Cloud Infrastructure Management Interface (CIMI) Primer
DMTF is a not-for-profit association of industry members dedicated to promoting enterprise and systems management and interoperability. Members and non-members may reproduce DMTF specifications and documents, provided that correct attribution is given. As DMTF specifications may be revised from time to time, the particular version and release date should always be noted.

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patent owner or claimant, and shall have no liability or responsibility for costs or losses incurred if a standard is withdrawn or modified after publication, and shall be indemnified and held harmless by any party implementing the standard from any and all claims of infringement by a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

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

33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84

Foreword .............................................................................................................................. 5
Introduction ............................................................................................................................ 7
1 Creating a new Machine ................................................................................................. 9
1.1 Retrieve the CEP ........................................................................................................... 9
1.2 Retrieve the list of Machine Images ............................................................................. 9
1.3 Choose a Machine Image .......................................................................................... 10
1.4 Retrieve the list of Machine Configurations ............................................................... 10
1.5 Choose a Machine Configuration .............................................................................. 11
1.6 Create a new Credential Resource .......................................................................... 11
1.7 Create a new Machine .............................................................................................. 13
1.8 Query new Machine .................................................................................................. 13
1.9 Start a Machine ......................................................................................................... 14
1.10 Query a Machine ...................................................................................................... 14
1.11 Stop a Machine ......................................................................................................... 14
1.12 Update a Machine’s attributes ................................................................................. 15
2 Adding a New Volume to a Machine ........................................................................... 15
2.1 Obtain the Machine URL .......................................................................................... 15
2.2 Retrieve the CEP ........................................................................................................ 15
2.3 Get the list of VolumeConfigurations ....................................................................... 16
2.4 Choose a Volume Configuration ............................................................................... 16
2.5 Create a new Volume ............................................................................................... 16
2.6 Retrieve the Volume information .............................................................................. 17
2.7 Retrieve the Machine’s Volume collection ................................................................. 17
2.8 Connect the new Volume to a Machine ..................................................................... 19
2.9 Query the Machine’s volume collection .................................................................... 19
3 Defining and using Machine Templates ........................................................................ 20
3.1 Retrieve the CEP ........................................................................................................ 20
3.2 Create a new Machine Template ............................................................................... 20
3.3 Create a new Machine by using a Machine Template .............................................. 20
4 Creating a new Machine from an existing Volume ....................................................... 21
4.1 Retrieve the CEP ........................................................................................................ 21
4.2 Get the list of Volumes ............................................................................................. 22
4.3 Choose a Volume ....................................................................................................... 22
4.4 Create a new Machine ............................................................................................. 22
4.5 Query new Machine .................................................................................................. 23
5 Defining and using System Templates .......................................................................... 24
5.1 Retrieve the CEP ........................................................................................................ 24
5.2 Create a new System Template ................................................................................ 24
5.3 Create a new System by using a System Template ................................................. 25
5.4 Query the new System ............................................................................................. 26
6 Editing System Templates ............................................................................................ 27
6.1 Edit an existing System Template .............................................................................. 27
6.2 Create a new System using a System Template ....................................................... 29
6.3 Query the new System .............................................................................................. 30
7 Creating a Public Facing Network ................................................................................. 30
7.1 Retrieve the Cloud Entry Point (CEP) ....................................................................... 30
7.2 Verify provider examples for network configurations .............................................. 31
7.3 Retrieve a list of network port configurations .......................................................... 32
7.4 Creating a PUBLIC accessible network .................................................................... 32
7.5 Verify that a PUBLIC network has been created ...................................................... 33
<table>
<thead>
<tr>
<th>Page</th>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>85</td>
<td>7.6 Create a network port</td>
<td>34</td>
</tr>
<tr>
<td>86</td>
<td>7.7 Create a Machine attached to the public network</td>
<td>34</td>
</tr>
<tr>
<td>87</td>
<td>8 Provider responses and return values</td>
<td>36</td>
</tr>
<tr>
<td>88</td>
<td>8.1 Unrecognized attributes</td>
<td>36</td>
</tr>
<tr>
<td>89</td>
<td>8.2 Unreasonable requests</td>
<td>36</td>
</tr>
<tr>
<td>90</td>
<td>ANNEX A (informative) Change log</td>
<td>37</td>
</tr>
<tr>
<td>91</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Foreword

The Cloud Infrastructure Management Interface (CIMI) Primer (DSP2027) was prepared by the Cloud Management Working Group of the DMTF. This document contains scenarios that describe common uses of the CIMI protocol.

This specification has been developed as a result of joint work with many individuals and teams, including:

Dies Köper        Fujitsu (Editor)
Larry Lamers       VMware (Editor)

Contributors:

Hemal Shah         Broadcom
John Crandall      Brocade Communications Systems
Paul Lipton        CA Technologies
Efraim Moscovich   CA Technologies
Mark Carlson       DMTF Fellow
Marvin Waschke     DMTF Fellow
Steven Neely        Cisco
Shishir Pardikar   Citrix Systems Inc.
George Ericson     EMC
Norbert Floeren    Ericsson AB
Arturo Martin de Nicolas    Ericsson AB
Jacques Durand     Fujitsu
Kazunori Iwasa     Fujitsu
Jesus Molina       Fujitsu
Tom Rutt           Fujitsu
Maarten Wiggers    Fujitsu
Bryan Murray       Hewlett
Derek Coleman      Hewlett Packard Company
Robert Freund      Hitachi, Ltd.
Fred Maciel        Hitachi, Ltd.
Eric Wells         Hitachi, Ltd.
Zhexuan Song       Huawei
Jeff Wheeler       Huawei
Aaron Zhang        Huawei
HengLiang Zhang    Huawei
Doug Davis         IBM
Mark Johnson       IBM
Andreas Maier      IBM
Matthew Rutkowski  IBM
Keith Bankston     Microsoft Corporation
Nathan Burkhart    Microsoft Corporation
Josh Cohen         Microsoft Corporation
Yigal Edery        Microsoft Corporation
Colleen Evans      Microsoft Corporation
Krishnan Gopalan  Microsoft Corporation
John Parchem       Microsoft Corporation
Nihar Shah        Microsoft Corporation
Hiroshi Dempo      NEC Corporation
James Livingston   NEC Corporation
Ryuichi Ogawa      NEC Corporation
Steve Carter       Novell
Ashok Malhotra    Oracle
<table>
<thead>
<tr>
<th></th>
<th>Name</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>144</td>
<td>Gilbert Pilz</td>
<td>Oracle</td>
</tr>
<tr>
<td>145</td>
<td>Jack Yu</td>
<td>Oracle</td>
</tr>
<tr>
<td>146</td>
<td>Marios Andreou</td>
<td>Red Hat</td>
</tr>
<tr>
<td>147</td>
<td>David Lutterkort</td>
<td>Red Hat</td>
</tr>
<tr>
<td>148</td>
<td>Steve Winkler</td>
<td>SAP AG</td>
</tr>
<tr>
<td>149</td>
<td>Enrico Ronco</td>
<td>Telecom Italia</td>
</tr>
<tr>
<td>150</td>
<td>Federico Rossini</td>
<td>Telecom Italia</td>
</tr>
<tr>
<td>151</td>
<td>Fernando de la Iglesia</td>
<td>Telefónica</td>
</tr>
<tr>
<td>152</td>
<td>Fermín Galán</td>
<td>Telefónica</td>
</tr>
<tr>
<td>153</td>
<td>Miguel Peñalvo</td>
<td>Telefónica</td>
</tr>
<tr>
<td>154</td>
<td>Alvaro Polo</td>
<td>Telefónica</td>
</tr>
<tr>
<td>155</td>
<td>Alan Sill</td>
<td>Texas Tech University</td>
</tr>
<tr>
<td>156</td>
<td>Vince Lubsey</td>
<td>Virtustream Inc.</td>
</tr>
<tr>
<td>157</td>
<td>Winston Bumpus</td>
<td>VMware Inc.</td>
</tr>
<tr>
<td>158</td>
<td>Jim Davis</td>
<td>WBEM Solutions</td>
</tr>
<tr>
<td>159</td>
<td>Ghazanfar Ali</td>
<td>ZTE Corporation</td>
</tr>
<tr>
<td>160</td>
<td>Junsheng Chu</td>
<td>ZTE Corporation</td>
</tr>
<tr>
<td>161</td>
<td>Bhupin Khasnabish</td>
<td>ZTE Corporation</td>
</tr>
</tbody>
</table>
Introduction

