Document Number: DSP0821

Date: 2009-07-14

Version: 1.0.0

SSH Service Profile SM CLP Command Mapping Specification

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Tables

75

Foreword

The *SSH Service Profile SM CLP Command Mapping Specification* (DSP0821) was prepared by the Server Management Working Group.

Conventions

The pseudo-code conventions utilized in this document are the Recipe Conventions as defined in SNIA SMI-S 1.1.0, section 7.6.

Acknowledgements

The authors wish to acknowledge the following participants from the DMTF Server Management Working Group:

- Aaron Merkin – IBM
- Jon Hass – Dell
- Khachatur Papanyan – Dell
- Jeff Hilland – HP
- Christina Shaw – HP
- Perry Vincent – Intel
- John Leung – Intel

Introduction

This document defines the SM CLP mapping for CIM elements described in the *SSH Service Profile*. The information in this specification, combined with *SM CLP-to-CIM Common Mapping Specification 1.0*, is intended to be sufficient to implement SM CLP commands relevant to the classes, properties and methods described in the *SSH Service Profile* using CIM operations.

The target audience for this specification is implementers of the SM CLP support for the *SSH Service Profile*.

SSH Service Profile SM CLP Command Mapping Specification

1 Scope

This specification contains the requirements for an implementation of the SM CLP to provide access to, and implement the behaviors of, the *SSH Service Profile*.

2 Normative References

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

2.1 Approved References

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

DMTF DSP1006, *SMASH Collections Profile1.0,*
http://www.dmtf.org/standards/published_documents/DSP1006_1.0.pdf

DMTF DSP1017, *SSH Service Profile 1.0*,
http://www.dmtf.org/standards/published_documents/DSP1017_1.0.pdf

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

2.2 Other References

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

3 Terms and Definitions

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

3.1
can
used for statements of possibility and capability, whether material, physical, or causal

3.2
cannot
used for statements of possibility and capability, whether material, physical or causal

3.3
conditional
indicates requirements to be followed strictly in order to conform to the document when the specified conditions are met

3.4
mandatory
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.5
may
indicates a course of action permissible within the limits of the document

3.6
need not
indicates a course of action permissible within the limits of the document

3.7
optional
indicates a course of action permissible within the limits of the document

3.8
shall
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.9
shall not
indicates requirements to be followed strictly in order to conform to the document and from which no deviation is permitted

3.10
should
indicates that among several possibilities, one is recommended as particularly suitable, without mentioning or excluding others, or that a certain course of action is preferred but not necessarily required

3.11
should not
indicates that a certain possibility or course of action is deprecated but not prohibited

4 Symbols and Abbreviated Terms

The following symbols and abbreviations are used in this document.

4.1
CIM
Common Information Model

4.2
CLP
Common Information Model

4.3
DMTF
Distributed Management Task Force

172 **4.4**
173 **IETF**
174 Internet Engineering Task Force

175 **4.5**
176 **SM**
177 Server Management

178 **4.6**
179 **SMI-S**
180 Storage Management Initiative Specification

181 **4.7**
182 **SNIA**
183 Storage Networking Industry Association

184 **4.8**
185 **UFsT**
186 User Friendly selection Tag

187 5 Recipes

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

190 • smStartRSC()
191 • smStopRSC()
192 • smResetRSC()
193 • smShowInstance()
194 • smShowInstances()
195 • smSetInstance()
196 • smShowAssociationInstances()
197 • smShowAssociationInstance()
198 • smDeleteInstance
199 • smMakeCommandStatus
200 • smNewInstance

201 For convenience, Table 1 lists each recipe defined in this mapping which is used for more than one verb
202 or class mapping.

203 **Table 1 – Local Recipes**

Recipe Name	Description	Definition
lShowTCPEndpoint	Show an instance of CIM_TCPProtocolEndpoint	See 5.1.

204 The following sections detail Local Recipes defined for use in this mapping.

5.1 lShowTCPEndpoint

5.1.1 Description

Reusable recipe for displaying an instance of CIM_TCPProtocolEndpoint. A recipe is defined for re-use by the `show` and `create` verbs applied to CIM_TCPProtocolEndpoint.

5.1.2 Preconditions

`$endpoint` contains the instance of CIM_TCPProtocolEndpoint to display.

`#all` indicates whether the "-all" option was specified.

5.1.3 Pseudo Code

```
sub lShowTCPEndpoint($endpoint, #all)
   {
   #propertylist[] = NULL;
   //if we're not displaying all of the properties, provide a list
   if (false == #all)
      {
      #propertylist[] = { //all mandatory non-key properties };
      }
   &smShowInstance ( $endpoint.GetObjectPath(), #propertyList[] );
   &smEnd;
} //lShowTCPEndpoint()
```

6 Mappings

The following sections detail the mapping of CLP verbs to CIM Operations for each CIM class defined in the *SSH Service Profile*. Requirements specified here related to support for a CLP verb for a particular class are solely within the context of this profile.

6.1 CIM_BindsTo

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

Table 2 – Command Verb Requirements for CIM_BindsTo

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 6.1.2.
start	Not supported	
stop	Not supported	

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

6.1.1 Ordering of Results

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

- Results for CIM_BindsTo are unordered; therefore, no algorithm is defined.

6.1.2 Show

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

The `show` command is used to display information about the CIM_BindsTo instance or instances.

6.1.2.1 Show Multiple Instances – CIM_IPProtocolEndpoint

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_BindsTo where only one reference is specified and the reference is to an instance of CIM_IPProtocolEndpoint.

6.1.2.1.1 Command Form

```
show <CIM_BindsTo multiple instances>
```

6.1.2.1.2 CIM Requirements

See CIM_BindsTo in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.1.2.1.3 Behavior Requirements

6.1.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_IPProtcolEndpoint which is referenced by CIM_BindsTo.

6.1.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_BindsTo”, $instance.getObjectPath() );
&smEnd;
```

6.1.2.2 Show Multiple Instances – CIM_TCPProtocolEndpoint

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_BindsTo where only one reference is specified and the reference is to an instance of CIM_TCPProtocolEndpoint.

6.1.2.2.1 Command Form

```
show <CIM_BindsTo multiple instances>
```

6.1.2.2.2 CIM Requirements

See CIM_BindsTo in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.1.2.2.3 Behavior Requirements

6.1.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_TCPProtcolEndpoint which is referenced by CIM_BindsTo.

6.1.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_BindsTo”, $instance.getObjectPath() );
&smEnd;
```

6.1.2.3 Show a Single Instance – CIM_SSHProtocolEndpoint Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_BindsTo where the reference specified is to an instance of CIM_SSHProtocolEndpoint. A single instance of CIM_TCPProtocolEndpoint is associated with each instance of CIM_SSHProtocolEndpoint. Therefore, a single instance will be returned.

6.1.2.3.1 Command Form

```
show <CIM_BindsTo single instance>
```

6.1.2.3.2 CIM Requirements

See CIM_BindsTo in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.1.2.3.3 Behavior Requirements

6.1.2.3.3.1 Preconditions

`$instance` contains the instance of CIM_SSHProtocolEndpoint which is referenced by CIM_BindsTo.

