995 This section describes how to implement the `show` verb when applied to an instance of
 996 `CIM_RemoteServiceAccessPoint`. Implementations shall support the use of the `show` verb with
 997 `CIM_RemoteServiceAccessPoint`.

998 The `show` verb is used to display information about the gateway.

999 **6.8.2.1 Show a Single Instance**

1000 This command form is for the `show` verb applied to a single instance of `CIM_RemoteServiceAccessPoint`.

1001 **6.8.2.1.1 Command Form**

```
1002 show <CIM_RemoteServiceAccessPoint single instance>
```

1003 **6.8.2.1.2 CIM Requirements**

1004 See `CIM_RemoteServiceAccessPoint` in the “CIM Elements” section of the [DNS Client Profile](#) for the list
 1005 of mandatory properties.

1006 **6.8.2.1.3 Behavior Requirements**1007 **6.8.2.1.3.1 Preconditions**

1008 `#all` is true if the “-all” option was specified with the command; otherwise, `#all` is false.

1009 **6.8.2.1.3.2 Pseudo Code**

```
1010 $instance=<CIM_RemoteServiceAccessPoint single instance>  

  1011 #propertylist[] = NULL;  

  1012 if (false == #all)  

  1013 {  

  1014     #propertylist[] = { "AccessContext", "AccessInfo", "InfoFormat", "ElementName" };  

  1015 }  

  1016 &smShowInstance ( $instance.getInstancePath(), #propertylist[] );  

  1017 &smEnd;
```

1018 **6.8.2.2 Show Multiple Instances**

1019 This command form is for the `show` verb applied to multiple instances of
 1020 `CIM_RemoteServiceAccessPoint`. This command form corresponds to UFsT-based selection within a
 1021 scoping system.

1022 **6.8.2.2.1 Command Form**

```
1023 show <CIM_RemoteServiceAccessPoint multiple instances>
```

1024 **6.8.2.2.2 CIM Requirements**

1025 See `CIM_RemoteServiceAccessPoint` in the “CIM Elements” section of the [DNS Client Profile](#) for the list
 1026 of mandatory properties.

1027 6.8.2.2.3 Behavior Requirements**1028 6.8.2.2.3.1 Preconditions**

1029 \$containerInstance contains the instance of CIM_ComputerSystem for which scoped
1030 CIM_RemoteServiceAccessPoint instances are displayed. The [DNS Client Profile](#) requires that the
1031 CIM_RemoteServiceAccessPoint instance be associated with its scoping system via an instance of the
1032 CIM_HostedAccessPoint association.

1033 #all is true if the “-all” option was specified with the command; otherwise, #all is false.

1034 6.8.2.2.3.2 Pseudo Code

```
1035 #propertylist[] = NULL;  
1036 if (false == #all)  
1037     {  
1038         #propertylist[] = { "AccessContext", "AccessInfo", "InfoFormat", "ElementName" };  
1039     }  
1040 &smShowInstances ( "CIM_RemoteServiceAccessPoint", "CIM_HostedAccessPoint",  
1041     $containerInstance.getInstancePath(), #propertylist[] );  
1042 &smEnd;
```

1043

ANNEX A
(informative)**Change Log**

Version	Date	Author	Description
1.0.0	2009-07-14		DMTF Standard Release

1044
1045
1046
1047
1048

1049