Document Number: DSP0820

Date: 2009-07-14

Version: 1.0.0

Telnet Service Profile SM CLP Command Mapping Specification

Document Type: Specification

Document Status: DMTF Standard

Document Language: E

CONTENTS

Tables

75

Foreword

The *Telnet Service Profile SM CLP Command Mapping Specification* (DSP0820) was prepared by the Server Management Working Group.

Conventions

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

Acknowledgements

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

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

Introduction

This document defines the SM CLP mapping for CIM elements described in the *Telnet Service Profile*. The information in this specification, combined with the *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 *Telnet Service Profile* using CIM operations.

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

Telnet 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 *Telnet 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 DSP1016, *Telnet Service Profile 1.0*,
http://www.dmtf.org/standards/published_documents/DSP1016_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
Command Line Protocol

4.3
DMTF
Distributed Management Task Force

4.4
IETF
Internet Engineering Task Force

4.5
SM
Server Management

4.6
SMI-S
Storage Management Initiative Specification

4.7
SNIA
Storage Networking Industry Association

4.8
UFsT
User Friendly selection Tag

5 Recipes

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

- smStartRSC()
- smStopRSC()
- smResetRSC()
- smShowInstance()
- smShowInstances()
- smSetInstance()
- smShowAssociationInstances()
- smShowAssociationInstance()
- smDeleteInstance
- smMakeCommandStatus
- smNewInstance

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

Table 1 – Local Recipes

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

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

5.1 lShowTCPEndpoint

5.1.1 Description

lShowTCPEndpoint is a reusable recipe for displaying an instance of CIM_TCPProtocolEndpoint. A recipe is defined for reuse 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 *Telnet 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 *Telnet 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 *Telnet 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_TelnetProtocolEndpoint 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_TelnetProtocolEndpoint. A single instance of CIM_TCPProtocolEndpoint is associated with each instance of CIM_TelnetProtocolEndpoint. 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 *Telnet 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_TelnetProtocolEndpoint 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_TelnetProtocolEndpoint 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 *Telnet 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_TelnetProtocolEndpoint 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;
```

6.1.2.4.4 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.

6.1.2.4.5 Command Form

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

6.1.2.4.6 CIM Requirements

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

6.1.2.4.7 Behavior Requirements

6.1.2.4.7.1 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.

6.1.2.4.7.2 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.

336 **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	

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

339 ### 6.2.1 Ordering of Results

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

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

343 ### 6.2.2 Show

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

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

349 #### 6.2.2.1 Show a Single Instance – CIM_TelnetCapabilities Reference

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

354 ##### 6.2.2.1.1 Command Form

355 ```
show <CIM_ElementCapabilities single instance>
```

356 ##### 6.2.2.1.2 CIM Requirements

357 See CIM_ElementCapabilities in the “CIM Elements” section of the *Telnet Service Profile* for the list of
358 mandatory properties.
```

6.2.2.1.3 Behavior Requirements

6.2.2.1.3.1 Preconditions

`$instance` contains the instance of CIM_TelnetCapabilities 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_TelnetCapabilities 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 *Telnet 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 *Telnet 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_TelnetCapabilities 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 *Telnet 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_TelnetSettingData and CIM_TelnetProtocolEndpoint

This command form corresponds to a `show` command issued against CIM_ElementSettingData where the reference specified is to an instance of CIM_TelnetSettingData. Note that when an instance of CIM_TelnetSettingData is associated with an instance of CIM_TelnetProtocolEndpoint, 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 *Telnet 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_TelnetSettingData 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_TelnetProtocolEndpoint Reference

This command form corresponds to a `show` command issued against CIM_ElementSettingData where the reference specified is to an instance of CIM_TelnetProtocolEndpoint. Note that when an instance of CIM_TelnetSettingData is associated with an instance of CIM_TelnetProtocolEndpoint, 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 *Telnet 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_TelnetProtocolEndpoint 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_TelnetSettingData and CIM_TelnetProtocolEndpoint

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 *Telnet 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_TelnetSettingData which is referenced by CIM_ElementSettingData.

`$instanceB` contains the instance of CIM_TelnetProtocolEndpoint 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_TelnetSettingData 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_TelnetSettingData. Note that when an instance of CIM_TelnetSettingData 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 *Telnet 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_TelnetSettingData 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_TelnetSettingData 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 *Telnet 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 *Telnet 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_TelnetSettingData 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.

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	

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

6.4.1 Ordering of Results

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

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

6.4.2 Show

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

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

6.4.2.1 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_HostedAccessPoint where only one reference is specified and the reference is to an instance of CIM_ComputerSystem.

6.4.2.1.1 Command Form

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

6.4.2.1.2 CIM Requirements

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

6.4.2.1.3 Behavior Requirements

6.4.2.1.3.1 Preconditions

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

620 **6.4.2.1.3.2 Pseudo Code**

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

623 **6.4.2.2 Show a Single Instance – CIM_TCPProtocolEndpoint or CIM_TelnetProtocolEndpoint**
624 **Reference**

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

629 **6.4.2.2.1 Command Form**

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

631 **6.4.2.2.2 CIM Requirements**

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

634 **6.4.2.2.3 Behavior Requirements**

635 **6.4.2.2.3.1 Preconditions**

636 `$instance` contains the instance of CIM_TCPProtocolEndpoint or CIM_TelnetProtocolEndpoint which is
637 referenced by CIM_HostedAccessPoint.

638 **6.4.2.2.3.2 Pseudo Code**

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

641 **6.4.2.3 Show a Single Instance – Both References**

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

645 **6.4.2.3.1 Command Form**

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

647 **6.4.2.3.2 CIM Requirements**

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

650 **6.4.2.3.3 Behavior Requirements**

651 **6.4.2.3.3.1 Preconditions**

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

654 `$instanceB` contains the instance of CIM_TelnetProtocolEndpoint or CIM_TCPProtocolEndpoint which
655 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.

682 **6.5.2.1.1 Command Form**

683 ```
show <CIM_HostedService multiple instances>
```

