**Plasma controller API Specification**

**Author :**

**Chester Kuo <** [**chester.kuo@intel.com**](mailto:chester.kuo@intel.com) **>**

**Ananth Narayan S <** [**ananth.s.narayan@intel.com**](mailto:ananth.s.narayan@intel.com) **>**

Contents

[Revision History 2](#_Toc446590860)

[Introduction 3](#_Introduction)

[Scope 3](#_Toc446590862)

[Terminology 3](#_Toc446590863)

[References 3](#_Toc446590864)

[API 4](#_API)

[Service Root 4](#_Toc446590867)

### Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| 0.1 | First draft | 6.17. 2016 | Chester Kuo |
| 0.2 | Added APIs for storage management, Flavor creation.  Added Introduction paragraph, update storage management for attachment/de-attachment | 6.22.2016 | Ananth Narayan S,  Chester Kuo |
| 0.3 | Revised mockup API data | 7.26.2016 | Chester Kuo |
| 0.4 | Updated details of flavor creator |  | Ananth Narayan S |

# Introduction

Plasma is a controller for Pooled and composable resources which adhere to DMTF Redfish with hardware management requests. That includes provisioning and management of RSD components and underlay features.

## Plasma and OpenStack

Current OpenStack components such as Nova, Neutron, Cinder, Ironic need to be modified to call Plasma APIs to provide the requested RSD feature. For example if a particular flavor system is requested which maps to a RSD system, such a call should go to Ironic which subsequently Plasma to provide that node. So Plasma will get subsumed under Ironic in the future.



## Scope

This document contains information about the Restful API of Plasma controller including networking, storage configuration and compute nodes composition and flavor creation..etc.

## Terminology

|  |  |
| --- | --- |
| Term | Definition |
| BMC | Baseboard Management Controller |
| CIMI | Cloud Infrastructure Management Interface |
| HTTP | Hypertext Transfer Protocol |
| JSON | JavaScript Object Notation |
| NIC | Network Interface Card |
| OCCI | Open Cloud Computing Interface |
| OData | Open Data Protocol |
| OVF | Open Virtualization Format |
| POD | A physical collection of multiple racks |
| PODM | POD Manager |
| PSME | Pooled System Management Engine |
| REST | Representational state transfer |
| SDV | Software Development Vehicle |
| URI | Uniform resource identifier |
| UUID | Universally Unique Identifier |
| XML | eXtensible Markup Language |

## References

|  |  |  |
| --- | --- | --- |
| Document name |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# **API**

Notes:

1. During the discovery phase, plasma queries the rack to obtain a list of entities and caches the data.
2. The plugin selects a refresh interval to update its cache. This is not exposed in the API.

## Service Root

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | List API verion | In | Type | Description |
| URI | / |  |  |  |
|  |  |  |  |  |
|  | versions | Body | Array | A list of version objects that describe API version available |
|  | min\_version | Body | String | If this version of the API supports microversions, the minimum microversion that is supported. This will be the empty string if microversions are not supported. |
|  | version | Body | String | Versioning of this API response |
|  | id | Body | String | A Major API version |
|  | links | Body | Array | A List of relative links. This allows a client to easily obtain rather than construct resource URIs. The following types of link relations are associated with resources.   * A self link contains a versioned link to the resource. Use these links when the link is followed immediately. * A bookmark link provides a permanent link to a resource that is appropriate for long term storage. |
|  | status | Body | String | The status of this API version, this can be one of : CURRENT : this is the preferred version of the API to use. SUPPORTED : this is an older , but still supported version of the API DEPRECATED : a deprecated version of the API that be removed. |
|  |  |  |  |  |

