White Paper

Introduction how to use CIM-based Interfaces of the iRMC S4

This document describes how to use CIM-based interfaces in Fujitsu PRIMERGY servers with iRMC S4 (integrated Remote Management Controller) and ServerView Agents. Especially the basic setup and examples are presented for an easy start to CIM-based server management.
Introduction

This document is aimed at users of CIM-based interfaces (CIM: Common Information Model) and describes how to use CIM-based Interfaces with iRMC S4 and ServerView Agents. Especially the basic setup and examples are presented for an easy start to CIM-based server management.

1. Basics

“CIM provides a common definition of management information for systems, networks, applications and services, and allows for vendor extensions. CIM's common definitions enable vendors to exchange semantically rich management information between systems throughout the network. CIM is composed of a Specification and a Schema. The Schema provides the actual model descriptions, while the Specification defines the details for integration with other management models.”

Making CIM available to a system it must provide a WBEM-Server that contains a CIM-Server and needed CIM-Clients. A CIM-Client on the WBEM-Server is in the end the implementation of a certain protocol that can exchange CIM-based information (e.g. WS-Man or CIM-XML). The CIM-Client handles requests and responses from a WBEM-Client that may access CIM-based information. The CIM-Client delegates the request and responses to and from the CIM Object Manager (CIMOM). For a certain request the CIMOM checks if the requested information is available in the current CIM-Repository. If the Information is available the CIM-Repository and also the right CIM-Provider can be determined. The CIM-Provider is requested to accesses the managed elements (Hardware or Software components that are managed by CIM) and builds actual instances of the accessed information as described in the CIM-Schema. These instances are returned to the CIM-Client via the CIMOM and are transferred to the requesting WBEM-Client with the right protocol (see figure 1 and [DSP2001]).

Additionally a set of so-called Profile Specification Documents exists, that encapsulates certain parts of the whole CIM-Schema regarding different management aspects (e.g. Computer System Profile, Fan Profile, Physical Asset Profile and a lot more). Each Profile Specification Document describes the behavioral aspect on a set of CIM-Elements. A CIM-Element can define a logical system element, a physical system element or an association relating the system elements to each other. The behavioral aspects are:

- Creation of instances of the CIM-Classes
- Directives to fill properties defined in the MOF-File for a certain CIM-Class
- Execution of methods on instances

The standard CIM-Schema may be extended by an Original Equipment Manufacturer (OEM) to define OEM-specific management information. In this case additional CIM-Providers or additional properties and methods for existing standard CIM-Providers can be defined.

For more foundation you can have a look at: [http://www.wbemsolutions.com/tutorials/DMTF/](http://www.wbemsolutions.com/tutorials/DMTF/).

2. General data access

The data-model is defined in the profile documents.

For DMTF standard-profiles see: [http://www.dmtf.org/standards/published_documents](http://www.dmtf.org/standards/published_documents)
- DSP1000 to DSP1999.


Each profile describes a set of CIM-Classes and CIM-Associations regarding a certain management purpose. A CIM-Class defines a set of CIM-Properties and CIM-Methods. A CIM-Property defines a name, value, data type and some additional meta-data called qualifiers. A CIM-Method is the definition of an extrinsic method to execute a synchronous or asynchronous management function. A CIM-Association defines a bidirectional link between two CIM-Classes.

---

1 [http://dmtf.org/standards/cim](http://dmtf.org/standards/cim)
The CIM defines also a set of so called intrinsic methods. These are standard methods that could be applied on the CIM-Elements. The most important intrinsic method is getting the CIM-Instances for a certain CIM-Class or CIM-Association. A CIM-Instance is the runtime representation of CIM-Classes and CIM-Association during runtime, thus the elements you will work with. Each CIM-Instance provides the actual CIM-Property values. As working with CIM-Instances you are also able to modify a CIM-Instance to change a certain CIM-Property value or call a defined CIM-Method. Each CIM-Instance is built conform to the definition of its CIM-Class or CIM-Association. Instances of CIM-Associations allow also the traversal between different CIM-Instances of CIM-Classes.

Most general access needs:
- Get all CIM-Instances of a CIM-Class or CIM-Association
- Get a certain CIM-Instance of a CIM-Class or CIM-Association
- Get all instances of a CIM-Association for a certain CIM-Class
- Execute a CIM-Method on a certain CIM-Instance
- Execute the intrinsic method modify instance to change CIM-Property values for a certain CIM-Instance

Example:

<table>
<thead>
<tr>
<th>SVS_iRMCBaseServer</th>
<th>SVS_iRMCAssociatedServerSettings</th>
<th>SVS_iRMCServerConfigSettings</th>
</tr>
</thead>
<tbody>
<tr>
<td>CreationClassName</td>
<td>string (KEY)</td>
<td>ElementName: string</td>
</tr>
<tr>
<td>Name</td>
<td>string (KEY)</td>
<td>InstanceID: string (KEY)</td>
</tr>
<tr>
<td>ElementName</td>
<td>string</td>
<td>ConfErrorOffRestartTime: uint16</td>
</tr>
<tr>
<td>HealthState</td>
<td>uint16</td>
<td>PowerOnCounter: datetime</td>
</tr>
<tr>
<td>EnabledState</td>
<td>uint16</td>
<td>ConfPowerCycleDelay: uint8</td>
</tr>
<tr>
<td>RequestStateChange()</td>
<td></td>
<td>ResetPowerOnCounter()</td>
</tr>
</tbody>
</table>

For getting a specific instance the key properties must be passed on to the enumerate request. Key properties identify a specific instance in a unique way.