For the sake of simplicity, in each of the following scenarios, the Cloud provider only supports the minimum features needed to demonstrate the features highlighted by each scenario. Therefore, the results of the query to the Cloud Entry Point (CEP) to retrieve the list of supported features are customized for each scenario. Additionally, the URI of the Cloud Entry Point is assumed to be http://example.com/CEP and all resources are assumed to be available in the example.com domain and accessible with the same protocol (HTTP). In the HTTP request-response examples in this document, the creation of the connection and HTTP headers not mandated by the CIMI specification are omitted for brevity.
Cloud Infrastructure Management Interface (CIMI) Primer

1 Creating a new Machine

This scenario creates a new Machine. The new Machine's configuration is based on existing configurations and images offered by the provider. However, a new Credential resource (userid and password) is created.

1.1 Retrieve the CEP

The CEP provides the links to the set of resources that are available in this Cloud. You retrieve the CEP to discover the URL to each collection:

GET /CEP HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/CloudEntryPoint",
  "id": "http://example.com/CEP",
  "baseURI": "http://example.com/",
  "resourceMetadata": { "href": "http://example.com/resourceMetadata" },
  "machines": { "href": "http://example.com/machines" },
  "machineConfigs": { "href": "http://example.com/machineConfigs" },
  "machineImages": { "href": "http://example.com/machineImages" },
  "credentials": { "href": "http://example.com/credentials" }
}
```

1.2 Retrieve the list of Machine Images

Before you can create a new Machine, first decide what kind of operating system or software you want to have preinstalled. The Machine Images collection is the set of Machine Images that this Cloud offers - note that some Machine Images may be predefined by the Cloud while some may be user created. The URI path comes from the data returned in the query to CEP for the `machineImages` key.

To retrieve the list of Machine Images, use the following syntax:

GET /machineImages HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineImageCollection",
  "id": "http://example.com/machineImages",
  "count": 3,
  "machineImages": [
    {
      "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineImage",
      "id": "http://example.com/images/WinXP-SP2",
      "name": "WinXP SP2",
      "description": "Windows XP with Service Pack 2",
      "created": "2012-01-01T12:00:00Z",
      "updated": "2012-01-01T12:00:00Z",
      "imageLocation": "http://example.com/data/8934322"
    },
    {
      "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineImage",
      "id": "http://example.com/images/Win7",
      "name": "Windows 7",
    }
  ]
}
```
1.3 Choose a Machine Image

Next examine each Machine Image to find one that meets your needs. The first one is acceptable, so it will be used later.

It is worth noting that if you knew you wanted to use the first item in the list and only wanted to see that one resource in the previous query, the following syntax could have been used instead:

```
GET /machineImages?first=1&last=1 HTTP/1.1
```

Note that you do not need to specify `first=1` in this case because "1" is its default value. The first machineImage is returned.

1.4 Retrieve the list of Machine Configurations

Next you decide onto what kind of virtual hardware you want to install your Machine Image. As with determining the kind of Machine Image you want, first ask for the list of available Machine Configurations:

```
GET /machineConfigs HTTP/1.1
```

Note that you do not need to specify `first=1` in this case because "1" is its default value. The first machineConfig is returned.
1.5 Choose a Machine Configuration

Next examine the returned list and select a Machine Configuration that suits your needs. The first one is acceptable, so it will be used later. It is identified by the id "http://example.com/machineConfigs/tiny".

1.6 Create a new Credential Resource

You want to use your own userName and password attributes for this new Machine, so you need to create a new Credential resource. This process is done by using the POST operation, but first you need to retrieve the Credential collection so that you know to where to POST a new Credential resource. To retrieve the Credential resource:

GET /credentials HTTP/1.1

The response is:

HTTP/1.1 200 OK
Content-Type: application/json

```json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/CredentialCollection",
```
Notice at this point that there are no Credential resources in the environment. Before you can create a new Credential resource, you must first discover this Cloud provider’s extension attributes for the Credential resource. Due default the CIMI specification does not define how the initial user of a new Machine is specified; rather it is left open for each Cloud provider to determine how this information should be provided. Clients can discover this information by querying the Credential resource metadata. To examine this resource, first look through the ResourceMetadata collection for this provider’s description of the Credential’s resource. Start by retrieving the ResourceMetadata collection from the URI referenced in the CEP:

GET /resourceMetadata HTTP/1.1
HTTP/1.1 200 OK
Content-Type: application/json

{  "resourceURI": "http://schemas.dmtf.org/cimi/1/ResourceMetadataCollection",
  "id": "http://example.com/resourceMetadata",
  "count": 1,
  "resourceMetadatas": [
    {  "resourceURI": "http://schemas.dmtf.org/cimi/1/ResourceMetadata",
      "id": "http://example.com/resources/Credential",
      "typeURI": "http://schemas.dmtf.org/cimi/1/Credential",
      "name": "Credential",
      "attributes": [
        {  "name": "userName", "namespace": "http://example.com",
          "type": "string", "required": "true" },
        {  "name": "password", "namespace": "http://example.com",
          "type": "string", "required": "true" }
      ]
    }
  ]
}

Now iterate over the list of resourceMetadata entries in the collection for the one whose "typeURI" is "http://schemas.dmtf.org/cimi/1/Credential". After you find it, you can now examine the extensions this provider has added to the Credential resource. The above indicates that the Credential resource has been extended and must include two attributes called "userID" and "password". Both are of type "string".