684 **6.5.2.1.2 CIM Requirements**

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

687 **6.5.2.1.3 Behavior Requirements**

688 **6.5.2.1.3.1 Preconditions**

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

690 **6.5.2.1.3.2 Pseudo Code**

691 ```
&smShowAssociationInstances ( “CIM_HostedService”, $instance.getObjectPath() );
692 &smEnd;
```

693 **6.5.2.2 Show a Single Instance – CIM_ProtocolService Reference**

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

698 **6.5.2.2.1 Command Form**

699 ```
show <CIM_HostedService single instance>
```

700 **6.5.2.2.2 CIM Requirements**

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

703 **6.5.2.2.3 Behavior Requirements**

704 **6.5.2.2.3.1 Preconditions**

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

706 **6.5.2.2.3.2 Pseudo Code**

707 ```
&smShowAssociationInstances ( “CIM_HostedService”, $instance.getObjectPath() );
708 &smEnd;
```

709 **6.5.2.3 Show a Single Instance – Both References**

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

713 **6.5.2.3.1 Command Form**

714 ```
show <CIM_HostedService single instance>
```
```

6.5.2.3.2 CIM Requirements

See CIM_HostedService in the “CIM Elements” section of the *Telnet 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_TelnetProtocolEndpoint 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_TelnetProtocolEndpoint. A single instance of CIM_ProtocolService is associated with each instance of a CIM_TelnetProtocolEndpoint 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 *Telnet 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_TelnetProtocolEndpoint 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_TelnetProtocolEndpoint 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 *Telnet 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 *Telnet 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_TelnetProtocolEndpoint 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 6.7.4.2.4.
stop	May	See 6.7.5.

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 *Telnet 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 *Telnet 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 *Telnet Service Profile*.

6.7.3.2.1 Command Form

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

6.7.3.2.2 CIM Requirements

See CIM_ProtocolService in the “CIM Elements” section of the *Telnet 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 *Telnet 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 *Telnet 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 scoped instances of the CIM_ProtocolService are displayed. The *Telnet 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;
```

6.7.4.2.4 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.

6.7.4.3 Start a Single Instance

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

6.7.4.3.1 Command Form

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

6.7.4.3.2 CIM Requirements

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

6.7.4.3.3 Behavior Requirements

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

6.7.5 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.