6.1.2.3.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_BindsTo”, $instance.getObjectPath() );
&smEnd;
```

6.1.2.4 Show a Single Instance – Both References A

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_BindsTo where a reference to CIM_SSHProtocolEndpoint and a reference to CIM_TCPProtocolEndpoint are specified and therefore the desired instance is unambiguously identified.

6.1.2.4.1 Command Form

```
show <CIM_BindsTo single instance>
```

6.1.2.4.2 CIM Requirements

See CIM_BindsTo in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.1.2.4.3 Behavior Requirements

6.1.2.4.3.1 Preconditions

`$instanceA` contains the instance of CIM_SSHProtocolEndpoint which is referenced by CIM_BindsTo.

`$instanceB` contains the instance of CIM_TCPProtocolEndpoint which is referenced by CIM_BindsTo.

6.1.2.4.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_BindsTo”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.1.2.5 Show a Single Instance – Both References B

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_BindsTo where a reference to CIM_IPProtocolEndpoint and a reference to CIM_TCPProtocolEndpoint are specified and therefore the desired instance is unambiguously identified.

6.1.2.5.1 Command Form

```
show <CIM_BindsTo single instance>
```

6.1.2.5.2 CIM Requirements

See CIM_BindsTo in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.1.2.5.3 Behavior Requirements

6.1.2.5.3.1 Preconditions

`$instanceA` contains the instance of CIM_IPProtocolEndpoint which is referenced by CIM_BindsTo.

`$instanceB` contains the instance of CIM_TCPProtocolEndpoint which is referenced by CIM_BindsTo.

6.1.2.5.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_BindsTo”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.2 CIM_ElementCapabilities

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 3 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 3 is for informational purposes only; in case of a conflict between Table 3 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 3.

335 **Table 3 – Command Verb Requirements for CIM_ElementCapabilities**

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 6.2.2.
start	Not supported	
stop	Not supported	

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

338 ### 6.2.1 Ordering of Results

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

341 - Results for CIM_ElementCapabilities are unordered; therefore, no algorithm is defined.

342 ### 6.2.2 Show

343 This section describes how to implement the `show` verb when applied to an instance of
344 CIM_ElementCapabilities. Implementations shall support the use of the `show` verb with
345 CIM_ElementCapabilities.

346 The `show` command is used to display information about the CIM_ElementCapabilities instance or
347 instances.

348 #### 6.2.2.1 Show a Single Instance – CIM_SSHCapabilities Reference

349 This command form is for the `show` verb applied to a single instance. This command form corresponds to
350 a `show` command issued against CIM_ElementCapabilities where the reference specified is to an
351 instance of CIM_SSHCapabilities. A single instance of CIM_ProtocolService is associated with each
352 instance of a CIM_SSHCapabilities. Therefore, a single instance will be returned.

353 ##### 6.2.2.1.1 Command Form

354 **`show <CIM_ElementCapabilities` *`single instance`*`>`**

355 ##### 6.2.2.1.2 CIM Requirements

356 See CIM_ElementCapabilities in the “CIM Elements” section of the *SSH Service Profile* for the list of
357 mandatory properties.

6.2.2.1.3 Behavior Requirements

6.2.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_SSHCapabilities which is referenced by CIM_ElementCapabilities.

6.2.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ElementCapabilities”, $instance.getObjectPath() );
&smEnd;
```

6.2.2.2 Show a Single Instance – CIM_ProtocolService Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ElementCapabilities where the reference specified is to an instance of CIM_ProtocolService. A single instance of CIM_SSHCapabilities is associated with each instance of CIM_ProtocolService. Therefore, a single instance will be returned.

6.2.2.2.1 Command Form

```
show <CIM_ElementCapabilities single instance>
```

6.2.2.2.2 CIM Requirements

See CIM_ElementCapabilities in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.2.2.2.3 Behavior Requirements

6.2.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_ProtocolService which is referenced by CIM_ElementCapabilities.

6.2.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ElementCapabilities”, $instance.getObjectPath() );
&smEnd;
```

6.2.2.3 Show a Single Instance – Both References

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

6.2.2.3.1 Command Form

```
show <CIM_ElementCapabilities single instance>
```

6.2.2.3.2 CIM Requirements

See CIM_ElementCapabilities in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.2.2.3.3 Behavior Requirements

6.2.2.3.3.1 Preconditions

`$instanceA` contains the instance of CIM_SSHCapabilities which is referenced by CIM_ElementCapabilities.

`$instanceB` contains the instance of CIM_ProtocolService which is referenced by CIM_ElementCapabilities.

6.2.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_ElementCapabilities”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.3 CIM_ElementSettingData

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

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	
set	May	See 6.3.2.
show	Shall	See 6.3.3.
start	Not supported	
stop	Not supported	

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

6.3.1 Ordering of Results

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

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

6.3.2 Set

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

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

6.3.2.1 Set of IsNext

The IsNext property is the only property of CIM_ElementSettingData which can be modified directly via the `set` verb.

6.3.2.1.1 Command Form

```
set <CIM_ElementSettingData single instance> IsNext=<propertyvalue>
```

6.3.2.1.2 CIM Requirements

See CIM_ElementSettingData in the “CIM Elements” section of the *SSH Service Profile* for the CIM_ElementSettingData.IsNext property.

6.3.2.1.3 Behavior Requirements

```
$instance=<CIM_ElementSettingData single instance>
#propertyNames[] = { “IsNext” };
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.3.3 Show

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

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

6.3.3.1 Show Multiple Instances – CIM_SSHSettingData and CIM_SSHProtocolEndpoint

This command form corresponds to a `show` command issued against CIM_ElementSettingData where the reference specified is to an instance of CIM_SSHSettingData. Note that when an instance of CIM_SSHSettingData is associated with an instance of CIM_SSHProtocolEndpoint, the IsCurrent property is the mandatory property.

6.3.3.1.1 Command Form

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

6.3.3.1.2 CIM Requirements

See CIM_ElementSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.3.3.1.3 Behavior Requirements

6.3.3.1.3.1 Preconditions

`$instance` contains the instance of CIM_SSHSettingData which is referenced by CIM_ElementSettingData.

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

6.3.3.1.3.2 Pseudo Code

```
#propertylist = NULL;
if (false == #all)
   {
   #propertylist = { “IsCurrent” };
   }
&smShowAssociationInstances ( “CIM_ElementSettingData”, $instance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.3.3.2 Show Multiple Instances – CIM_SSHProtocolEndpoint Reference

This command form corresponds to a `show` command issued against CIM_ElementSettingData where the reference specified is to an instance of CIM_SSHProtocolEndpoint. Note that when an instance of CIM_SSHSettingData is associated with an instance of CIM_SSHProtocolEndpoint, the IsCurrent property is the mandatory property.

6.3.3.2.1 Command Form

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

6.3.3.2.2 CIM Requirements

See CIM_ElementSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.3.3.2.3 Behavior Requirements

6.3.3.2.3.1 Preconditions

`$instance` contains the instance of CIM_SSHProtocolEndpoint which is referenced by CIM_ElementSettingData.

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

6.3.3.2.3.2 Pseudo Code

```
#propertylist = NULL;
if (false == #all)
   {
   #propertylist = { “IsCurrent” };
   }
&smShowAssociationInstances ( “CIM_ElementSettingData”, $instance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.3.3.3 Show a Single Instance – CIM_SSHSettingData and CIM_SSHProtocolEndpoint

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

6.3.3.3.1 Command Form

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

6.3.3.3.2 CIM Requirements

See CIM_ElementSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.3.3.3.3 Behavior Requirements

6.3.3.3.3.1 Preconditions

`$instanceA` contains the instance of CIM_SSHSettingData which is referenced by CIM_ElementSettingData.