Operations

#### **GET**

**GET /**

#### Response

Normal Response codes: 200

**{**

**"name" : "OpenStack Plasma API",**

**"description" : "Plasma is an OpenStack project which aims to provide node composition based on redfish API.",**

**"default\_version" : {**

**"status" : "CURRENT",**

**"version" : "1.1",**

**"links" : [**

**{**

**"rel" : "self",**

**"href" : "http://openstack.example.com:8881/v1/"**

**}**

**],**

**"id" : "v1",**

**"min\_version" : "1.0"**

**},**

**"versions" : [**

**{**

**"status" : "CURRENT",**

**"links" : [**

**{**

**"href" : "http://openstack.example.com:8881/v1/",**

**"rel" : "self"**

**}**

**],**

**"id" : "v1",**

**"version" : "1.1",**

**"min\_version" : "1.0"**

**}**

**]**

**}**

## API version, get details of a specific API

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Showing v1 API | In | Type | Description |
| URI | /v1 |  |  |  |
| Property |  |  |  |  |
|  |  |  |  |  |
|  | id | Body | String | Major API version |
|  | links | Body | Array | Links to the resources. |
|  |  |  |  |  |

Normal Response codes: 200

### Operations

#### **GET**

**GET /v1**

#### Response

**{**

**"id" : "v1",**

**"links" : [**

**{**

**"href" : "http://openstack.example.com:8881/v1/",**

**"rel" : "self"**

**},**

**{**

**"rel" : "describedby",**

**"type" : "text/html",**

**"href" : "http://docs.openstack.org/developer/plasma/dev/api-spec-v1.html"**

**}**

**],**

**"nodes" : [**

**{**

**"rel" : "self",**

**"href" : "http://openstack.example.com:8881/v1/nodes/"**

**},**

**{**

**"rel" : "bookmark",**

**"href" : "http://openstack.example.com:8881/nodes/"**

**}**

**],**

**"storages" : [**

**{**

**"href" : "http://openstack.example.com:8881/v1/storages/",**

**"rel" : "self"**

**},**

**{**

**"rel" : "bookmark",**

**"href" : "http://openstack.example.com:8881/storages/"**

**}**

**],**

**"flavors" : [**

**{**

**"href" : "http://openstack.example.com:8881/v1/flavors/",**

**"rel" : "self"**

**},**

**{**

**"rel" : "bookmark",**

**"href" : "http://openstack.example.com:8881/flavors/"**

**}**

**],**

**"media\_types" : [**

**{**

**"type" : "application/vnd.openstack.plasma.v1+json",**

**"base" : "application/json"**

**}**

**]**

**}**

All API calls through the rest of this document require authentication with the OpenStack Identity service. They also required a base service url that is extracted from the Identity token of type plasma. This will be the root url that every call below will be added to build a full path.

[http://developer.openstack.org/api-guide/quick-start/api-quick-start.html]

## **Composed Nodes collections**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Node Collection | In | Type | Description |
| URI | /nodes |  |  |  |
|  |  |  |  |  |
|  | nodestate (optional) | Query | String | Filter the list of return nodes , and only return those with the specificed “nodestate” |
|  | field (optional) | Query | String | one or more fields to be returned in the response |
|  |  |  |  |  |
|  | links | Body | String | A list of relative links |
|  | id | Body | String | UUID of nodes |
|  | nodestate | Body | String | The current composed node state of this node. Allocating,  Allocated,  Assembling,  PoweredOn,  PoweredOff,  Failed |

Return a list of nodes with some information about each node, some filtering is possible by passing in with request.

### Operations

Normal response codes: 200

Error response codes: badRequest(400), unauthorized(401), forbidden(403)