Now create a new Credential resource by using the POST operation:

POST /credentials HTTP/1.1
Content-Type: application/json

{  "resourceURI": "http://schemas.dmtf.org/cimi/1/CredentialCreate",
  "name": "Default",
  "description": "My Default User",
  "credentialTemplate": {
    "userName": "JoeSmith",
    "password": "letmein"
  }
}

HTTP/1.1 201 Created
Location: http://example.com/creds/12345

Note While the "userID" and "password" attributes were discovered via the Credential ResourceMetadata, the "name" and "description" attributes are part of the common set of attributes available on all resources. In a future scenario it is shown how the client knew that "userID" and "password" were the proper attribute names for this image type and Cloud provider.
1.7 Create a new Machine

Retrieve the Machines collection so that you know to where to POST a new Machine:

```
GET /machines HTTP/1.1
HTTP/1.1 200 OK
Content-Type: application/json

{  
  "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineCollection",
  "id": "http://example.com/machines",
  "count": 0,
  "operations": [ { "rel": "add", "href": "http://example.com/machines" } ]
}
```

If you only want to know the available operations, issue the following command.

```
GET /machines?$select=operations HTTP/1.1
HTTP/1.1 200 OK
Content-Type: application/json

{  
  "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineCollection",
  "operations": [ { "rel": "add", "href": "http://example.com/machines" } ]
}
```

Now create a new one:

```
POST /machines HTTP/1.1
Content-Type: application/json

{  
  "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineCreate",
  "name": "myMachine1",
  "description": "My very first machine",
  "machineTemplate": { 
    "machineConfig": { "href": "http://example.com/machineConfigs/tiny" },
    "machineImage": { "href": "http://example.com/images/WinXP-SP2" },
    "credential": { "href": "http://example.com/creds/12345" }
  }
}
```

HTTP/1.1 201 Created
Location: http://example.com/machines/843752

The response returns a unique machine reference "http://example.com/machines/843752" that is used in the following subclause.

1.8 Query new Machine

Retrieve the Machine to get the full representation of the new Machine:

```
GET /machines/843752 HTTP/1.1
HTTP/1.1 200 OK
Content-Type: application/json

{  
  "resourceURI": "http://schemas.dmtf.org/cimi/1/Machine",
  "id": "http://example.com/machines/843752",
  "name": "myMachine1",
  "description": "My very first machine",
  "created": "2012-08-15T12:15:00Z",
  "updated": "2012-08-15T12:15:00Z",
  "state": "STOPPED",
  "cpu": 1,
```
"memory": 4000000,
"disks": [ "href": "http://example.com/machines/843752/disks" ],
"networkInterfaces": [ "href": "http://example.com/machines/843752/NIs" ],
"operations": [ { "rel": "edit", "href": "http://example.com/machines/843752" },
{ "rel": "delete", "href": "http://example.com/machines/843752" },
{ "rel": "http://schemas.dmtf.org/cimi/1/action/start",
"href": "http://example.com/machines/843752" }
]
}

Notice the "state" attribute on the Machine is "STOPPED" because that is the initial state of a new machine.

1.9 Start a Machine

The presence of the "start" operation in the "operations" array of the Machine representation indicates not only the URI to which to POST the "start" operation, but that you are able to do it at this time.

POST /machines/843752 HTTP/1.1
Content-Type: application/json

{ "resourceURI": "http://schemas.dmtf.org/cimi/1/Action",
 "action": "http://schemas.dmtf.org/cimi/1/action/start"
}

HTTP/1.1 204 No Content

1.10 Query a Machine

Query the Machine again to verify that it is started:

GET /machines/843752 HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

{ "resourceURI": "http://schemas.dmtf.org/cimi/1/Machine",
 "id": "http://example.com/machines/843752",
 "name": "myMachine1",
 "description": "My very first machine",
 "created": "2012-08-15T12:15:00Z",
 "updated": "2012-08-15T12:15:00Z",
 "state": "STARTED",
 "cpu": 1,
 "memory": 4000000,
 "disks": [ "href": "http://example.com/machines/843752/disks" ],
 "networkInterfaces": [ "href": "http://example.com/machines/843752/NIs" ],
 "operations": [ { "rel": "edit", "href": "http://example.com/machines/843752" },
 { "rel": "delete", "href": "http://example.com/machines/843752" },
 { "rel": "http://schemas.dmtf.org/cimi/1/action/stop",
 "href": "http://example.com/machines/843752" }
 ]
}

Notice the "state" attribute on the Machine is "STARTED" and that the "operations" array no longer indicates that the "start" operation is available; but rather the "stop" operation is available now instead.

1.11 Stop a Machine

Using the "stop" operation's URL, you can now ask for the Machine to be stopped:
1.12 Update a Machine's attributes

Using a PUT operation on the "edit" operation's URL, you can update some of the attributes of the Machine, for example the "name" and "description":

```plaintext
PUT /machines/843752?$select=name,description HTTP/1.1
Content-Type: application/json

{ "resourceURI": "http://schemas.dmtf.org/cimi/1/Machine",
  "name": "Cool Demo #1"
}
```

HTTP/1.1 200 OK

```plaintext
{ "name": "Cool Demo #1" }
```

Notice that URL of the "edit" operation has been modified to indicate which attributes are being updated; only those attributes are touched. Because the URL includes the "description" attribute but the HTTP request body does not, that attribute is erased.

2 Adding a New Volume to a Machine

This scenario creates a new Volume and connects it to an existing Machine.

2.1 Obtain the Machine URL

Machine:

```
http://example.com/machines/843752
```

2.2 Retrieve the CEP

The CEP provides the links to the set of resources that are available in this Cloud. Retrieve the CEP to discover the URL to each collection:

```
GET /CEP HTTP/1.1
Content-Type: application/json

HTTP/1.1 200 OK

{ "resourceURI": "http://schemas.dmtf.org/cimi/1/CloudEntryPoint",
  "id": "http://example.com/CEP",
  "baseURI": "http://example.com/",
  "machines": { "href": "http://example.com/machines" },
  "machineConfigs": { "href": "http://example.com/machineConfigs" },
  "machineImages": { "href": "http://example.com/machineImages" },
  "credentials": { "href": "http://example.com/credentials" },
  "volumes": { "href": "http://example.com/volumes" },
  "volumeConfigs": { "href": "http://example.com/volumeConfigs" }
}
```
2.3 Get the list of VolumeConfigurations