6.7.5.1 Stop a Single Instance

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

6.7.5.1.1 Command Form

```
stop <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>
&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_ProtocolService in the “CIM Elements” section of the *Telnet 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 *Telnet 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 *Telnet 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_TelnetCapabilities

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.

1042 **Table 10 – Command Verb Requirements for CIM_TelnetCapabilities**

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 6.9.2.
start	Not supported	
stop	Not supported	

1043 No mapping is defined for the following verbs for the specified target: `create`, `delete`, `dump`, `load`,
1044 `reset`, `set`, `start`, and `stop`.

1045 ### 6.9.1 Ordering of Results

1046 When results are returned for multiple instances of CIM_TelnetCapabilities, implementations shall utilize
1047 the following algorithm to produce the natural (that is, default) ordering:

1048 - Results for CIM_TelnetCapabilities are unordered; therefore, no algorithm is defined.

1049 ### 6.9.2 Show

1050 This section describes how to implement the `show` verb when applied to an instance of
1051 CIM_TelnetCapabilities. Implementations shall support the use of the `show` verb with
1052 CIM_TelnetCapabilities.

1053 The `show` verb is used to display information about an instance or instances of the
1054 CIM_TelnetCapabilities class.

1055 #### 6.9.2.1 Show a Single Instance

1056 This command form is for the `show` verb applied to a single instance of CIM_TelnetCapabilities.

1057 ##### 6.9.2.1.1 Command Form

1058 **`show <CIM_TelnetCapabilities`** ***`single instance`*****`>`**

1059 ##### 6.9.2.1.2 CIM Requirements

1060 See CIM_TelnetCapabilities in the “CIM Elements” section of the *Telnet Service Profile* for the list of
1061 mandatory properties.

1062 ##### 6.9.2.1.3 Behavior Requirements

1063 ###### 6.9.2.1.3.1 Preconditions

1064 `#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_TelnetCapabilities 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_TelnetCapabilities. This command form corresponds to UFsT-based selection within a capabilities collection.

6.9.2.2.1 Command Form

```
show <CIM_TelnetCapabilities multiple instances>
```

6.9.2.2.2 CIM Requirements

See CIM_TelnetCapabilities in the “CIM Elements” section of the *Telnet 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_TelnetCapabilities”, “CIM_MemberOfCollection”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.10 CIM_TelnetProtocolEndpoint

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.

1104 **Table 11 – Command Verb Requirements for CIM_TelnetProtocolEndpoint**

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.
show	Shall	See 6.10.3.
start	Not supported	
stop	May	See 6.10.4.

1105 No mapping is defined for the following verbs for the specified target: `dump` and `load`.

1106 ### 6.10.1 Ordering of Results

1107 When results are returned for multiple instances of CIM_TelnetProtocolEndpoint, implementations shall
1108 utilize the following algorithm to produce the natural (that is, default) ordering:

1109 - Results for CIM_TelnetProtocolEndpoint are unordered; therefore, no algorithm is defined.

1110 ### 6.10.2 Set

1111 This section describes how to implement the `set` verb when it is applied to an instance of
1112 CIM_TelnetProtocolEndpoint. Implementations may support the use of the `set` verb with
1113 CIM_TelnetProtocolEndpoint.

1114 The `set` verb is used to modify descriptive properties of the CIM_TelnetProtocolEndpoint instance.

1115 #### 6.10.2.1 General Usage of Set for a Single Property

1116 This command form corresponds to the general usage of the `set` verb to modify a single property of a
1117 target instance. This is the most common case.

1118 The requirement for supporting modification of a property using this command form shall be equivalent to
1119 the requirement for supporting modification of the property using the ModifyInstance operation as defined
1120 in the *Telnet Service Profile*.

1121 ##### 6.10.2.1.1 Command Form

1122 **`set <CIM_TelnetProtocolEndpoint single instance> <propertyname>=<propertyvalue>`**

1123 ##### 6.10.2.1.2 CIM Requirements

1124 See CIM_TelnetProtocolEndpoint in the “CIM Elements” section of the *Telnet Service Profile* for the list of
1125 modifiable properties.

1126 ##### 6.10.2.1.3 Behavior Requirements

```
1127 $instance=<CIM_TelnetProtocolEndpoint single instance>
1128 #propertyNames[] = {<propertyname>};
1129 #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 *Telnet Service Profile*.

6.10.2.2.1 Command Form

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

6.10.2.2.2 CIM Requirements

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

6.10.2.2.3 Behavior Requirements

```
$instance=<CIM_TelnetProtocolEndpoint 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_TelnetProtocolEndpoint. Implementations shall support the use of the `show` verb with CIM_TelnetProtocolEndpoint.

The `show` verb is used to display information about a Telnet 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_TelnetProtocolEndpoint. 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_TelnetProtocolEndpoint.

6.10.3.1.1 Command Form

```
show <CIM_TelnetProtocolEndpoint single instance>
```

6.10.3.1.2 CIM Requirements

See CIM_TelnetProtocolEndpoint in the "CIM Elements" section of the *Telnet 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_TelnetProtocolEndpoint 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 a System

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

6.10.3.2.1 Command Form

```
show <CIM_TelnetProtocolEndpoint multiple instances>
```

6.10.3.2.2 CIM Requirements

See CIM_TelnetProtocolEndpoint in the "CIM Elements" section of the *Telnet 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_TelnetProtocolEndpoint instances) are displayed. The *Telnet Service Profile* requires that the CIM_TelnetProtocolEndpoint 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_TelnetProtocolEndpoint", "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_TelnetProtocolEndpoint. This command form corresponds to UFsT-based selection within a scoping CIM_TCPProtocolEndpoint instance.

6.10.3.3.1 Command Form