To call extrinsic methods like ResetPowerOnCounter on the specific SVS_iRMCServerConfigSettings instance

SVS_iRMCServerConfigSettings (InstanceID = “SVS:iRMCServerConfigSettings”)

You need to pass the key properties.

Also for calling the intrinsic method modify instance you need to access a certain instance by passing the key property values.
Modify instance for CIM-Instances of SVS_iRMCServerConfigSetting(InstanceID = “SVS:iRMCServerConfigSettings”) for property ConfErrorOffRestartTime = 4

executed on the instance:

<table>
<thead>
<tr>
<th>ElementName</th>
<th>= Configuration Space Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceID</td>
<td>= SVS:iRMCServerConfigSettings</td>
</tr>
<tr>
<td>ConfErrorOffRestartTime</td>
<td>= 2</td>
</tr>
<tr>
<td>PowerOnCounter</td>
<td>= 143.05.:00:00</td>
</tr>
<tr>
<td>ConfPowerCycleDelay</td>
<td>= 7</td>
</tr>
</tbody>
</table>

will result in:

<table>
<thead>
<tr>
<th>ElementName</th>
<th>= Configuration Space Settings</th>
</tr>
</thead>
<tbody>
<tr>
<td>InstanceID</td>
<td>= SVS:iRMCServerConfigSettings</td>
</tr>
<tr>
<td>ConfErrorOffRestartTime</td>
<td>= 4</td>
</tr>
<tr>
<td>PowerOnCounter</td>
<td>= 143.05.:00:00</td>
</tr>
<tr>
<td>ConfPowerCycleDelay</td>
<td>= 7</td>
</tr>
</tbody>
</table>

3. **CIM-based interfaces**

CIM-based interfaces are provided in-band and out-of-band.

3.1. **Basic configuration settings**

3.1.1. **Resource URI**


3.1.2. **Class Name**

The Standard DMTF class name prefix is “CIM_”, whereas the FTS Class Name prefix is “SVS_PGY” for OEM in-band CIM-Provider and “SVS_iRMC” for OEM out-of-band CIM-Provider.

3.1.3. **Namespace**

To access a CIM object, you must specify the namespace, in which the CIM object is stored. See the following examples:

- Standard DMTF namespace (and also default): root\cimv2
- Interop namespace: root\interop
- FTS implementation namespace: root\svs

3.1.4. **Ports**

Standard ports, also used in many applications for WS-MAN based access, are:

- 80 HTTP
- 443 HTTPS

Standard ports, also used in many applications for WS-MAN based access, are:

- 5988 CIM-XML
- 5989 CIM-XML secure port (via ssl)

3.1.5. **Encoding**

The supported encoding is: utf-8

3.1.6. **Authentication**

The supported authentication is: basic.
### 3.2. Out-of-band CIM-based Interfaces

For out-of-band CIM-based management you access the integrated Remote Management Controller S4 (iRMC). The iRMC S4 supports CIM-XML and WS-Man. Besides providing remote access the iRMC S4 also support SMASH-CLP.

For WS-Man and CIM-XML the basic configuration settings apply (see Section 3.1).

#### 3.2.1. SMASH-CLP

For accessing CIM-based information via SMASH-CLP you need access to the Remote Manager of the iRMC S4. You need to login to Remote Manager via Telnet or SSH. Be sure Telnet and or SSH is enabled on the iRMC S4.


#### 3.2.1.1. Example: SMASH-CLP log-in example

Log-in to iRMC S4 via ssh admin@targetIP

```
System Information Menu
***************************************************************************
* Welcome to PRIMERGY Remote Manager                             *
* Firmware Revision 97.71F (1.00)                                   *
* SDR 3.71 ID 0356 RX300S8                                         *
* Firmware built Feb 6 2015 11:15:32 CET                           *
***************************************************************************

System Type : PRIMERGY RX300 S8
System ID : YLN7000507
System Name : WIN-VNCUONE34B
System OS : Windows Server 2012 R2 Standard
System Status : (Identify LED is OFF)
Power Status : On
Asset Tag : System Asset Tag

Main Menu
(1) System Information...
(2) Power Management...
(3) Enclosure Information...
(4) Service Processor...
(5) RAID Management...
(c) Change password
(r) Console Redirection (EMS/SAC)
(s) Start a Command Line shell...
(l) Console Logging

Enter selection or (0) to quit:

Press s for "Start a Command Line shell..."

Shell Menu
(1) Start SMASH CLP shell...
(2) Start IPMI Terminal Mode shell...

Press 1 for "Start SMASH CLP shell..."

>> SMASH-CLP <<
```

#### 3.2.1.2. Example: Getting CIM-based information via SMASH-CLP

General available Verbs:
show / cd / exit / help / version / ...

```
CIM-based information accessible by SMASH-CLP is organized in a tree structure. Each target shows its accessible (sub)Targets and available verbs.

“show -l all” will start a complete walkthrough for all SMASH-CLP (sub)Targets. You may start manual exploration with a `show` and get access (sub)Targets with “cd”. “cd ..” navigates back to the (super)Target.

Example to power on the host