`$instanceB` contains the instance of CIM_SSHProtocolEndpoint which is referenced by CIM_ElementSettingData.

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

6.3.3.3.3.2 Pseudo Code

```
#propertylist = NULL;
if (false == #all)
   {
   #propertylist = { “IsCurrent” };
   }
&smShowAssociationInstance ( “CIM_ElementSettingData”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath(), #propertylist[] );
&smEnd;
```

6.3.3.4 Show Multiple Instances – CIM_SSHSettingData and CIM_ProtocolService

This command form corresponds to a `show` command issued against CIM_ElementSettingData where the reference specified is to an instance of CIM_SSHSettingData. Note that when an instance of CIM_SSHSettingData is associated with an instance of CIM_ProtocolService, the IsNext and IsDefault properties are mandatory.

6.3.3.4.1 Command Form

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

6.3.3.4.2 CIM Requirements

See CIM_ElementSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.3.3.4.3 Behavior Requirements

6.3.3.4.3.1 Preconditions

`$instance` contains the instance of CIM_SSHSettingData which is referenced by CIM_ElementSettingData.

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

6.3.3.4.3.2 Pseudo Code

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist = { “IsNext”, “IsDefault” };
   }
&smShowAssociationInstances ( “CIM_ElementSettingData”, $instance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.3.3.5 Show Multiple Instances – CIM_ProtocolService Reference

This command form corresponds to a `show` command issued against CIM_ElementSettingData where the reference specified is to an instance of CIM_ProtocolService. Note that when an instance of CIM_SSHSettingData is associated with an instance of CIM_ProtocolService, the IsNext and IsDefault properties are mandatory.

6.3.3.5.1 Command Form

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

6.3.3.5.2 CIM Requirements

See CIM_ElementSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.3.3.5.3 Behavior Requirements

6.3.3.5.3.1 Preconditions

`$instance` contains the instance of CIM_ProtocolService which is referenced by CIM_ElementSettingData.

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

6.3.3.5.3.2 Pseudo Code

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist = { “IsNext”, “IsDefault” };
   }
&smShowAssociationInstances ( “CIM_ElementSettingData”, $instance.getObjectPath(),
   #propertylist[] );
&smEnd;
```

6.3.3.6 Show a Single Instance – SettingData and ProtocolService

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

6.3.3.6.1 Command Form

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

6.3.3.6.2 CIM Requirements

See CIM_ElementSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.3.3.6.3 Behavior Requirements

6.3.3.6.3.1 Preconditions

`$instanceA` contains the instance of CIM_SSHSettingData which is referenced by CIM_ElementSettingData.

`$instanceB` contains the instance of CIM_ProtocolService which is referenced by CIM_ElementSettingData.

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

6.3.3.6.3.2 Pseudo Code

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist = { “IsNext”, “IsDefault” };
   }
&smShowAssociationInstance ( “CIM_ElementSettingData”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath(), #propertylist[] );
&smEnd;
```

6.4 CIM_HostedAccessPoint

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

593 **Table 5 – 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 6.4.2.
start	Not supported	
stop	Not supported	

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

596 ### 6.4.1 Ordering of Results

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

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

600 ### 6.4.2 Show

601 This section describes how to implement the `show` verb when applied to an instance of
602 CIM_HostedAccessPoint. Implementations shall support the use of the `show` verb with
603 CIM_HostedAccessPoint.

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

606 #### 6.4.2.1 Show Multiple Instances

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

610 ##### 6.4.2.1.1 Command Form

611 ```
show <CIM_HostedAccessPoint multiple instances>
```

612 ##### 6.4.2.1.2 CIM Requirements

613 See CIM_HostedAccessPoint in the “CIM Elements” section of the *SSH Service Profile* for the list of
614 mandatory properties.

615 ##### 6.4.2.1.3 Behavior Requirements

616 ###### 6.4.2.1.3.1 Preconditions

617 `$instance` contains the instance of CIM_ComputerSystem which is referenced by
618 CIM_HostedAccessPoint.
```

6.4.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_HostedAccessPoint”, $instance.getObjectPath() );
&smEnd;
```

6.4.2.2 Show a Single Instance – CIM_TCPProtocolEndpoint or CIM_SSHProtocolEndpoint Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_HostedAccessPoint where the reference specified is to an instance of CIM_TCPProtocolEndpoint or CIM_SSHProtocolEndpoint. A single instance will be returned.

6.4.2.2.1 Command Form

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

6.4.2.2.2 CIM Requirements

See CIM_HostedAccessPoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.4.2.2.3 Behavior Requirements

6.4.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_TCPProtocolEndpoint or CIM_SSHProtocolEndpoint which is referenced by CIM_HostedAccessPoint.

6.4.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_HostedAccessPoint”, $instance.getObjectPath() );
&smEnd;
```

6.4.2.3 Show a Single Instance – Both References

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

6.4.2.3.1 Command Form

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

6.4.2.3.2 CIM Requirements

See CIM_HostedAccessPoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.4.2.3.3 Behavior Requirements

6.4.2.3.3.1 Preconditions

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

`$instanceB` contains the instance of CIM_SSHProtocolEndpoint or CIM_TCPProtocolEndpoint which is referenced by CIM_HostedAccessPoint.

6.4.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_HostedAccessPoint”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.5 CIM_HostedService

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

Table 6 – Command Verb Requirements for CIM_HostedService

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	

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

6.5.1 Ordering of Results

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

- Results for CIM_HostedService are unordered; therefore, no algorithm is defined.

6.5.2 Show

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

The `show` command is used to display information about the CIM_HostedService instance or instances.

6.5.2.1 Show Multiple Instances

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

680 **6.5.2.1.1 Command Form**

681 ```
show <CIM_HostedService multiple instances>
```

682 **6.5.2.1.2 CIM Requirements**

683 See CIM_HostedService in the “CIM Elements” section of the *SSH Service Profile* for the list of
684 mandatory properties.

685 **6.5.2.1.3 Behavior Requirements**

686 **6.5.2.1.3.1 Preconditions**

687 `$instance` contains the instance of CIM_ComputerSystem which is referenced by CIM_HostedService.

688 **6.5.2.1.3.2 Pseudo Code**

689 ```
&smShowAssociationInstances ( “CIM_HostedService”, $instance.getObjectPath() );
690 &smEnd;
```

691 **6.5.2.2 Show a Single Instance – CIM_ProtocolService Reference**

692 This command form is for the `show` verb applied to a single instance. This command form corresponds to
693 a `show` command issued against CIM_HostedService where the reference specified is to an instance of
694 CIM_ProtocolService. An instance of CIM_ProtocolService is referenced by exactly one instance of
695 CIM_HostedService. Therefore, a single instance will be returned.