When you create a new Volume, you need to decide what kind of Volume to create, e.g., its size, format, etc. The `volumeConfigurations` collection is the set of predefined Volume Configurations that this Cloud offers:

```json
GET /volumeConfigs HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/VolumeConfigurationCollection",
  "id": "http://example.com/volumeConfigs",
  "volumeConfigurations": [
    {
      "resourceURI": "http://schemas.dmtf.org/cimi/1/VolumeConfiguration",
      "id": "http://example.com/volumeConfigs/small",
      "name": "Small",
      "description": "A pretty small one",
      "created": "2012-08-15T12:15:00Z",
      "updated": "2012-08-15T12:15:00Z",
      "type": "http://schemas.dmtf.org/cimi/1/mapped",
      "format": "NTFS",
      "capacity": 60000000
    },
    {
      "resourceURI": "http://schemas.dmtf.org/cimi/1/VolumeConfiguration",
      "id": "http://example.com/volumeConfigs/medium",
      "name": "Medium",
      "description": "A medium sized one",
      "created": "2012-08-15T12:15:00Z",
      "updated": "2012-08-15T12:15:00Z",
      "type": "http://schemas.dmtf.org/cimi/1/mapped",
      "format": "NTFS",
      "capacity": 500000000
    },
    {
      "resourceURI": "http://schemas.dmtf.org/cimi/1/VolumeConfiguration",
      "id": "http://example.com/volumeConfigs/large",
      "name": "Large",
      "description": "A large one",
      "created": "2012-08-15T12:15:00Z",
      "updated": "2012-08-15T12:15:00Z",
      "type": "http://schemas.dmtf.org/cimi/1/mapped",
      "format": "NTFS",
      "capacity": 1000000000
    }
  ]
}
```

2.4 Choose a Volume Configuration

Next examine each Volume Configuration to find the one that meets your needs. The first one is acceptable, so it will be used later.

2.5 Create a new Volume

Retrieve the `Volumes` collection so that you know to where to POST a new Volume:

```http
GET /volumes HTTP/1.1

HTTP/1.1 200 OK
```
Content-Type: application/json

```json
{
    "resourceURI": "http://schemas.dmtf.org/cim/1/VolumeCollection",
    "id": "http://example.com/volumes",
    "operations": [
        { "rel": "add", "href": "http://example.com/volumes" }
    ]
}
```

Now create a new Volume:

```plaintext
POST /volumes HTTP/1.1
Content-Type: application/json

```json
{
    "resourceURI": "http://schemas.dmtf.org/cim/1/VolumeCreate",
    "name": "myVolume1",
    "description": "My first new volume",
    "volumeTemplate": {
        "volumeConfig": { "href": "http://example.com/volumeConfigs/small" }
    }
}
```

HTTP/1.1 201 Created

Location: http://example.com/volumes/35782

2.6 Retrieve the Volume information

To verify that the Volume you created and connected to the Machine is what you wanted, follow the reference that was returned from the previous step:

```plaintext
GET /volumes/35782 HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

```json
{
    "resourceURI": "http://schemas.dmtf.org/cim/1/Volume",
    "id": "http://example.com/volumes/35782",
    "name": "myVolume1",
    "created": "2012-09-15T12:15:00Z",
    "updated": "2012-09-15T12:15:00Z",
    "description": "My first new volume",
    "type": "http://schemas.dmtf.org/cimi/1/mapped",
    "capacity": 60000000,
    "operations": [
        { "rel": "edit", "href": "http://example.com/volumes/35782" },
        { "rel": "delete", "href": "http://example.com/volumes/35782" }
    ]
}
```

2.7 Retrieve the Machine's Volume collection

Before you can connect this new Volume to your Machine, you first need to retrieve the Machine's Volume collection so that you know to where to send your request. First retrieve the Machine to get the reference to the collection:

```plaintext
GET /machines/843752 HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/Machine",
    "id": "http://example.com/machines/843752",
    "name": "myMachine1",
    "description": "My very first machine",
    "created": "2012-08-15T12:15:00Z",
    "updated": "2012-08-15T12:15:00Z",
}
```
"state": "STARTED",
"cpu": 1,
"memory": 4000000,
"disks": { "href": "http://example.com/machines/843752/disks" },
"volumes": [ "href": "http://example.com/machines/843752/volumes" ],
"networkInterfaces": [ "href": "http://example.com/machines/843752/NIs" ],
"operations": [ {
  "rel": "edit", "href": "http://example.com/machines/843752" },
  { "rel": "delete", "href": "http://example.com/machines/843752" },
  { "rel": "http://schemas.dmtf.org/cimi/1/action/stop", "href": "http://example.com/machines/843752" }
]
}

Note that in the previous scenario, the "volumes" attribute was not present due to the limited scope of that scenario; however, now the "volumes" attribute appears because the scenario (and features of our sample provider) are expanded to include support for Volumes.

Now retrieve the Volume collection:
GET /machines/843752/volumes HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

{ "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineVolumeCollection",
  "id": "http://example.com/machines/843752/volumes",
  "operations": [ {
    "rel": "add", "href": "http://example.com/machines/843752/volumes" }
  ]
}

Note that there are no Volumes currently connected to this Machine.

Alternatively, as an optimization, this collection could have been retrieved at the same time as the original Machine by using the $expand query parameter:
GET /machines/843752?$expand=volumes HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

{ "resourceURI": "http://schemas.dmtf.org/cimi/1/Machine",
  "id": "http://example.com/machines/843752",
  "name": "myMachine1",
  "description": "My very first machine",
  "created": "2012-08-15T12:15:02Z",
  "updated": "2012-08-15T12:15:02Z",
  "state": "STARTED",
  "cpu": 1,
  "memory": 4000000,
  "disks": { "href": "http://example.com/machines/843752/disks" },
  "volumes": { "href": "http://example.com/machines/843752/volumes" },
  "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineVolumeCollection",
  "id": "http://example.com/machines/843752/volumes",
  "operations": [ {
    "rel": "add", "href": "http://example.com/machines/843752/volumes" }
  ]
},

"networkInterfaces": [ "href": "http://example.com/machines/843752/NIs" ],
"operations": [ {
  "rel": "edit", "href": "http://example.com/machines/843752" },
  { "rel": "delete", "href": "http://example.com/machines/843752" },
2.8 Connect the new Volume to a Machine

You connect the Volume to the Machine by using the "add" operation on the Volume collection and pass in a new MachineVolume resource:

```json
POST /machines/843752/volumes HTTP/1.1
Content-Type: application/json

{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineVolume",
    "initialLocation": "V",
    "volume": { "href": "http://example.com/volumes/35782" }
}
```

HTTP/1.1 201 Created
Content-Type: application/json

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineVolume",
    "id": "http://example.com/machines/843752/volumes/1",
    "initialLocation": "V",
    "volume": { "href": "http://example.com/volumes/35782" },
    "operations": [
        { "rel": "edit", "href": "http://example.com/machines/843752/volumes/1" },
        { "rel": "delete", "href": "http://example.com/machines/843752/volumes/1" }
    ]
}
```

2.9 Query the Machine's volume collection

Retrieve the Machine’s volume collection to get the complete list of Volumes and use the list to verify that the update was successful:

```http
GET /machines/843752/volumes HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

```