```
>> SMASH-CLP <<

-> show

Command Status: COMMAND COMPLETED
ufip=/
Targets: admin1
Verbs: cd exit help show version

-> cd admin1
Command Status: COMMAND COMPLETED
ufip=/admin1

-> show

Command Status: COMMAND COMPLETED
ufip=/admin1
Targets: hdwr1 profile1 profile2 system1
Associations:
  SVS_iRMCElementConformsToProfile=>/admin1/profile1
  SVS_iRMCBaseServerSystemComponent=>/admin1/system1
Properties:
  ElementName=SM CLP Admin Domain
  CreationClassName=SVS_iRMCAccessDomain
  Name=MAP
Verbs: cd exit help show version

-> cd system1
Command Status: COMMAND COMPLETED
ufip=/admin1/system1

-> show

Command Status: COMMAND COMPLETED
ufip=/admin1/system1
Targets: capabilities1 cpu1 enetport1 enetport2
```
 Associations:
  SVS_IRMCElementConformsToProfile=>/admin1/system1/profile1
  SVS_IRMCElementConformsToProfile=>/admin1/system1/profile2
  SVS_IRMCElementView=>/admin1/system1/oemsvs_systemview1
  ...
  SVS_IRMCEBaseServerElementCapabilities=>/admin1/system1/capabilities1/hostsyscap1

Properties:
OEMSVSLastBootTime=20150210141728.000000+000
PrimaryOwnerContact=
OEMSVSServerUpdateStatus=0
Status=Error
RequestedState=0
PrimaryStatus=3
OEMSVSManagementIPAddress=172.17.122.111
OEMSVSLastPowerOnReason=29
OEMSVSLastPowerOffReason=29
HealthState=20
EnabledState=2
ElementName= unknown
Caption=RX300S8R4
OEMSVSAdminURLIPv4=http://172.17.122.111:80/
OEMSVSAdminURLDNS=http://iRMCFD6971.vlan743.qalab:80/
Name=YLNT000507
CreationClassName=SVS_IRMCEBaseServer
OtherDedicatedDescriptions={}
OtherIdentifyingInfo={03000200-0400-0500-0006-000700080009,YLNT000507}
IdentifyingDescriptions={CIM:GUID,CIM:Model:SerialNumber}
Dedicated={0}
OperationalStatus={2}

Verbs:
cd
exit
help
show
version
load
reset
start
stop

-> start
4. Getting CIM-based information

There exist several client approaches to access CIM-based information. In the following we present our most preferred.

<table>
<thead>
<tr>
<th>Client</th>
<th>Supported Protocol</th>
<th>OS</th>
</tr>
</thead>
<tbody>
<tr>
<td>WinRM</td>
<td>WMI (Microsofts implementation of the WS-Man Protocol)</td>
<td>Windows</td>
</tr>
<tr>
<td>Microsoft Power Shell CIM-cmdlets</td>
<td>WMI (Microsofts implementation of the WS-Man Protocol)</td>
<td>Windows</td>
</tr>
<tr>
<td>Wbemcli</td>
<td>CIM-XML</td>
<td>Linux</td>
</tr>
<tr>
<td>PyWbem</td>
<td>CIM-XML</td>
<td>Linux</td>
</tr>
</tbody>
</table>

4.1. WinRM

"Windows Remote Management (WinRM) is the Microsoft implementation of WS-Management Protocol, a standard Simple Object Access Protocol (SOAP)-based, firewall-friendly protocol that allows hardware and operating systems, from different vendors, to interoperate."²

4.1.1. Configure WinRM

WinRM must be installed and configured.

Some How-Tos:
- Details for "Installation and Configuration for Windows Remote Management" see e.g. the corresponding MSDN web site.

4.1.1.1. Client Configuration

Some very simple settings (Attention: lowering security settings!) must be done on the client side for trying first steps.

You need to open Windows command line (cmd) with administrator rights on your remote client!
The current used Windows user must have a password set.

```
winrm quickconfig
```

This command will do the first steps (start the WinRM service, if it was not started, configure Windows Firewall and other settings).

Additional setting must be done to enable a remote connection.

```
winrm set winrm/config/client @{TrustedHosts="*"}
winrm set winrm/config/client/Auth @{Basic="true"}
winrm set winrm/config/@{MaxEnvelopeSizekb="500"}
winrm set winrm/config/client @{AllowUnencrypted="true"}
```

Attention: some commands may need even bigger "MaxEnvelopeSizekb"!

4.1.1.2. Server Configuration

If using the in-band CIM-Provider you need to make some server-side configurations (Attention: lowering security settings!)

You need to open Windows command line (cmd) with administrator rights on your local server host!

```
winrm quickconfig
```

This command will do the first steps (start the WinRM service, if it was not started, configure Windows Firewall and other settings).

Additional settings must be done to enable a remote connection.