696 **6.5.2.2.1 Command Form**

697 ```
show <CIM_HostedService single instance>
```

698 **6.5.2.2.2 CIM Requirements**

699 See CIM_HostedService in the “CIM Elements” section of the *SSH Service Profile* for the list of
700 mandatory properties.

701 **6.5.2.2.3 Behavior Requirements**

702 **6.5.2.2.3.1 Preconditions**

703 `$instance` contains the instance of CIM_ProtocolService which is referenced by CIM_HostedService.

704 **6.5.2.2.3.2 Pseudo Code**

705 ```
&smShowAssociationInstances ( “CIM_HostedService”, $instance.getObjectPath() );
706 &smEnd;
```

707 **6.5.2.3 Show a Single Instance – Both References**

708 This command form is for the `show` verb applied to a single instance. This command form corresponds to
709 a `show` command issued against CIM_HostedService where both references are specified and therefore
710 the desired instance is unambiguously identified.

711 **6.5.2.3.1 Command Form**

712 ```
show <CIM_HostedService single instance>
```
```

6.5.2.3.2 CIM Requirements

See CIM_HostedService in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.5.2.3.3 Behavior Requirements

6.5.2.3.3.1 Preconditions

`$instanceA` contains the instance of CIM_ComputerSystem which is referenced by CIM_HostedService.

`$instanceB` contains the instance of CIM_ProtocolService which is referenced by CIM_HostedService.

6.5.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_HostedService”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.6 CIM_ProvidesEndpoint

The `cd` and `help` verbs shall be supported as described in DSP0216.

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

Table 7 – Command Verb Requirements for CIM_ProvidesEndpoint

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	

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

6.6.1 Ordering of Results

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

- Results for CIM_ProvidesEndpoint are unordered; therefore, no algorithm is defined.

6.6.2 Show

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

The `show` command is used to display information about the CIM_ProvidesEndpoint instance or instances.

6.6.2.1 Show a Single Instance – CIM_SSHProtocolEndpoint Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ProvidesEndpoint where the reference specified is to an instance of CIM_SSHProtocolEndpoint. A single instance of CIM_ProtocolService is associated with each instance of a CIM_SSHProtocolEndpoint. Therefore, a single instance will be returned.

6.6.2.1.1 Command Form

```
show <CIM_ProvidesEndpoint single instance>
```

6.6.2.1.2 CIM Requirements

See CIM_ProvidesEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.6.2.1.3 Behavior Requirements

6.6.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_SSHProtocolEndpoint which is referenced by CIM_ProvidesEndpoint.

6.6.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ProvidesEndpoint”, $instance.getObjectPath() );
&smEnd;
```

6.6.2.2 Show Multiple Instances – CIM_ProtocolService Reference

This command form is for the `show` verb applied to a single instance. This command form corresponds to a `show` command issued against CIM_ProvidesEndpoint where the reference specified is to an instance of CIM_ProtocolService. A single instance of CIM_ProtocolService is associated with multiple instances of a CIM_SSHProtocolEndpoint. Therefore, multiple instances may be returned.

6.6.2.2.1 Command Form

```
show <CIM_ProvidesEndpoint multiple instances>
```

6.6.2.2.2 CIM Requirements

See CIM_ProvidesEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.6.2.2.3 Behavior Requirements

6.6.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_ProtocolService which is referenced by CIM_ProvidesEndpoint.

6.6.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ProvidesEndpoint”, $instance.getObjectPath() );
&smEnd;
```

6.6.2.3 Show a Single Instance – Both References

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

6.6.2.3.1 Command Form

```
show <CIM_ProvidesEndpoint single instance>
```

6.6.2.3.2 CIM Requirements

See CIM_ProvidesEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.6.2.3.3 Behavior Requirements

6.6.2.3.3.1 Preconditions

`$instanceA` contains the instance of CIM_SSHProtocolEndpoint which is referenced by CIM_ProvidesEndpoint.

`$instanceB` contains the instance of CIM_ProtocolService which is referenced by CIM_ProvidesEndpoint.

6.6.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_ProvidesEndpoint”,$instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.7 CIM_ProtocolService

The cd and help verbs shall be supported as described in DSP0216.

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

Table 8 – Command Verb Requirements for CIM_ProtocolService

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	May	See 6.7.2.
set	May	See 6.7.3.

Command Verb	Requirement	Comments
show	Shall	See 6.7.4.
start	May	See 6.7.5.
stop	May	See 6.7.6.

No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, and `load`.

6.7.1 Ordering of Results

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

- Results for CIM_ProtocolService are unordered; therefore, no algorithm is defined.

6.7.2 Reset

This section describes how to implement the `reset` verb when applied to an instance of CIM_ProtocolService. Implementations may support the use of the `reset` verb with CIM_ProtocolService.

The `reset` verb is used to initiate a reset of the CIM_ProtocolService.

6.7.2.1 Reset a Single Instance

This command form is for the initiation of a reset action against a single instance of the CIM_ProtocolService. The mapping is implemented as an invocation of the RequestStateChange( ) method on the instance.

6.7.2.1.1 Command Form

reset <CIM_ProtocolService *single instance*>

6.7.2.1.2 CIM Requirements

```
uint16 EnabledState;
uint16 RequestedState;
uint32 EnabledLogicalElement.RequestStateChange (
   [IN] uint16 RequestedState,
   [OUT] REF CIM_ConcreteJob Job,
   [IN] datetime TimeoutPeriod );
```

6.7.2.1.3 Behavior Requirements

```
$instance=<CIM_ProtocolService single instance>
&smResetRSC ( $instance.getObjectPath() );
&smEnd;
```

6.7.3 Set

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

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

6.7.3.1 General Usage of Set for a Single Property

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *SSH Service Profile*.

6.7.3.1.1 Command Form

```
set <CIM_ProtocolService single instance> <propertyname>=<propertyvalue>
```

6.7.3.1.2 CIM Requirements

See CIM_ProtocolService in the “CIM Elements” section of the *SSH Service Profile* for the list of modifiable properties.