```json
{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineVolumeCollection",
    "id": "http://example.com/machines/843752/volumes",
    "machineVolumes": [
        { "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineVolumes",
        "id": "http://example.com/machines/843752/volumes/1",
        "initialLocation": "V",
        "volume": { "href": "http://example.com/volumes/35782" },
        "operations": [
            { "rel": "edit", "href": "http://example.com/machines/843752/volumes/1" },
            { "rel": "delete", "href": "http://example.com/machines/843752/volumes/1" }
        ]
    },
    "operations": [
        { "rel": "add", "href": "http://example.com/machines/843752/volumes" }
    ]
}
```
3 Defining and using Machine Templates

This scenario creates a new Machine Template that is used to create a new Machine. Machine Templates are convenience resources that allow for well-defined descriptions (configuration, image, etc.) of a Machine to be persisted such that it can be reused later. This feature is particularly useful when the user of the new Machine may not be technically savvy enough to know all of the details necessary to create the Machine. Commonly, Machine Templates are created for demos, or complex configurations, where a particular Machine Image must be used on a particular Machine Configuration. Machine Templates allow this information to be persisted and easily reused.

For convenience, reuse the configuration information already obtained in the previous scenarios.

3.1 Retrieve the CEP

The CEP provides the links to the set of resources that are available in this Cloud. Retrieve the CEP to discover the URL to each collection:

```
GET /CEP HTTP/1.1
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/CloudEntryPoint",
  "id": "http://example.com/CEP",
  "baseURI": "http://example.com/",
  "machines": { "href": "http://example.com/machines" },
  "machineTemplates": { "href": "http://example.com/machineTemplates" },
  "machineConfigs": { "href": "http://example.com/machineConfigs" },
  "machineImages": { "href": "http://example.com/machineImages" },
  "credentials": { "href": "http://example.com/credentials" }
}
```

3.2 Create a new Machine Template

From the previous scenarios, you already have the MachineConfiguration, MachineImage, and Credential resources that are reused for this MachineTemplate:

```
MachineConfiguration: http://example.com/machineConfigs/tiny
```

```
MachineImage: http://example.com/images/WinXP-SP2
```

```
Credential: http://example.com/creds/12345
```

Before you can create the new MachineTemplate, you first need to determine the URL to which the POST is sent. This location is obtained from the MachineTemplate collection URL that was returned as part of the CEP:

```
GET /machineTemplates HTTP/1.1
```
HTTP/1.1 200 OK
Content-Type: application/json

{
"resourceURI": "http://schemas.dmtf.org/cimi/1/MachineTemplateCollection",
"id": "http://example.com/machineTemplates",
"operations": [
{ "rel": "add", "href": "http://example.com/machineTemplates" }
]
}

Note that there are no MachineTemplates in the environment right now.

Now create the new MachineTemplate resource:

POST /machineTemplates HTTP/1.1
Content-Type: application/json

{
"resourceURI": "http://schemas.dmtf.org/cimi/1/MachineTemplate",
"name": "Demo1",
"description": "My first demo",
"machineConfig": { "href": "http://example.com/machineConfigs/tiny" },
"machineImage": { "href": "http://example.com/images/WinXP-SP2" },
"credential": { "href": "http://example.com/creds/12345" }
}

HTTP/1.1 201 Created
Location: http://example.com/machineTemplates/82754

3.3 Create a new Machine by using a Machine Template

Now create a new Machine by using this Machine Template:

POST /machines HTTP/1.1
Content-Type: application/json

{
"resourceURI": "http://schemas.dmtf.org/cimi/1/MachineCreate",
"name": "myMachine2",
"description": "My second machine",
"machineTemplate": { "href": "http://example.com/machineTemplates/82754" }
}

HTTP/1.1 201 Created
Location: http://example.com/machines/843799

4 Creating a new Machine from an existing Volume

This scenario creates a new Machine that boots from an existing Volume. This simple example assumes that the user knows that there is an existing Volume with the bootable property equal to true.

4.1 Retrieve the CEP

The CEP provides the links to the set of resources that are available in this Cloud. Retrieve the CEP to discover the URL to each collection:

GET /CEP HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

{ "resourceType": "http://schemas.dmtf.org/cimi/1/CloudEntryPoint",
"id": "http://example.com/CEP" }
4.2 Get the list of Volumes

When you create a new Machine from a Volume, you need to decide which Volume to use. The Volume collection is the set of existing Volumes that this Cloud offers:

```
GET /volumes HTTP/1.1
HTTP/1.1 200 OK
Content-Type: application/json
```

```
{
  "resourceType": "http://schemas.dmtf.org/cimi/1/VolumeCollection",
  "id": "http://example.com/volumes",
  "volumes": [
    {
      "resourceType": "http://schemas.dmtf.org/cimi/1/Volume",
      "id": "http://example.com/volumes/vol1",
      "name": "Win7-Bootable",
      "created": "2012-08-15T12:15:00Z",
      "updated": "2012-08-15T12:15:00Z",
      "description": "A bootable volume running Windows 7",
      "state": "AVAILABLE",
      "capacity": 60000000,
      "bootable": true
    },
    {
      "resourceType": "http://schemas.dmtf.org/cimi/1/Volume",
      "id": "http://example.com/volumes/vol2",
      "name": "Generic Volume",
      "created": "2012-08-15T12:15:00Z",
      "updated": "2012-08-15T12:15:00Z",
      "description": "A generic volume for Windows",
      "state": "AVAILABLE",
      "capacity": 60000000,
      "bootable": true
    }
  ]
}
```

4.3 Choose a Volume

Next examine each Volume to find the one that meets your needs. The first one is acceptable, so it will be used later.

4.4 Create a new Machine

Retrieve the Machines collection so you know to where to POST a new Machine:

```
GET /machines HTTP/1.1
HTTP/1.1 200 OK
Content-Type: application/json
```
Now create a new one, connecting it to the bootable Volume:

```
POST /machines HTTP/1.1
Content-Type: application/json

{
    "resourceType": "http://schemas.dmtf.org/cimi/1/MachineCreate",
    "name": "myMachine2",
    "description": "My second machine",
    "machineTemplate": {
        "volumes": [
            { "initialLocation": "V",
              "href": "http://example.com/volumes/voll" }
        ]
    }
}
```

Note that the MachineTemplate in this case does not specify a MachineImage or MachineConfiguration to use. In this example, for simplicity, you can assume that the provider has default values for those.