```
winrm set winrm/config/service/Auth @{Basic="true"}
winrm set winrm/config/service @{AllowUnencrypted="true"}
```

4.1.2. Use WinRM

First we present the standard command templates to access CIM-Based information based on the Microsoft command line shell CMD.

You should open CMD with admin rights.

4.1.2.1. WinRM command templates

The following templates describe the separate commands, for clarity the one line commands are formatted to several lines.

**Enumerate Instances**

```
winrm e [RESOURCEURI] [CLASSNAME]
(?__cimnamespace=[NAMESPACE])
-r:http://[HOSTIPADDRESS]:80/wsman
-a:basic
-u:[USERNAME]
-p:[PASSWORD]
-encoding:utf-8
```

**Get Instance**

```
winrm g "[RESOURCEURI] [CLASSNAME]
? [KEYPROPERTYNAME 1]= [KEYPROPERTYVALUE 1] +
 [KEYPROPERTYNAME 2]= [KEYPROPERTYVALUE 2] +
 ...
 [KEYPROPERTYNAME N]= [KEYPROPERTYVALUE N]
(+__cimnamespace=[NAMESPACE])"
-r:http://[HOSTIPADDRESS]:80/wsman
-a:basic
-u:[USERNAME]
-p:[PASSWORD]
-encoding:utf-8
```

Notes:
- The resource URI needs to be enclosed with quotes when the key values contain spaces in between.
- All key properties must be defined for uniquely identification of the instance.

**Set Instance**

```
winrm p "[RESOURCEURI] [CLASSNAME]
? [KEYPROPERTYNAME 1]= [KEYPROPERTYVALUE 1] +
 [KEYPROPERTYNAME 2]= [KEYPROPERTYVALUE 2] +
 ...
 [KEYPROPERTYNAME N]= [KEYPROPERTYVALUE N]
(+__cimnamespace=[NAMESPACE])"
-r:http://[HOSTIPADDRESS]:80/wsman
-a:basic
-u:[USERNAME]
-p:[PASSWORD]
-encoding:utf-8
@{[PROPERTYNAME]= [NEW PROPERTY VALUE]}
```

Notes:
- @ attaches property and new property name
- Only possible when intrinsic method set instance is supported for the property
- See SVS iRMC S4 Profile documents for supported properties.

**Create Instance**

Notes:
- Not supported by iRMC S4 implementation

**Delete Instance**

```
winrm d "[RESOURCEURI] [CLASSNAME]
? [KEYPROPERTYNAME 1]= [KEYPROPERTYVALUE 1] +
 [KEYPROPERTYNAME 2]= [KEYPROPERTYVALUE 2] +
 ...
 [KEYPROPERTYNAME N]= [KEYPROPERTYVALUE N]
```

Invoke Extrinsic Method

winrm i [EXTRINSIC METHOD NAME]
"[RESOURCEURI][CLASSNAME]
?([KEYPROPERTYNAME 1]=[KEYPROPERTYVALUE 1]+
 [KEYPROPERTYNAME 2]=[KEYPROPERTYVALUE 2]+
 ...+
 [KEYPROPERTYNAME N]=[KEYPROPERTYVALUE N]
(+__cimnamespace=[NAMESPACE])"
-r:http://[HOSTIPADDRESS]:80/wsman
-a:basic
-u:[USERNAME]
-p:[PASSWORD]
-encoding:utf-8

Notes:
- Works only on defined extrinsic methods: have a look to the profile documents

Identify

winrm id
-r:http://[HOSTIPADDRESS]:80/wsman
-a:basic
-u:[USERNAME]
-p:[PASSWORD]
-encoding:utf-8

Now we present some real live examples.

4.1.2.2. Enumerate Instances

winrm e http://schemas.ts.fujitsu.com/wbem/wscim/1/cim-schema/2/SVS_iRMCBaseServer?__cimnamespace=root/svs

SVS_iRMCAssociatedServer

ManagedElement

4.1.2.3. Enumerate Association instance

winrm e

SVS_iRMCAssociatedServerSettings

ManagedElement
4.1.4. Call extrinsic method
Reset the power on counter on the instance of SVS_iRMCServerConfigSettings

```
winrm i ResetPowerOnCounter
ResetPowerOnCounter_OUTPUT
ReturnValue = 0
```

4.1.2.5. Modify Instance
Change ConfErrorOffRestartTime property from "2" to "4".

```
winrm e
SVS_iRMCServerConfigSettings
BIOSRecoveryFlash = FALSE
Caption = Configuration Space Settings
ConfDisableFanTest = FALSE
ConfErrorOffRestartTime = 4
ConfFanDailyTestTime = 1380
ConfPowerCycleDelay = 7
ConfRebootRetryCounter = 3
ConfRetryCounterMax = 3
ConfTempCriticalPowerOff = FALSE
Description = Collection of important configuration space values for a PRIMERGY server
ElementName = Configuration Space Settings
InstanceID = SVS:iRMCServerConfigSettings
PowerOnCounter
Interval = P6DT20H50M0S
```

```
winrm p
SVS_iRMCServerConfigSettings
BIOSRecoveryFlash = FALSE
Caption = Configuration Space Settings
ConfDisableFanTest = FALSE
ConfErrorOffRestartTime = 4
ConfFanDailyTestTime = 1380
ConfPowerCycleDelay = 7
ConfRebootRetryCounter = 3
ConfRetryCounterMax = 3
ConfTempCriticalPowerOff = FALSE
Description = Collection of important configuration space values for a PRIMERGY server
ElementName = Configuration Space Settings
InstanceID = SVS:iRMCServerConfigSettings
PowerOnCounter
Interval = P6DT20H50M0S
```
4.2. Microsoft PowerShell CIM-cmdlets

Windows PowerShell is a task-based command-line shell and scripting language designed especially for system administration. Special CIM Cmdlets are available to access CIM-based management information.

4.2.1. Configure PowerShell


To check your PowerShell version you may use the following command.