6.7.3.1.3 Behavior Requirements

```
$instance=<CIM_ProtocolService single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.7.3.2 General Usage of Set for Multiple Properties

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *SSH Service Profile*.

6.7.3.2.1 Command Form

```
set <CIM_ProtocolService single instance> <propertyname1>=<propertyvalue1>
   <propertynamen>=<propertyvaluen>
```

6.7.3.2.2 CIM Requirements

See CIM_ProtocolService in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.7.3.2.3 Behavior Requirements

```
$instance=<CIM_ProtocolService single instance>
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.7.4 Show

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

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

6.7.4.1 Show a Single Instance

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

6.7.4.1.1 Command Form

```
show <CIM_ProtocolService single instance>
```

6.7.4.1.2 CIM Requirements

See CIM_ProtocolService in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.7.4.1.3 Behavior Requirements

6.7.4.1.3.1 Preconditions

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

6.7.4.1.3.2 Pseudo Code

```
$instance=<CIM_ProtocolService single instance>
#propertylist[] = NULL;
if (false == #all)
   {
   propertylist[] = { //all mandatory non-key properties };
   }
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.7.4.2 Show Multiple Instances

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

6.7.4.2.1 Command Form

```
show <CIM_ProtocolService multiple instances>
```

6.7.4.2.2 CIM Requirements

See CIM_ProtocolService in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.7.4.2.3 Behavior Requirements

6.7.4.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ComputerSystem for which we are displaying scoped instances of the CIM_ProtocolService. The *SSH Service Profile* requires that the

CIM_ProtocolService instance be associated with its scoping system via an instance of the CIM_HostedService association.

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

6.7.4.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = { //all mandatory non-key properties };
   }
&smShowInstances ( "CIM_ProtocolService", "CIM_HostedService",
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.7.5 Start

This section describes how to implement the `start` verb when applied to an instance of CIM_ProtocolService. Implementations may support the use of the `start` verb with CIM_ProtocolService.

The `start` verb is used to enable the CIM_ProtocolService.

6.7.5.1 Start a Single Instance

This command form is for the `start` verb applied to a single instance of CIM_ProtocolService.

6.7.5.1.1 Command Form

```
start <CIM_ProtocolService single instance>
```

6.7.5.1.2 CIM Requirements

```
uint16 EnabledState;
uint16 RequestedState;
uint32 EnabledLogicalElement.RequestStateChange (
   [IN] uint16 RequestedState,
   [OUT] REF CIM_ConcreteJob Job,
   [IN] datetime TimeoutPeriod );
```

6.7.5.1.3 Behavior Requirements

```
$instance=<CIM_ProtocolService single instance>
&smStartRSC ( $instance.getObjectPath() );
&smEnd;
```

6.7.6 Stop

This section describes how to implement the `stop` verb when applied to an instance of CIM_ProtocolService. Implementations may support the use of the `stop` verb with CIM_ProtocolService.

The `stop` verb is used to disable the CIM_ProtocolService.

6.7.6.1 Stop a Single Instance

This command form is for the `stop` verb applied to a single instance of CIM_ProtocolService.

6.7.6.1.1 Command Form

```
stop <CIM_ProtocolService single instance>
```

6.7.6.1.2 CIM Requirements

```
uint16 EnabledState;
uint16 RequestedState;
uint32 EnabledLogicalElement.RequestStateChange (
   [IN] uint16 RequestedState,
   [OUT] REF CIM_ConcreteJob Job,
   [IN] datetime TimeoutPeriod );
```

6.7.6.1.3 Behavior Requirements

```
$instance=<CIM_ProtocolService single instance>
&smStopRSC ( $instance.getObjectPath() );
&smEnd;
```

6.8 CIM_ServiceAccessBySAP

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 9 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 9 is for informational purposes only; in case of a conflict between Table 9 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 9.

Table 9 – Command Verb Requirements for CIM_ServiceAccessBySAP

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	

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

6.8.1 Ordering of Results

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

- Results for CIM_ServiceAccessBySAP are unordered; therefore, no algorithm is defined.

6.8.2 Show

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

The `show` command is used to display information about the CIM_ServiceAccessBySAP instance or instances.

6.8.2.1 Show Multiple Instances – CIM_ProtocolService Reference

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_ServiceAccessBySAP where only one reference is specified and the reference is to an instance of CIM_ProtocolService.

6.8.2.1.1 Command Form

```
show <CIM_ServiceAccessBySAP multiple instances>
```

6.8.2.1.2 CIM Requirements

See CIM_ServiceAccessBySAP in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.8.2.1.3 Behavior Requirements

6.8.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_ProtocolService which is referenced by CIM_ServiceAccessBySAP.

6.8.2.1.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ServiceAccessBySAP”, $instance.getObjectPath() );
&smEnd;
```

6.8.2.2 Show Multiple Instances – CIM_TCPProtocolEndpoint Reference

This command form is for the `show` verb applied to multiple instances. This command form corresponds to a `show` command issued against CIM_ServiceAccessBySAP where the reference specified is to an instance of CIM_TCPProtocolEndpoint.

6.8.2.2.1 Command Form

```
show <CIM_ServiceAccessBySAP multiple instances>
```

6.8.2.2.2 CIM Requirements

See CIM_ServiceAccessBySAP in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.8.2.2.3 Behavior Requirements

6.8.2.2.3.1 Preconditions

`$instance` contains the instance of CIM_TCPProtocolEndpoint which is referenced by CIM_ServiceAccessBySAP.

6.8.2.2.3.2 Pseudo Code

```
&smShowAssociationInstances ( “CIM_ServiceAccessBySAP”, $instance.getObjectPath() );
&smEnd;
```

6.8.2.3 Show a Single Instance – Both References

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

6.8.2.3.1 Command Form

```
show <CIM_ServiceAccessBySAP single instance>
```

6.8.2.3.2 CIM Requirements

See CIM_ServiceAccessBySAP in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.8.2.3.3 Behavior Requirements

6.8.2.3.3.1 Preconditions

`$instanceA` contains the instance of CIM_TCPProtocolEndpoint which is referenced by CIM_ServiceAccessBySAP.

`$instanceB` contains the instance of CIM_ProtocolService which is referenced by CIM_ServiceAccessBySAP.

6.8.2.3.3.2 Pseudo Code

```
&smShowAssociationInstance ( “CIM_ServiceAccessBySAP”, $instanceA.getObjectPath(),
   $instanceB.getObjectPath() );
&smEnd;
```

6.9 CIM_SSHCapabilities

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 10 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 10 is for informational purposes only; in case of a conflict between Table 10 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 10.

Table 10 – Command Verb Requirements for CIM_SSHCapabilities

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

Command Verb	Requirement	Comments
show	Shall	See 6.9.2.
start	Not supported	
stop	Not supported	

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

6.9.1 Ordering of Results

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

- Results for CIM_SSHCapabilities are unordered; therefore, no algorithm is defined.

6.9.2 Show

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

The `show` verb is used to display information about an instance or instances of the CIM_SSHCapabilities class.