```
HTTP/1.1 201 Created
Location: http://example.com/machines/852108
```

### 4.5 Query new Machine

Retrieve the Machine to get the full representation of the new Machine:

```
GET /machines/852108 HTTP/1.1
```

```
HTTP/1.1 200 OK
Content-Type: application/json

{
    "resourceType": "http://schemas.dmtf.org/cimi/1/Machine",
    "id": "http://example.com/machines/852108",
    "name": "myMachine2",
    "description": "My second machine",
    "created": "2012-03-26T10:04:00Z",
    "updated": "2012-03-26T10:04:00Z",
    "state": "STOPPED",
    "cpu": "1",
    "memory": 4000000,
    "disks": { "href": "http://example.com/machines/852108/disks" },
    "volumes": { "href": "http://example.com/machines/852108/volumes" },
    "networkInterfaces": { "href": "http://example.com/machines/852108/NIs" },
    "operations": [
        { "rel": "edit", "href": "http://example.com/machines/852108" },
    ]
}
```
Notice the "state" attribute on the Machine is "STOPPED", which is the initial state of a new machine.

5  Defining and using System Templates

This scenario creates a new System Template that is used to create a new System. System Templates are convenience resources that allow for well-defined descriptions (configuration, image, etc.) of a System to be persisted such that it can be reused later.

5.1 Retrieve the CEP

The CEP provides the links to the set of resources that are available in this Cloud. Retrieve the CEP to discover the URL to each collection:

GET /CEP HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/CloudEntryPoint",
  "id": "http://example.com/CEP",
  "baseURI": "http://example.com/",
  "systemTemplates": { "href": "http://example.com/systemTemplates" },
  "machineTemplates": { "href": "http://example.com/machineTemplates" },
  "credentialTemplates": { "href": "http://example.com/credentialTemplates" },
  "volumeTemplates": { "href": "http://example.com/volumeTemplates" }
}

5.2 Create a new System Template

A SystemTemplate is defined so that when instantiated the result is a Machine is created, a Volume is connected to the Machine, and a Credential resource exists. To achieve this configuration, the following are included: a SystemTemplate definition, a MachineTemplate by value, a VolumeTemplate by reference, and a CredentialTemplate by reference. The VolumeTemplate and CredentialTemplate resources are already available:

VolumeTemplate:

http://example.com/volumeTemplates/95839

CredentialTemplate:

http://example.com/credentialTemplates/72000

Note  Alternatively, the VolumeTemplate and CredentialTemplate may be included by value in the MachineTemplate definition below. However, it is beneficial to immediately see in the SystemTemplate the resources that are involved and in general, automatic creation of the credential is more secure.

Before creating a SystemTemplate, the URL to which the POST is sent needs to be determined. This location is obtained from the SystemTemplate collection URL that was returned as part of the CEP.

GET /systemTemplates HTTP/1.1
HTTP/1.1 200 OK
Content-Type: application/json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/SystemTemplateCollection",
  "id": "http://example.com/systemTemplates",
  "operations": [
    { "rel": "add", "href": "http://example.com/systemTemplates" }
  ]
}

Now create the new System Template resource:
POST /systemTemplates HTTP/1.1
Content-Type: application/json
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/SystemTemplate",
  "name": "System Demo1",
  "description": "My first system template demo",
  "componentDescriptors": [
    { "name": "MyMachine",
      "type": "http://schemas.dmtf.org/cimi/1/Machine",
      "machineTemplate":
        { "name": "Machine in system demo",
          "description": "Machine in system",
          "machineConfig": { "href": "http://example.com/machineConfigs/tiny" },
          "machineImage": { "href": "http://example.com/images/WinXP-SP2" },
          "credential": { "href": "#MyCredential" },
          "volumes": [ [ "initialLocation": "/vol",
                      "href": "#MyVolume"
                    ] ]
        },
    { "name": "MyCredential",
      "type": "http://schemas.dmtf.org/cimi/1/Credential",
      "credentialTemplate":
        { "href": "http://example.com/credentialTemplates/72000" }
    },
    { "name": "MyVolume",
      "type": "http://schemas.dmtf.org/cimi/1/Volume",
      "volumeTemplate": { "href": "http://example.com/volumeTemplates/95839" }
    }
  ]
}
HTTP/1.1 201 Created
Location: http://example.com/systemTemplates/48920

5.3 Create a new System by using a System Template
Now create a new System by using this System Template:
POST /systems HTTP/1.1
Content-Type: application/json

```
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/SystemCreate",
  "name": "MySystem1",
  "description": "My first system",
  "systemTemplate": {
    "href": "http://example.com/systemTemplates/48920"
  }
}
```

HTTP/1.1 201 Created
Location: http://example.com/systems/78342

Note that, alternatively, the provider could have decided to return a reference to a Job resource instead of waiting until the System is completely created. Instead of the above 201 response, this type of request could have resulted in the following response:

```
HTTP/1.1 202 Accepted
CIMI-Job-URI: http://example.com/Jobs/90001

{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/Job",
  "id": "http://example.com/Jobs/90001",
  "name": "SystemCreationJob",
  "created": "2012-03-15T12:15:00Z",
  "updated": "2012-03-15T12:15:00Z",
  "targetResource": {
    "href": "http://example.com/systems",
    "affectedResources": {
      "href": "http://example.com/systems/110001",
      "action": "add",
      "status": "RUNNING",
      "progress": 30,
      "timeOfStatusChange": "2012-03-15T12:15:00Z",
      "isCancellable": "true",
      "nestedJobs": [
        {
          "href": "http://example.com/Jobs/90002"
        },
        {
          "href": "http://example.com/Jobs/90003"
        }
      ]
    }
  }
}
```

According to this response, the provider chose to create two nested Jobs and the "affectedResources" attribute includes a reference to the newly created System. Periodic retrieval of the Job's representation allows the Consumer to determine when the Job is completed, i.e., it is completed when the "progress" attribute has a value of 100.

### 5.4 Query the new System

Retrieve the System to get the full representation of the new System:

```
GET /systems/87342
```

HTTP/1.1 200 OK
Content-Type: application/json
6 Editing System Templates

In this scenario a second Machine is added to an existing System Template.

6.1 Edit an existing System Template

Edit the System Template created in a previous scenario and add another machine that shares its credential and volume resources:

SystemTemplate:

http://example.com/systemTemplates/48920

Retrieve the existing SystemTemplate definition:

GET /systemTemplates/48920 HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

{ "resourceURI": "http://schemas.dmtf.org/cimi/1/SystemTemplate",
  "name": "System Demo1",
  "description": "My first system template demo",
  "created": "2012-08-15T12:15:00Z",
  "updated": "2012-08-15T12:15:00Z",
  "componentDescriptors": [
    { "name": "MyMachine",
      "type": "http://schemas.dmtf.org/cimi/1/Machine",
      "machineTemplate": {
        "name": "Machine in system demo",
        "description": "Machine in system",
        "machineConfig": { "href": "http://example.com/machineConfigs/tiny" },
        "machineImage": { "href": "http://example.com/images/WinXP-SP2" },
        "credential": { "href": "/#MyCredential" },
        "volumes": [
          { "initialLocation": "/vol",
            "href": "/#MyVolume"
          }
        ]
      }
    }
  ]
}
Now update the SystemTemplate resource:

```
PUT /systemTemplates/48920 HTTP/1.1
Content-Type: application/json

{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/SystemTemplate",
  "name": "System Demo1",
  "description": "My first system template demo",
  "componentDescriptors": [
    {
      "name": "MyMachine",
      "type": "http://schemas.dmtf.org/cimi/1/Machine",
      "machineTemplate": {
        "name": "Machine in system demo",
        "description": "Machine in system",
        "machineConfig": { "href": "http://example.com/machineConfigs/tiny" },
        "machineImage": { "href": "http://example.com/images/WinXP-SP2" },
        "credential": { "href":="#MyCredential" },
        "volumes": [ {
          "initialLocation": "/vol",
          "href": "#MyVolume",
        } ],
        "quantity": 2
      },
    },
    {
      "name": "MyCredential",
      "type": "http://schemas.dmtf.org/cimi/1/Credential",
      "credentialTemplate": {
        "href": "http://example.com/credentialTemplates/72000" }
    },
    {
      "name": "MyVolume",
      "type": "http://schemas.dmtf.org/cimi/1/Volume",
      "volumeTemplate": { "href": "http://example.com/volumeTemplates/95839" }
    }
  ],
  "operations": [
    {
      "rel": "edit",
      "href": "http://example.com/systemTemplates/48920"
    }
  ]
}
```
Create a new System using a System Template

Now create a new System using this System Template by sending a POST to the same URL that you used in the previous scenario to create a new System:

```
POST /systems HTTP/1.1
```
Content-Type: application/json

```
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/SystemCreate",
  "name": "MySystem2",
  "description": "My second system",
  "systemTemplate": {
    "href": "http://example.com/systemTemplates/48920"
  }
}
```

HTTP/1.1 201 Created
Location: http://example.com/systems/78343

### 6.3 Query the new System

Retrieve the System to get the full representation of the new System:

```
GET /systems/87343
```

HTTP/1.1 200 OK

Content-Type: application/json

```
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/System",
  "id": "http://example.com/systems/78342",
  "name": "MySystem2",
  "description": "My second system",
  "created": "2012-10-15T12:15:00Z",
  "updated": "2012-10-15T12:15:00Z",
  "state": "STOPPED",
  "machines": {
    "href": "http://example.com/systems/87432/machines"
  },
  "credentials": {
    "href": "http://example.com/systems/87342/creds"
  },
  "volumes": {
    "href": "http://example.com/systems/87342/vols"
  },
  "operations": [
    {"rel": "edit", "href": "http://example.com/systems/78342"}
  ]
}
```

### 7 Creating a Public Facing Network

This clause is intended to provide guidance for basic network creation and deployment.

This scenario is to describe the actions required to create a Public (Internet Facing Network). Due to serialization, start with the network first unless the Provider offers a default network "starting point".

#### 7.1 Retrieve the Cloud Entry Point (CEP)

The CEP provides the links to the network resources that are available.

```
GET /CEP HTTP/1.1
```

HTTP/1.1 200 OK

Content-Type: application/json

```
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/CloudEntryPoint",
  "id": "http://example.com/CEP",
  "baseURI": "http://example.com/"
}
```
For additional details about beginning to use CIMI at the CEP, refer to the opening scenarios of this Primer.

### 7.2 Verify provider examples for network configurations

GET /networkConfigs HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

```json
{ "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkConfigurationCollection", "id": "http://example.com/networkConfigs", "count": 3, "networkConfigurations": [
  { "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkConfiguration", "id": "http://example.com/networkConfigs/InternetFacingNetwork", "name": "Public Access 1", "description": "internet reachable", "created": "2013-07-07T12:00:00Z", "updated": "2013-07-07T12:00:00Z", "networkType": "PUBLIC", "mtu": 1492, "classOfService": "GOLD" },
  { "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkConfiguration", "id": "http://example.com/networkConfigs/RED1", "name": "Private Red Network", "description": "Red Network", "created": "2013-07-07T12:00:00Z", "updated": "2013-07-07T12:00:00Z", "networkType": "PRIVATE", "mtu": 1500, "classOfService": "SILVER" },
  { "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkConfiguration", "id": "http://example.com/networkConfigs/BLUE2", "name": "Private Blue Network", "description": "Blue Network", "created": "2013-07-07T12:00:00Z", "updated": "2013-07-07T12:00:00Z", "classOfService": "SILVER" }
] }
```
Locate the Internet facing example network with a networkType "PUBLIC". Note the return information because this information is used later.

Various MTU sizes are often supported within an end-to-end network infrastructure. These MTU sizes vary due to encryption, protocol requirements, and equipment manufacturer limitations.

7.3 Retrieve a list of network port configurations

```plaintext
GET /networkPortConfigs HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkPortConfigurationCollection",
    "id": "http://example.com/networkPortConfigs",
    "count": 2,
    "networkPortConfigurations": [
        {
            "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkPortConfiguration",
            "id": "http://example.com/networkPortConfigs/GoldAccessPort",
            "name": "Gold Access Port",
            "description": "Enhanced Access Port",
            "created": "2013-07-07T12:00:00Z",
            "updated": "2013-07-07T12:00:00Z",
            "portType": "ACCESS",
            "classOfService": "GOLD"
        },
        {
            "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkPortConfiguration",
            "id": "http://example.com/networkPortConfigs/BaseAccessPort",
            "name": "Base Access Port",
            "description": "AccessPort",
            "created": "2013-07-07T12:00:00Z",
            "updated": "2013-07-07T12:00:00Z",
            "portType": "ACCESS",
            "classOfService": "BRONZE"
        }
    ]
}
```

Locate the network configuration with the classOfService attribute of GOLD. This network configuration is used in a later example.

7.4 Creating a PUBLIC accessible network

Retrieve the CEP network collection so that you know to where to POST a new network:
GET /networks HTTP/1.1
1369 HTTP/1.1 200 OK
1370 Content-Type: application/json
1371
1372 { "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkCollection",
1373   "id": "http://example.com/networks",
1374   "count": 0,
1375   "operations": [ { "rel": "add", "href": "http://example.com/networks" } ]
1376 }
1377
POST /networks HTTP/1.1
1378 Content-Type: application/json
1379
1380 { "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkCreate",
1381   "name": "PublicFacingNetwork",
1382   "description": "Public facing network",
1383   "networkTemplate": { "networkConfig":
1384     { "href": "http://example.com/networkConfigs/InternetFacingNetwork" } },
1385 }
1386
1387 HTTP/1.1 201 Created
1388 Location: http://example.com/networks/255111
1389
1390 7.5 Verify that a PUBLIC network has been created
1391
GET /networks/255111 HTTP/1.1
1392 HTTP/1.1 200 OK
1393 Content-Type: application/json
1394
1395 { "resourceURI": "http://schemas.dmtf.org/cimi/1/Network",
1396   "id": "http://example.com/networks/255111",
1397   "name": "PublicFacingNetwork",
1398   "description": "Public facing network",
1399   "created": "2013-07-07T12:15:00Z",
1400   "updated": "2013-07-07T12:15:00Z",
1401   "state": "STARTED",
1402   "networkType": "PUBLIC",
1403   "mtu": 1492,
1404   "classOfService": "GOLD",
1405   "networkPorts": { "href": "http://example.com/networks/255111/networkPorts" },
1406   "operations": [
1407     { "rel": "edit", "href": "http://example.com/networks/255111" },
1408     { "rel": "delete", "href": "http://example.com/networks/255111" },
1409     { "rel": "http://schemas.dmtf.org/cimi/1/action/start",
1410      "href": "http://example.com/networks/255111" }
1411   ]
1412 }
1413
1414 The networkPorts collection is an empty collection at this time.
### 7.6 Create a network port

Retrieve the CEP `networkPorts` collection so that you know to where to POST:

```
GET /networkPorts HTTP/1.1
```

HTTP/1.1 200 OK
Content-Type: application/json

```
{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkPortCollection",
    "id": "http://example.com/networkPorts",
    "count": 0,
    "operations": [
        { "rel": "add", "href": "http://example.com/networkPorts" }
    ]
}
```

Create a new network port. Note that the network must be indicated:

```
POST /networkPorts HTTP/1.1
Content-Type: application/json

{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/NetworkPortCreate",
    "name": "publicNetworkPort1",
    "description": "Port 1 of the public network",
    "networkPortTemplate": {
        "network": { "href": "http://example.com/networks/255111" },
        "networkPortConfig": { "href": "http://example.com/networkPortConfigs/GoldAccessPort" }
    }
}
```

HTTP/1.1 201 Created
Location: http://example.com/networkPorts/885412

### 7.7 Create a Machine attached to the public network

The previous sections in the Primer have illustrated how to create a machine. It is not the intent to duplicate that material but to present here those properties and attributes that should be queried or configured or both to connect a machine to a network by using CIMI.

Create a machine indicating a network to which to connect and a network port to use:

```
POST /machines HTTP/1.1
Content-Type: application/json

{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineCreate",
    "name": "myMachine1",
    "description": "My connected machine",
    "machineTemplate": {
        "machineConfig": { "href": "http://example.com/machineConfigs/tiny" },
        "machineImage": { "href": "http://example.com/images/WinXP-SP2" },
        "credential": { "href": "http://example.com/creds/12345" },
        "networkInterfaces": [
            { "network": { "href": "http://example.com/networks/255111" },
            "networkPort": { "href": "http://example.com/networkPorts/885412" }
        ]
    }
}
```

HTTP/1.1 201 Created
Location: http://example.com/machines/885412

Query the created machine and the `networkInterfaces` collection within it by using a `$expand` query parameter:
GET /machines/885412?$expand=networkInterfaces HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

```
{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/Machine",
    "id": "http://example.com/machines/885412",
    "name": "myMachine1",
    "description": "My connected machine",
    "created": "2012-08-15T15:00Z",
    "updated": "2012-08-15T15:00Z",
    "state": "STOPPED",
    "cpu": 1,
    "memory": 4000000,
    "disks": { "href": "http://example.com/machines/885412/disks"},
    "networkInterfaces": { "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineNetworkInterfaceCollection",
                           "id": "http://example.com/machines/885412/networkInterfaces",
                           "href": "http://example.com/machines/885412/networkInterfaces",
                           "count":1,
                           "networkInterfaces": [
                           { "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineNetworkInterface",
                            "id": "http://example.com/machines/885412/networkInterfaces/1",
                            "addresses": { "href": "http://example.com/machines/885412/networkInterfaces/1/addresses"},
                            "network": { "href": "http://example.com/networks/255111"},
                            "networkPort": { "href": "http://example.com/networkPorts/885412"},
                            "state": "PASSIVE"
                           }
                           ]
                           },
                           "operations": [
                           { "rel": "edit", "href": "http://example.com/machines/885412"},
                           { "rel": "delete", "href": "http://example.com/machines/885412"},
                           { "rel": "http://schemas.dmtf.org/cimi/1/action/start",
                            "href": "http://example.com/machines/885412" }
                           ]
}
```

Because we did not provide any address within the networkInterface section of the machine template when we created the machine, the provider must have allocated one. Query the address collection to verify it:

GET /machines/885412/networkInterfaces/1/addresses HTTP/1.1

HTTP/1.1 200 OK
Content-Type: application/json

```
{
    "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineNetworkInterfaceAddressCollection",
    "id": "http://example.com/machines/885412/networkInterfaces/1/addresses",
    "count": 1,
    "machineNetworkInterfaceAddresses": [
    { "resourceURI": "http://schemas.dmtf.org/cimi/1/MachineNetworkInterfaceAddress",
      "id": "http://example.com/machines/885412/networkInterfaces/1/addresses/344277",
      "name": "Address 1003",
      "description": "Public address",
      "address": { "href": "http://example.com/addresses/53234" } }
    ]
}
```
Details about the allocated address can be retrieved by querying the referenced address:

```
GET /addresses/53234 HTTP/1.1
```

HTTP/1.1 200 OK
Content-Type: application/json

```
{
  "resourceURI": "http://schemas.dmtf.org/cimi/1/Address",
  "id": "http://example.com/addresses/53234",
  "name": "192.0.2.240 (Public)",
  "description": "Public address",
  "ip": "192.0.2.240",
  "allocation": "dynamic",
  "protocol": "IPv4",
  "network": {"href": "http://example.com/networks/255111"},
  "resource": {"href": "http://example.com/machines/885412/networkInterfaces/1"}
}
```

This step provides a subnet structure that maps to real-world use of Layer 3 Services. Among other options, the Address can be provided by the provider or consumer. This option varies with each vendor, provider, implementation, and network architecture in use.

## 8 Provider responses and return values

### 8.1 Unrecognized attributes

A syntax error in the filter expression results in an error being generated. The provider returns a 400 ‘Bad Request’ response to a query with an unrecognized attribute name used in the `$filter` query parameter.

For example:

```
GET /machines?$filter=aaa='bbb' HTTP/1.1
```

The Machine resource has no "aaa" attribute: The recommended action is to return a 400 ‘Bad Request’ because the consumer did not follow `$filter`'s syntax, i.e., "aaa" is not a "resource attribute name".

### 8.2 Unreasonable requests

CIMI clients are expected to make reasonable requests. CIMI providers are not required to advertise maximum attributes for all resources. A CIMI provider may set limits on the length of attribute values it finds reasonable. It may reject a request it deems unreasonable. This is common practice in web-based protocols today.

These limits may not all be advertised in the ResourceMetadata, although it is recommended that CIMI providers do so. A CIMI provider that receives a request that exceeds any of these limits, returns a response with an appropriate standard HTTP status code, e.g., HTTP return code 413 ‘Request Entity Too Large’.
ANNEX A
(informative)

Change log

<table>
<thead>
<tr>
<th>Version</th>
<th>Date</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0.0</td>
<td>2012-08-28</td>
<td></td>
</tr>
<tr>
<td>1.0.1</td>
<td>2012-09-12</td>
<td>Errata</td>
</tr>
<tr>
<td>1.1.0</td>
<td>2014-07-31</td>
<td>DMTF Informational release</td>
</tr>
</tbody>
</table>