```powershell
PSVersionTable.PSVersion
```

To adjust your Execution Policy for remote access you need to start the PowerShell as Administrator and perform the following command.

```powershell
Set-ExecutionPolicy RemoteSigned
```

Make sure the WinRM Client is working.

```powershell
winrm get winrm/config
```

If WinRM is not working it must be configured first (see section 4.1.1.).

If you want to use the PowerShell itself for the initial WinRM configuration there's a slightly different notation to use.

Client Configuration

```powershell
winrm set winrm/config/client '@{TrustedHosts="*"}'
winrm set winrm/config/client/Auth '@{Basic="true"}'
winrm set winrm/config '@{MaxEnvelopeSizekb="500"}'
winrm set winrm/config/client '@{AllowUnencrypted="true"}'
```

Attention: some commands may need even bigger "MaxEnvelopeSizekb"!

Server Configuration

```powershell
winrm set winrm/config/service/Auth '@{Basic="true"}'
winrm set winrm/config/service '@{AllowUnencrypted="true"}'
```

4.2.2. Use PowerShell


We present some first steps here.

For more detailed information some example PowerShell scripts are available at: [https://support.ts.fujitsu.com/Download/ShowDescription.asp?SoftwareGUID=F1218592-7DD9-4202-8BDD-F36BD7101565&Info=FTS&lng=COM](https://support.ts.fujitsu.com/Download/ShowDescription.asp?SoftwareGUID=F1218592-7DD9-4202-8BDD-F36BD7101565&Info=FTS&lng=COM)

4.2.2.1. Get Remote Access to iRMC S4

Create a remote session to access the iRMC S4 remotely without defining connection settings for each script.

Set credentials:

```powershell
$c = Get-Credential
```

Enter Username and Password.

Create CIM session:

```powershell
$s = New-CimSession -Authentication basic -Credential $c -ComputerName <IPAddress> -Port 80
```
Show session:

Get-CimSession

<table>
<thead>
<tr>
<th>Id</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>CimSession6</td>
</tr>
<tr>
<td>InstanceId</td>
<td>56c80b11-3688-4763-a8c3-f3398db96cd4</td>
</tr>
<tr>
<td>ComputerName</td>
<td>123.45.67.89</td>
</tr>
<tr>
<td>Protocol</td>
<td>WSMAN</td>
</tr>
</tbody>
</table>

4.2.2.2. Enumerate instances

Save SVS_iRMCBaseServer instances in variable $i

```powershell
```

Print instances.

```powershell
$i
```

<table>
<thead>
<tr>
<th>CreationClassName</th>
<th>SVS_iRMCBaseServer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dedicated</td>
<td>0</td>
</tr>
<tr>
<td>ElementName</td>
<td>Unknown.IPMI BMC DeviceID.4</td>
</tr>
<tr>
<td>EnabledState</td>
<td>2</td>
</tr>
<tr>
<td>HealthState</td>
<td>5</td>
</tr>
<tr>
<td>IdentifyingDescriptions</td>
<td>{CIM:GUID, CIM:Model:SerialNumber}</td>
</tr>
<tr>
<td>Name</td>
<td>Unknown.IPMI BMC DeviceID.4</td>
</tr>
<tr>
<td>OperationalStatus</td>
<td>2</td>
</tr>
<tr>
<td>OtherIdentifyingInfo</td>
<td>{fcaa0b1c-c699-ea47-beef-8fd983b3749, YLNT000007}</td>
</tr>
<tr>
<td>PrimaryStatus</td>
<td>1</td>
</tr>
<tr>
<td>RequestedState</td>
<td>0</td>
</tr>
<tr>
<td>Status</td>
<td>OK</td>
</tr>
<tr>
<td>PSComputerName</td>
<td>172.17.122.87</td>
</tr>
</tbody>
</table>

4.2.2.3. Enumerate instances

Save SVS_iRMCAssociatedServerSettings in variable $i.

```powershell
```

Print instances.

```powershell
$i
```

<table>
<thead>
<tr>
<th>ManagedElement</th>
<th>SettingData</th>
<th>PSComputerName</th>
</tr>
</thead>
<tbody>
<tr>
<td>SVS_iRMCBaseServer</td>
<td>(Name = &quot;YLNT000029&quot;, Creation... SVS_iRMCMConfigSettings (InstanceID = &quot;SVS:1...</td>
<td>172.17.167.208</td>
</tr>
</tbody>
</table>

4.2.2.4. Traverse via Associations

Save SVS_iRMCBaseServer instances in variable $i

```powershell
```

Get all instances that are associated with SVS_iRMCAssociatedServerSettings from SVS_iRMCBaseServer

We know only 1 instance of SVS_iRMCBaseServer is available.

```powershell
$a = Get-CimAssociatedInstance -InputObject $i -Association SVS_iRMCAssociatedServerSettings
```

Print instances.