6.9.2.1 Show a Single Instance

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

6.9.2.1.1 Command Form

```
show <CIM_SSHCapabilities single instance>
```

6.9.2.1.2 CIM Requirements

See CIM_SSHCapabilities in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.9.2.1.3 Behavior Requirements

6.9.2.1.3.1 Preconditions

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

6.9.2.1.3.2 Pseudo Code

```
$instance=<CIM_SSHCapabilities single instance>
#propertylist[] = NULL;
if ( false == #all)
   {
   #propertylist[] = {//all mandatory non-key properties }
   }
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.9.2.2 Show Multiple Instances

This command form is for the `show` verb applied to multiple instances of CIM_SSHCapabilities. This command form corresponds to UFsT-based selection within a capabilities collection.

6.9.2.2.1 Command Form

```
show <CIM_SSHCapabilities multiple instances>
```

6.9.2.2.2 CIM Requirements

See CIM_SSHCapabilities in the "CIM Elements" section of the *SSH Service Profile* for the list of mandatory properties.

6.9.2.2.3 Behavior Requirements

6.9.2.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ConcreteCollection for which contained CIM_Capabilities instances are displayed. CIM_Capabilities instances are addressed via an aggregating instance of CIM_ConcreteCollection.

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

6.9.2.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if ( false == #all)
   {
   #propertylist[] = {//all mandatory non-key properties }
   }
&smShowInstances ( "CIM_SSHCapabilities", "CIM_MemberOfCollection",
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.10 CIM_SSHProtocolEndpoint

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 11 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 11 is for informational purposes only; in case of a conflict between Table 11 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 11.

Table 11 – Command Verb Requirements for CIM_SSHProtocolEndpoint

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.10.2.

Command Verb	Requirement	Comments
show	Shall	See 6.10.3.
start	Not supported	
stop	May	See 6.10.4.

No mapping is defined for the following verbs for the specified target: `dump` and `load`.

6.10.1 Ordering of Results

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

- Results for CIM_SSHProtocolEndpoint are unordered; therefore, no algorithm is defined.

6.10.2 Set

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

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

6.10.2.1 General Usage of Set for a Single Property

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *SSH Service Profile*.

6.10.2.1.1 Command Form

```
set <CIM_SSHProtocolEndpoint single instance> <propertyname>=<propertyvalue>
```

6.10.2.1.2 CIM Requirements

See CIM_SSHProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of modifiable properties.

6.10.2.1.3 Behavior Requirements

```
$instance=<CIM_SSHProtocolEndpoint single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.10.2.2 General Usage of Set for Multiple Properties

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *SSH Service Profile*.

6.10.2.2.1 Command Form

```
set <CIM_SSHProtocolEndpoint single instance> <propertyname1>=<propertyvalue1>
   <propertynamen>=<propertyvaluen>
```

6.10.2.2.2 CIM Requirements

See CIM_SSHProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.10.2.2.3 Behavior Requirements

```
$instance=<CIM_SSHProtocolEndpoint single instance>
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.10.3 Show

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

The `show` verb is used to display information about an SSH session.

Note that CIM_BindsTo and CIM_HostedAccessPoint are both Addressing Associations. Thus, an implementation of the SM CLP has a choice when exposing the address for an instance of CIM_SSHProtocolEndpoint. For completeness, mappings are shown for both associations, though only one would be applicable in a given implementation.

6.10.3.1 Show a Single Instance

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

6.10.3.1.1 Command Form

```
show <CIM_SSHProtocolEndpoint single instance>
```

6.10.3.1.2 CIM Requirements

See CIM_SSHProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.10.3.1.3 Behavior Requirements

6.10.3.1.3.1 Preconditions

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

6.10.3.1.3.2 Pseudo Code

```
$instance=<CIM_SSHProtocolEndpoint single instance>
#propertylist[] = NULL;
if ( false == #all)
   {
   #propertylist[] = {//all mandatory non-key properties }
   }
&smShowInstance ( $instance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.10.3.2 Show Multiple Instances Scoped by System

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

6.10.3.2.1 Command Form

```
show <CIM_SSHProtocolEndpoint multiple instances>
```

6.10.3.2.2 CIM Requirements

See CIM_SSHProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.10.3.2.3 Behavior Requirements

6.10.3.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ComputerSystem for which scoped endpoints (CIM_SSHProtocolEndpoint instances) are displayed. The *SSH Service Profile* requires that the CIM_SSHProtocolEndpoint instance be associated with its scoping system via an instance of the CIM_HostedAccessPoint association.

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

6.10.3.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = {//all mandatory non-key properties }
   }
&smShowInstances ( “CIM_SSHProtocolEndpoint”, “CIM_HostedAccessPoint”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.10.3.3 Show Multiple Instances Scoped by a TCPProtocolEndpoint

This command form is for the `show` verb applied to multiple instances of CIM_SSHProtocolEndpoint. This command form corresponds to UFsT-based selection within a scoping CIM_TCPProtocolEndpoint instance.

6.10.3.3.1 Command Form

```
show <CIM_SSHProtocolEndpoint multiple instances>
```

6.10.3.3.2 CIM Requirements

See CIM_SSHProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.10.3.3.3 Behavior Requirements

6.10.3.3.3.1 Preconditions

`$containerInstance` contains the instance of CIM_TCPProtocolEndpoint for which scoped endpoints (CIM_SSHProtocolEndpoint instances) are displayed. The *SSH Service Profile* requires that the CIM_SSHProtocolEndpoint instance be associated with a CIM_TCPProtocolEndpoint instance via an instance of the CIM_BindsTo association.

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

6.10.3.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
if (false == #all)
   {
   #propertylist[] = {//all mandatory non-key properties };
   }
&smShowInstances ( “CIM_SSHProtocolEndpoint”, “CIM_BindsTo”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.10.4 Stop

This section describes how to implement the `stop` verb when applied to an instance of CIM_SSHProtocolEndpoint. Implementations may support the use of the `stop` verb with CIM_SSHProtocolEndpoint.

The `stop` verb is used to disable an endpoint.

6.10.4.1 Stop a Single Instance

This command form is for the `stop` verb applied to a single instance of CIM_SSHProtocolEndpoint. The lifecycle of an SSH session corresponds to the lifecycle of the CIM_SSHProtocolEndpoint which represents it. Therefore, stopping an SSH service corresponds to a delete of the underlying instance.

6.10.4.1.1 Command Form

```
stop <CIM_SSHProtocolEndpoint single instance>
```

6.10.4.1.2 CIM Requirements

See CIM_SSHProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.10.4.1.3 Behavior Requirements

```
$instance=<CIM_SSHProtocolEndpoint single instance>
&smDeleteInstance ( $instance.getObjectPath() );
&smEnd;
```

6.11 CIM_SSHSettingData

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 12 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 12 is for informational purposes only; in case of a conflict between Table 12 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 12.

Table 12 – Command Verb Requirements for CIM_SSHSettingData

Command Verb	Requirement	Comments
create	Not supported	
delete	Not supported	
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.11.2.
show	Shall	See 6.11.3.
start	Not supported	
stop	Not supported	

No mapping is defined for the following verbs for the specified target: `dump` and `load`.

6.11.1 Ordering of Results

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

- Results for CIM_SSHSettingData are unordered; therefore, no algorithm is defined.

6.11.2 Set

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

The `set` verb is used to modify a configuration represented by an instance of CIM_SSHSettingData.

6.11.2.1 General Usage of Set for a Single Property

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *SSH Service Profile*.

6.11.2.1.1 Command Form

```
set <CIM_SSHSettingData single instance> <propertyname>=<propertyvalue>
```

6.11.2.1.2 CIM Requirements

See CIM_SSHSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of modifiable properties.