```
show <CIM_TelnetProtocolEndpoint multiple instances>
```

6.10.3.3.2 CIM Requirements

See CIM_TelnetProtocolEndpoint in the "CIM Elements" section of the *Telnet 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_TelnetProtocolEndpoint instances) are displayed. The *Telnet Service Profile* requires that the CIM_TelnetProtocolEndpoint 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_TelnetProtocolEndpoint", "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_TelnetProtocolEndpoint. Implementations may support the use of the `stop` verb with CIM_TelnetProtocolEndpoint.

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_TelnetProtocolEndpoint. The lifecycle of a Telnet session corresponds to the lifecycle of the CIM_TelnetProtocolEndpoint which represents it. Therefore, stopping a Telnet service corresponds to a delete of the underlying instance.

6.10.4.1.1 Command Form

```
stop <CIM_TelnetProtocolEndpoint single instance>
```

6.10.4.1.2 CIM Requirements

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

6.10.4.1.3 Behavior Requirements

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

6.11 CIM_TelnetSettingData

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_TelnetSettingData

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_TelnetSettingData, implementations shall utilize the following algorithm to produce the natural (that is, default) ordering:

- Results for CIM_TelnetSettingData 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_TelnetSettingData. Implementations may support the use of the `set` verb with CIM_TelnetSettingData.

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

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 *Telnet Service Profile*.

6.11.2.1.1 Command Form

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

6.11.2.1.2 CIM Requirements

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

6.11.2.1.3 Behavior Requirements

```
$instance=<CIM_TelnetSettingData single instance>
#propertyNames[] = {<propertyname>};
#propertyValues[] = {<propertyvalue>};
&smSetInstance ( $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 *Telnet Service Profile*.

6.11.2.2.1 Command Form

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

6.11.2.2.2 CIM Requirements

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

6.11.2.2.3 Behavior Requirements

```
$instance=<CIM_TelnetSettingData 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_TelnetSettingData. Implementations shall support the use of the `show` verb with CIM_TelnetSettingData.

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

6.11.3.1 Show a Single Instance

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

6.11.3.1.1 Command Form

```
show <CIM_TelnetSettingData single instance>
```

6.11.3.1.2 CIM Requirements

See CIM_TelnetSettingData in the “CIM Elements” section of the *Telnet 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_TelnetSettingData 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_TelnetSettingData. This command form corresponds to UFsT-based selection within an instance of CIM_ConcreteCollection.

6.11.3.2.1 Command Form

```
show <CIM_TelnetSettingData multiple instances>
```

6.11.3.2.2 CIM Requirements

See CIM_TelnetSettingData in the “CIM Elements” section of the *Telnet 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_TelnetSettingData instances are displayed. The *SMASH Collections Profile* requires that the CIM_TelnetSettingData 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_TelnetSettingData”, “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 Telnet service is listening.

6.12.2.1 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 *Telnet 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 we are creating a new endpoint.

6.12.2.1.3.2 Pseudo Code

```
// container instance specified in the Resultant Address
//the desired address is required, if its 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 (1 == #returnStatus)  {
   //and make nice Unsupported msg.
      $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   {
   //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);
}
```

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 *Telnet 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 *Telnet 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 *Telnet 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 *Telnet Service Profile*.

6.12.4.2.1 Command Form

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

6.12.4.2.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *Telnet 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 *Telnet 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 *Telnet 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 scoped endpoints (CIM_TCPProtocolEndpoint instances) are displayed. The *Telnet 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;
```

6.12.5.3 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.

6.12.5.3.1 Command Form

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

6.12.5.3.2 CIM Requirements

See CIM_TCPProtocolEndpoint in the “CIM Elements” section of the *Telnet 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_ProtocolService for which associated endpoints (CIM_TCPProtocolEndpoint instances) are displayed. The *Telnet 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.

6.12.5.3.3.2 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_ServiceAccessBySAP”,
   $containerInstance.getObjectPath(), #propertylist[] );
&smEnd;
```

6.12.5.4 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.

6.12.5.4.1 Command Form

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

6.12.5.4.2 CIM Requirements

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

6.12.5.4.3 Behavior Requirements

6.12.5.4.3.1 Preconditions

`$containerInstance` contains the instance of CIM_ProtocolEndpoint for which associated endpoints (CIM_TCPProtocolEndpoint instances) are displayed. The *Telnet 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.

6.12.5.4.3.2 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