```powershell
$a
```

<table>
<thead>
<tr>
<th>BIOSRecoveryFlash</th>
<th>FALSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>Caption</td>
<td>Configuration Space Settings</td>
</tr>
<tr>
<td>ConfDisableFanTest</td>
<td>FALSE</td>
</tr>
<tr>
<td>ConfErrorOffRestartTime</td>
<td>2</td>
</tr>
<tr>
<td>ConfFanDailyTestTime</td>
<td>1380</td>
</tr>
<tr>
<td>ConfPowerCycleDelay</td>
<td>7</td>
</tr>
</tbody>
</table>
4.2.2.5. Call extrinsic method

Get instance of SVS_iRMCServerConfigSettings.

```powershell
```

Call extrinsic method ResetPowerOnCounter() on instance of SVS_iRMCServerConfigSettings.

```powershell
Invoke-CimMethod -CimSession $s -InputObject $i[0] -MethodName ResetPowerOnCounter
```

Get instance of SVS_iRMCServerConfigSettings to check for the change.

```powershell
Print instance.
```

```powershell
$i
```

4.2.2.6. Modify Instance

Save SVS_iRMCChassis instance in $i

```powershell
```

We know only 1 instance is available

Change UserTracking property value to "NewName"

```powershell
Set-CimInstance -CimInstance $i -Property @{UserTracking="NewName"} -PassThru
```

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIOSRecoveryFlash</td>
<td>FALSE</td>
</tr>
<tr>
<td>Caption</td>
<td>Configuration Space Settings</td>
</tr>
<tr>
<td>ConfDisableFanTest</td>
<td>FALSE</td>
</tr>
<tr>
<td>ConfErrorAfterRestartTime</td>
<td>2</td>
</tr>
<tr>
<td>ConfFanDailyTestTime</td>
<td>1380</td>
</tr>
<tr>
<td>ConfPowerCycleDelay</td>
<td>7</td>
</tr>
<tr>
<td>ConfRebootRetryCounter</td>
<td>3</td>
</tr>
<tr>
<td>ConfRetryCounterMax</td>
<td>3</td>
</tr>
<tr>
<td>ConfTempCriticalBiosOff</td>
<td>FALSE</td>
</tr>
<tr>
<td>ConfTempCriticalPowerOff</td>
<td>Description: Collection of important configuration space values for a PRIMERGY server</td>
</tr>
</tbody>
</table>

```powershell
ElementName : Configuration Space Settings
InstanceID : SVS:iRMCServerConfigSettings
PowerOnCounter : 551.12:50:00
PSComputerName : 172.17.167.208

4.2.2.6. Modify Instance

Save SVS_iRMCChassis instance in $i

```powershell
```

We know only 1 instance is available

Change UserTracking property value to "NewName"

```powershell
Set-CimInstance -CimInstance $i -Property @{UserTracking="NewName"} -PassThru
```

<table>
<thead>
<tr>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIOSRecoveryFlash</td>
<td>FALSE</td>
</tr>
<tr>
<td>Caption</td>
<td>Configuration Space Settings</td>
</tr>
<tr>
<td>ConfDisableFanTest</td>
<td>FALSE</td>
</tr>
<tr>
<td>ConfErrorAfterRestartTime</td>
<td>2</td>
</tr>
<tr>
<td>ConfFanDailyTestTime</td>
<td>1380</td>
</tr>
<tr>
<td>ConfPowerCycleDelay</td>
<td>7</td>
</tr>
<tr>
<td>ConfRebootRetryCounter</td>
<td>3</td>
</tr>
<tr>
<td>ConfRetryCounterMax</td>
<td>3</td>
</tr>
<tr>
<td>ConfTempCriticalBiosOff</td>
<td>FALSE</td>
</tr>
<tr>
<td>ConfTempCriticalPowerOff</td>
<td>Description: Collection of important configuration space values for a PRIMERGY server</td>
</tr>
</tbody>
</table>

```powershell
ElementName : Configuration Space Settings
InstanceID : SVS:iRMCServerConfigSettings
PowerOnCounter : 00:00:00
PSComputerName : 172.17.167.208

```
4.3. Wbemcli

SBLIM (pronounced “sublime”), the Standards Based Linux Instrumentation for Manageability is an IBM-initiated Open Source project, intended to enhance the manageability of GNU/Linux systems. It does so by enabling WBEM, Web Based Enterprise Management.\(^3\) This opensource project comes up with different tooling for CIM. Especially a client to access CIM-based information is provided.

The WBEM Command Line Interface is a standalone, convenient systems management utility for CIMOM access. Invocation and output syntax are problem-oriented and well suited for interactively inclusion in Shell and Perl scripts. It is especially suited for administrators writing their own management scripts or for WBEM developers that want to test their providers.\(^4\)

Wbemcli is available for Linux OS.

4.3.1. Configure Wbemcli

Wbemcli client can be downloaded from sourceforge: [http://sourceforge.net/projects/sblim/](http://sourceforge.net/projects/sblim/)

4.3.2. Use Wbemcli

In the following some basic commands are described.