6.11.2.1.3 Behavior Requirements

```
$instance=<CIM_SSHSettingData single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstancez ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.11.2.2 General Usage of Set for Multiple Properties

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *SSH Service Profile*.

6.11.2.2.1 Command Form

```
set <CIM_SSHSettingData single instance> <propertyname1>=<propertyvalue1>
   <propertynamen>=<propertyvaluen>
```

6.11.2.2.2 CIM Requirements

See CIM_SSHSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.11.2.2.3 Behavior Requirements

```
$instance=<CIM_SSHSettingData single instance>
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.11.3 Show

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

The `show` verb is used to display information about the CIM_SSHSettingData instance.

6.11.3.1 Show a Single Instance

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

6.11.3.1.1 Command Form

```
show <CIM_SSHSettingData single instance>
```

6.11.3.1.2 CIM Requirements

See CIM_SSHSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.11.3.1.3 Behavior Requirements

6.11.3.1.3.1 Preconditions

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

6.11.3.1.3.2 Pseudo Code

```
$instance=<CIM_SSHSettingData single instance>
&lShowTCPEndpoint ( $instance, #all );
&smEnd;
```

6.11.3.2 Show Multiple Instances Scoped by ConcreteCollection

This command form is for the `show` verb applied to multiple instances of CIM_SSHSettingData. This command form corresponds to UFsT-based selection within an instance of CIM_ConcreteCollection.

6.11.3.2.1 Command Form

```
show <CIM_SSHSettingData multiple instances>
```

6.11.3.2.2 CIM Requirements

See CIM_SSHSettingData in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.11.3.2.3 Behavior Requirements

6.11.3.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ConcreteCollection for which contained CIM_SSHSettingData instances are displayed. The *SMASH Collections Profile* requires that the CIM_SSHSettingData instances be aggregated into an addressing collection via CIM_MemberOfCollection.

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

6.11.3.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
//this property list will match the property list in lShowTCPEndpoint()
if (false == #all)
   {
   #propertylist[] = {//all mandatory non-key properties }
   }
&smShowInstances ( “CIM_SSHSettingData”, “CIM_MemberOfCollection”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.12 CIM_TCPProtocolEndpoint

The `cd` and `help` verbs shall be supported as described in DSP0216.

Table 13 lists each SM CLP verb, the required level of support for the verb in conjunction with instances of the target class, and, when appropriate, a cross-reference to the section detailing the mapping for the verb and target. Table 13 is for informational purposes only; in case of a conflict between Table 13 and requirements detailed in the following sections, the text detailed in the following sections supersedes the information in Table 13.

Table 13 – Command Verb Requirements for CIM_TCPProtocolEndpoint

Command Verb	Requirement	Comments
create	May	See 6.12.2.
delete	May	See 6.12.3.
dump	Not supported	
load	Not supported	
reset	Not supported	
set	May	See 6.12.4.
show	Shall	See 6.12.5.
start	Not supported	
stop	Not supported	

No mapping is defined for the following verbs for the specified target: `dump` and `load`.

6.12.1 Ordering of Results

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

- Results for CIM_TCPProtocolEndpoint are unordered; therefore, no algorithm is defined.

6.12.2 Create

This section describes how to implement the `create` verb when applied to an instance of CIM_TCPProtocolEndpoint. Implementations may support the use of the `create` verb with CIM_TCPProtocolEndpoint.

The `create` verb is used to create an additional CIM_TCPProtocolEndpoint instance representing a port upon which the SSH service is listening.

6.12.2.1 Create Specifying the Required Port Number

In order to create an instance of CIM_TCPProtocolEndpoint, a client is required to supply the desired IP port.

6.12.2.1.1 Command Form

```
create <CIM_TCPProtocolEndpoint single instance> portnumber=<desiredport>
```

6.12.2.1.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the CIM_ProtocolService.AddListeningPort property.

6.12.2.1.3 Behavior Requirements

6.12.2.1.3.1 Preconditions

$Service contains the CIM_ProtocolService instance for which a new endpoint is created.

6.12.2.1.3.2 Pseudo Code

```
// container instance specified in the Resultant Address
//the desired address is required, if it is not specified, fail
if (NULL == <desiredport>) {
   $OperationError = smNewInstance(“CIM_Error”);
   //CIM_ERR_FAILED
   $OperationError.CIMStatusCode = 1;
   //Software Error
   $OperationError.ErrorType = 4;
   //Unknown
   $OperationError.PerceivedSeverity = 0;
   $OperationError.OwningEntity = DMTF:SMCLP;
   $OperationError.MessageID = 0x0000000D;
   $OperationError.Message = “A required property was not specified.”;
   &smAddError($job, $OperationError);
   &smMakeCommandStatus($job);
   &smEnd;
}
$Endpoint = smNewInstance (“CIM_TCPProtocolEndpoint”);
//build the parameter lists and invoke the method
%InArguments[] = {newArgument(“PortNumber”, <desiredport>}
%OutArguments[] = { newArgument(“Endpoint”,
                    $Endpoint.GetObjectPath()) };
//invoke method
#returnStatus = smOpInvokeMethod ($Service.GetObjectPath(),
   “AddListeningEndpoint”,
   %InArguments[],
   %OutArguments[]);
// process return code to CLP Command Status
if (0 != #Error.code)  {
   //method invocation failed
   if ( (NULL != #Error.$error) && (NULL != #Error.$error[0]) )    {
      // if the method invocation contains an embedded error
      // use it for the Error for the overall job
   &smAddError($job, #Error.$error[0]);
   &smMakeCommandStatus($job);
   &smEnd;
   }
   else if (#Error.code == 17)   {
   //trap for CIM_METHOD_NOT_FOUND
```

```
   //and make nice Unsupported msg.
      //unsupported
      $OperationError = smNewInstance(“CIM_Error”);
      //CIM_ERR_NOT_SUPPORTED
      $OperationError.CIMStatusCode = 7;
      //Other
      $OperationError.ErrorType = 1;
      //Low
      $OperationError.PerceivedSeverity = 2;
      $OperationError.OwningEntity = DMTF:SMCLP;
      $OperationError.MessageID = 0x00000001;
      $OperationError.Message = “Operation is not supported.”;
      &smAddError($job, $OperationError);
      &smMakeCommandStatus($job);
      &smEnd;
}
else {
   //operation failed, but no detailed error instance, need to make one up
   //make an Error instance and associate with job for Operation
      $OperationError = smNewInstance(“CIM_Error”);
      //CIM_ERR_FAILED
      $OperationError.CIMStatusCode = 1;
      //Software Error
      $OperationError.ErrorType = 4;
      //Unknown
      $OperationError.PerceivedSeverity = 0;
      $OperationError.OwningEntity = DMTF:SMCLP;
      $OperationError.MessageID = 0x00000009;
      $OperationError.Message = “An internal software error has occurred.”;
      &smAddError($job, $OperationError);
      &smMakeCommandStatus($job);
      &smEnd;
   }
}//if CIM op failed
else if (0 == #returnStatus)  {
   //completed successfully
   &lShowTCPEndpoint($Endpoint, “false”);
   &smEnd;
}
else if (4 == #returnStatus)  {
   //generic failure
   $OperationError = smNewInstance(“CIM_Error”);
   //CIM_ERR_FAILED
   $OperationError.CIMStatusCode = 1;
   //Other
   $OperationError.ErrorType = 1;
   //Low
   $OperationError.PerceivedSeverity = 2;
   $OperationError.OwningEntity = DMTF:SMCLP;
```

```
   $OperationError.MessageID = 0x00000002;
   $OperationError.Message = “Failed.  No further information is available.”;
   &smAddError($job, $OperationError);
   &smMakeCommandStatus($job);
}
else   {
   //invalid parameter
   $OperationError = smNewInstance(“CIM_Error”);
   //CIM_ERR_FAILED
   $OperationError.CIMStatusCode = 1;
   //Other
   $OperationError.ErrorType = 1;
   //Low
   $OperationError.PerceivedSeverity = 2;
   $OperationError.OwningEntity = DMTF:SMCLP;
   $OperationError.MessageID = 0x00000004;
   $OperationError.Message = “One or more parameters specified are invalid.”;
   &smAddError($job, $OperationError);
   &smMakeCommandStatus($job);
   &smEnd;
}
```