4.3.2.1. Enumerate Class Names available for a namespace

```
wbemcli ecn http://admin:admin@172.17.122.111:5988/root/svs -nl
```

4.3.2.2. Enumerate instances

```
wbemcli ei http://admin:admin@172.17.122.111:5988/root/svs:SVS_iRMCBaseServer -nl
```

4.3.2.3. Get a single instance with Key Properties

```
wbemcli gi "http://admin:admin@172.17.122.111:5988/root/svs:SVS_iRMCServerConfigSettings.InstanceID="SVS_iRMCServerConfigSettings"" -nl
```

4.3.2.4. Modify a specific instance

```
wbemcli mi "http://admin:admin@172.17.122.111:5988/root/svs:SVS_iRMCServerConfigSettings.InstanceID="SVS_iRMCServerConfigSettings"" 'ConfErrorOffRestartTime="4"' -nl
```

4.3.2.5. Get Associated instances

This also works without -ac option. -ac defines the association class that should be considered.

```
wbemcli ain 'http://admin:admin@172.17.122.111:5988/root/svs:SVS_iRMCServerConfigSettings.InstanceID="SVS_iRMCServerConfigSettings"''-ac SVS_iRMCAssociatedServerSettings -nl
```

4.3.2.6. Call extrinsic method

```
wbemcli cm 'http://localhost/root/ svs:SVS_iRMCServerConfigSettings.InstanceID="SVS_iRMCServerConfigSettings"'' RequestStateChange.RequestedState=2 ResetPowerOnCounter -nl
```

---


4.4. **PyWBEM**

PyWBEM is a pure-Python library for performing client operations using the WBEM CIM-XML protocol. PyWBEM can be downloaded on sourceforge: [http://sourceforge.net/projects/pywbem/files/](http://sourceforge.net/projects/pywbem/files/)

4.4.1. **Configure PyWBEM**

Installation and Tutorials are also available on sourceforge: [http://pywbem.sourceforge.net/documentation.shtml](http://pywbem.sourceforge.net/documentation.shtml)

4.4.2. **Use PyWBEM**

The following functions and aliases are supported:

- `ein` = EnumerateInstanceNames
- `ei` = EnumerateInstances
- `gi` = GetInstance
- `di` = DeleteInstance
- `mi` = ModifyInstance
- `ci` = CreateInstance
- `im` = InvokeMethod
- `an` = AssociateNames
- `ao` = Associations
- `rn` = ReferenceNames
- `re` = References
- `ecn` = EnumerateClassNames
- `ec` = EnumerateClasses
- `gc` = GetClass
- `dc` = DeleteClass
- `mc` = ModifyClass
- `cc` = CreateClass
- `eq` = EnumerateQualifiers
- `gq` = GetQualifier
- `sq` = SetQualifier
- `dq` = DeleteQualifier

4.4.2.1. **Get Remote Connection**

Usage: `wbemcli HOSTNAME [-u USER -p PASS] [-n NAMESPACE] [--no-ssl]`

```bash
root$ wbemcli 172.17.167.208 -u admin -p admin -n root/svs --no-ssl
Connected to http://172.17.167.208 as admin
```

4.4.2.2. **Get all supported class names**

```python
>>> i = ecn()
>>> print i
data = ['u'CIM_SettingsDefineCapabilities', 'u'CIM_MemberOfCollection', 'u'CIM_RedirectionService', ...
```

4.4.2.3. **Enumerate instances**

Store instances for SVS_iRMCBaseServer

```python
>>> instances = ei('SVS_iRMCBaseServer')
```

Get the first instance.

```python
>>> i = instances[0]
```

---

Print instance

```
>>> i.tomof()
u'instance of SVS_iRMCBaseServer {
	RequestedState = 0;
	Dedicated = {0};
	LastPowerOnReason = 29;
	HealthState = 20;
	LastBootTime = 20150225132916.000000+000;
	ManagementIPAddress = "172.17.167.208";
	PrimaryOwnerContact = "BlaBla";
	OtherIdentifyingInfo = ("03000200-0400-0500-0006-000700080009", "YLNT000029");
	IdentifyingDescriptions = ("CIM:GUID", "CIM:Model:SerialNumber");
	ServerUpdateStatus = 3;
	OperationalStatus = {2};
	AdminURLIPv4 = "http://172.17.167.208:80/";
	AdminURLDNS = "http://iRMCA2FA42.vlan575.qalab:80/";
	Status = "Error";
	EnabledState = 2;
	CreationClassName = "SVS_iRMCBaseServer";
	ElementName = "WIN-OHID5C6AOON";
	Name = "YLNT000029";
	UnitLocation = "Fasel";
	Caption = "RX300S8R4";
	OtherDedicatedDescriptions = {""};
	LastPowerOffReason = 29;
};
```

Print instance more human readable.

```
>>> print i.tomof()
instance of SVS_iRMCBaseServer {
    RequestedState = 0;
    Dedicated = {0};
    LastPowerOnReason = 29;
    HealthState = 20;
    LastBootTime = 20150225132916.000000+000;
    ManagementIPAddress = "172.17.167.208";
    PrimaryOwnerContact = "BlaBla";
    OtherIdentifyingInfo = ("03000200-0400-0500-0006-000700080009", "YLNT000029");
    IdentifyingDescriptions = ("CIM:GUID", "CIM:Model:SerialNumber");
    ServerUpdateStatus = 3;
    OperationalStatus = {2};
    AdminURLIPv4 = "http://172.17.167.208:80/";
    AdminURLDNS = "http://iRMCA2FA42.vlan575.qalab:80/";
    Status = "Error";
    EnabledState = 2;
    CreationClassName = "SVS_iRMCBaseServer";
    PrimaryStatus = 3;
    ElementName = "WIN-OHID5C6AOON";
    Name = "YLNT000029";
    UnitLocation = "Fasel";
    Caption = "RX300S8R4";
    OtherDedicatedDescriptions = {""};
    LastPowerOffReason = 29;
};
```

4.4.2.4. Traverse via Associations and get the specified instance

Get all instances for SVS_iRMCMonitorSettings

```
>>> scs = ei('SVS_iRMCMonitorSettings')
```

Get the only one instance from all instances.

```
>>> scs1 = scs[0]
```

Get all associators for the instance.

```
>>> associators = ao(scs1.path)
>>> associators
[CIMInstance(classname=u'SVS_iRMCCollection', ...), CIMInstance(classname=u'SVS_iRMCBaseServer', ...)]
```

We see instance 2 refers to SVS_iRMCBaseServer. Now let's get the associated instance.

```
>>> bs = gi(associators[1].path)
>>> print bs.tomof()
instance of SVS_iRMCBaseServer {
    RequestedState = 0;
    Dedicated = {0};
    LastPowerOnReason = 29;
    HealthState = 20;
    LastBootTime = 20150225132916.000000+000;
    ManagementIPAddress = "172.17.167.208";
    PrimaryOwnerContact = "BlaBla";
    OtherIdentifyingInfo = ("03000200-0400-0500-0006-000700080009", "YLNT000029");
```
IdentifyingDescriptions = {"CIM:GUID", "CIM:Model:SerialNumber"};
ServerUpdateStatus = 3;
OperationalStatus = {2};
AdminURLIPv4 = "http://172.17.167.208:80/";
AdminURLDNS = "http://IRMCA2FA42.vlan575.qalab:80/";
Status = "Error";
EnabledState = 2;
CreationClassName = "SVS_iRMCBaseServer";
PrimaryStatus = 3;
ElementName = "WIN-OHID5C6AOON";
Name = "YLTN000029";
UnitLocation = "Fasel";
Caption = "RX300S8R4";
OtherDedicatedDescriptions = {""};
LastPowerOffReason = 29;
};

4.4.2.5. Call extrinsic method
Get all instances; get a certain instance out of all.

```python
>>> scs = ei('SVS_iRMCServerConfigSettings')
>>> scs1 = scs[0]

Again, for testing purposes use gi in to get the specific instance.

```python
>>> css = gi(scs1.path)
```

Invoke method ResetPowerOnCounter on the instance of SVS_iRMCServerConfigSettings.

```python
>>> ret = im ('ResetPowerOnCounter', css.path)

Check the return code.

```python
>>> ret
(0L, {})```

Get the instance again and check the result

```python
>>> check = gi(css.path)
```

```python
instance of SVS_iRMCServerConfigSettings {
  ElementName = "Configuration Space Settings";
  PowerOnCounter = 00000000000000.000000:000;
  Caption = "Configuration Space Settings";
  ConfErrorOffRestartTime = 2;
  ConfRetryCounterMax = 3;
  Description = "Collection of important configuration space values for a PRIMERGY server";
  ConfPowerCycleDelay = 7;
  BIOSRecoveryFlash = False;
  ConfRebootRetryCounter = 3;
  ConfTempCriticalBiosOff = False;
  InstanceID = "SVS:iRMCServerConfigSettings";
  ConfDisableFanTest = False;
  ConfFanDailyTestTime = 1380;
};
```
Terms used

CIM (Common Information Model)

CIM provides a common definition of management information for systems, networks, applications and services, and allows for vendor extensions. CIM’s common definitions enable vendors to exchange semantically rich management information between systems throughout the network.

CIM Infrastructure specification

http://www.dmtf.org/sites/default/files/standards/documents/DSP0004V2.3_final.pdf
http://www.dmtf.org/sites/default/files/standards/documents/DSP0004_2.7.0.pdf

From the DMTF CIM FAQ:

CIM defines the data that is transported via the encoding and transport definitions of WBEM.

CIM Cmdlets in Windows PowerShell


DMTF

www.dmtf.org

DSP0226

Web Services for Management (WS-Management) Specification

http://www.dmtf.org/sites/default/files/standards/documents/DSP0226_1.0.0.pdf

SOAP Introduction

SOAP stands for Simple Object Access Protocol. SOAP is a protocol for accessing web services. SOAP is based on XML.

WBEM (Web-Based Enterprise Management)

Web-Based Enterprise Management (WBEM) is a set of management and Internet standard technologies developed to unify the management of distributed computing environments. WBEM provides the ability for the industry to deliver a well-integrated set of standard-based management tools, facilitating the exchange of data across otherwise disparate technologies and platforms.

http://www.dmtf.org/standards/wbem

Windows Remote Management (WinRM)


Windows Remote Management (WinRM) is the Windows implementation of WS-Management, an industry-standard Web services-based protocol. WinRM provides a secure, efficient way for management applications and scripts to communicate with local and remote computers. The Windows service that WinRM installs and uses is also named WinRM.

WS-Management Protocol