6.12.3 Delete

This section describes how to implement the `delete` verb when applied to an instance of CIM_TCPProtocolEndpoint. Implementations may support the use of the `delete` verb with CIM_TCPProtocolEndpoint.

The `delete` command is used to remove an instance of CIM_TCPProtocolEndpoint which represents a virtual MAC.

6.12.3.1 Delete a Single Instance

Delete a single instance of CIM_TCPProtocolEndpoint.

6.12.3.1.1 Command Form

`delete <CIM_TCPProtocolEndpoint single instance>`

6.12.3.1.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the CIM_TCPProtocolEndpoint property.

6.12.3.1.3 Behavior Requirements

```
$instance=<CIM_TCPProtocolEndpoint single instance>
&smOpDeleteInstance ( $instance.GetObjectPath() );
&smEnd;
```

6.12.4 Set

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

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

6.12.4.1 General Usage of Set for a Single Property

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *SSH Service Profile*.

6.12.4.1.1 Command Form

```
set <CIM_TCPProtocolEndpoint single instance> <propertyname>=<propertyvalue>
```

6.12.4.1.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of modifiable properties.

6.12.4.1.3 Behavior Requirements

```
$instance=<CIM_TCPProtocolEndpoint single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.12.4.2 General Usage of Set for Multiple Properties

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

The requirement for supporting modification of a property using this command form shall be equivalent to the requirement for supporting modification of the property using the ModifyInstance operation as defined in the *SSH Service Profile*.

6.12.4.2.1 Command Form

```
set <CIM_TCPProtocolEndpoint single instance> <propertyname1>=<propertyvalue1>
   <propertyname n>=<propertyvalue n>
```

6.12.4.2.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.12.4.2.3 Behavior Requirements

```
$instance=<CIM_TCPProtocolEndpoint single instance>
#propertyNames[] = {<propertyname>};
for #i < n
   {
   #propertyNames[#i] = <propertname#i>
   #propertyValues[#i] = <propertyvalue#i>
   }
&smSetInstance ( $instance, #propertyNames[], #propertyValues[] );
&smEnd;
```

6.12.5 Show

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

The `show` verb is used to display information about a CIM_TCPProtocolEndpoint instance.

6.12.5.1 Show a Single Instance

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

6.12.5.1.1 Command Form

```
show <CIM_TCPProtocolEndpoint single instance>
```

6.12.5.1.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.12.5.1.3 Behavior Requirements

6.12.5.1.3.1 Preconditions

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

6.12.5.1.3.2 Pseudo Code

```
$instance=<CIM_TCPProtocolEndpoint single instance>
&lShowTCPEndpoint ( $instance, #all );
&smEnd;
```

6.12.5.2 Show Multiple Instances Scoped by a System

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

6.12.5.2.1 Command Form

```
show <CIM_TCPProtocolEndpoint multiple instances>
```

6.12.5.2.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.12.5.2.3 Behavior Requirements

6.12.5.2.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ComputerSystem for which we are displaying scoped endpoints (CIM_TCPProtocolEndpoint instances). The *SSH Service Profile* requires that the CIM_TCPProtocolEndpoint instance be associated with its scoping system via an instance of the CIM_HostedAccessPoint association.

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

6.12.5.2.3.2 Pseudo Code

```
#propertylist[] = NULL;
//this property list will match the property list in lShowTCPEndpoint()
if (false == #all)
   {
   #propertylist[] = { //all mandatory non-key properties };
   }
&smShowInstances ( “CIM_TCPProtocolEndpoint”, “CIM_HostedAccessPoint”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.12.5.2.4 Show Multiple Instances Scoped by a ProtocolService

This command form is for the `show` verb applied to multiple instances of CIM_TCPProtocolEndpoint. This command form corresponds to UFsT-based selection within a scoping ProtocolService instance.

6.12.5.2.5 Command Form

```
show <CIM_TCPProtocolEndpoint multiple instances>
```

6.12.5.2.6 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.12.5.2.7 Behavior Requirements

6.12.5.2.7.1 Preconditions

`$containerInstance` contains the instance of CIM_ProtocolService for which we are displaying associated endpoints (CIM_TCPProtocolEndpoint instances). The *SSH Service Profile* requires that the CIM_TCPProtocolEndpoint instance be associated with an instance of CIM_ProtocolService via an instance of CIM_ServiceAccessBySAP.

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

6.12.5.2.7.2 Pseudo Code

```
#propertylist[] = NULL;
//this property list will match the property list in lShowTCPEndpoint()
if (false == #all)
   {
   #propertylist[] = { //all mandatory non-key properties };
   }
&smShowInstances ( “CIM_TCPProtocolEndpoint”, “CIM_ServiceAccessBySAP”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.12.5.3 Show Multiple Instances Scoped by a ProtocolEndpoint

This command form is for the `show` verb applied to multiple instances of CIM_TCPProtocolEndpoint. This command form corresponds to UFsT-based selection within a scoping CIM_ProtocolEndpoint instance with which the CIM_TCPProtocolEndpoint instances are associated via instances of CIM_BindsTo.

6.12.5.3.1 Command Form

```
show <CIM_TCPProtocolEndpoint multiple instances>
```

6.12.5.3.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *SSH Service Profile* for the list of mandatory properties.

6.12.5.3.3 Behavior Requirements

6.12.5.3.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ProtocolEndpoint for which we are displaying associated endpoints (CIM_TCPProtocolEndpoint instances). The *SSH Service Profile* indicates that the CIM_TCPProtocolEndpoint instance can be associated with an instance of CIM_ProtocolEndpoint via an instance of CIM_BindsTo.

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

6.12.5.3.3.2 Pseudo Code

```
#propertylist[] = NULL;
//this property list will match the property list in lShowTCPEndpoint()
if (false == #all)
   {
   #propertylist[] = { //all mandatory non-key properties };
   }
&smShowInstances ( “CIM_TCPProtocolEndpoint”, “CIM_BindsTo”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

ANNEX A
(informative)

Change Log

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