#### **GET**

**GET /nodes**

#### Response

**{**

**"nodes" : [**

**{**

**"id" : "ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0",**

**"name" : "Server 1" ,**

**"nodestate" : "PoweredOn" ,**

**"links": [**

**{**

**"rel" : "self",**

**"href" : "https://openstack.example.com/v1/nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0"**

**},**

**{**

**"href" : "https://openstack.example.com/nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0",**

**"rel" : "bookmark"**

**}**

**]**

**},**

**{**

**"id" : "4d8c3732-a248-40ed-bebc-539a6ffd25c0" ,**

**"name" : "Server 2",**

**"nodestate" : "PoweredOff" ,**

**"links" : [**

**{**

**"ref" : "self",**

**"href" : "https://openstack.example.com/v1/nodes/4d8c3732-a248-40ed-bebc-539a6ffd25c0"**

**},**

**{**

**"ref" : "bookmark",**

**"href" : "https://openstack.example.com/nodes/4d8c3732-a248-40ed-bebc-539a6ffd25c0"**

**}**

**]**

**}**

**]**

**}**

Filtering nodes

Filter criterion are provided as part of the HTTP GET URI. The number of search criteria that can be provided is limited to the length of the HTTP GET request. Only exact match filters are supported and filtering can be based on any (non nested) key of

|  |  |  |
| --- | --- | --- |
| URI | /v1/nodes?name1=value1&name2=value2 |  |
|  |  |  |
| Example | /v1/nodes?ram=10 | Filter in hosts that have 10GB RAM only. |
|  | /v1/nodes?ram=10&nodestate=PoweredOn | Filter in hosts that have 10GB Ram and whose nodestate is ‘PoweredON’ |
|  |  |  |

**GET /nodes?nodestate=PoweredOn**

#### Response

**{**

**"nodes": [{**

**"id": "eeeecccc-dddd-ffff-aaaa-uuuukkkk",**

**"nodestate": "PoweredOn",**

**"Links": [{**

**"rel": "self",**

**"href": "https://openstack.example.com/v1/nodes/eeeecccc-dddd-ffff-aaaa-uuuukkkk"**

**}, {**

**"href": "https://openstack.example.com/v1/nodes/eeeecccc-dddd-ffff-aaaa-uuuukkkk",**

**"rel": "bookmark"**

**}],**

**"name": "Server 1"**

**}]**

**}**

#### **POST** Create a new composed node with specific resource or empty request.

When you create a server, the response shows only the server ID, its links. You can get additional attributes through subsequent GET requests on the server’s id.

Normal response codes: 200

Error response codes: 401, 403, 404

#### 

**POST /nodes**

Content-Type: application/json

{

}

#### Response

**{**

**"node" :**

**{**

**"id" : "eeeecccc-dddd-ffff-aaaa-uuuukkkk",**

**"Links" : [**

**{**

**"rel" : "self",**

**"href" : "https://openstack.example.com/v1/odes/eeeecccc-dddd-ffff-aaaa-uuuukkkk"**

**},**

**{**

**"href" : "https://openstack.example.com/v1/nodes/eeeecccc-dddd-ffff-aaaa-uuuukkkk",**

**"rel" : "bookmark"**

**}**

**]**

**}**

**}**

## **Composed Node details**

Return details of single node

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Node details | In | Type | Description |
| URI | /nodes/{node\_ident} |  |  |  |
|  |  |  |  |  |
|  | field (optional) | Query | String | Fields to be returned in the response.  The following request return only the uuid and power\_state fields for each composed node.  Ex: GET /v1/nodes?field=id,name |
|  |  |  |  |  |
|  | id | Body | String | UUID of the resources |
|  | nodestate | Body | String | Current Composed node state, |
|  | boot\_source | Body | String | Current booting source target,  None, Pxe, Localdisk |
|  | pending\_boot\_source | Body | String | Pending booting source target |
|  | pooling\_group\_id | Body | String | Pooling resource group id |
|  | health\_status | Body | String | Health status,  OK, Warning, Critical |
|  | name | Body | String | Human-readable identifier for compose node |
|  | metadata | Body | String | Compute node Metadata ,  Host NIC MAC address, BMC MAC address, RackID, compute node ID, Serial No. This is used for metadata server of deployment. |
|  | node\_property | Body | String | CPU, memory , and storage asset info |
|  | links | Body | String |  |
|  | created\_at updated\_at | Body | String | TimeStamp for compose node creation. |
|  | provision\_state | Body | String | Node Provision state set by deployment tool, like Puppet, Ansible…etc |

### Operations

#### **GET**

Normal response codes: 200

Error response codes: 401, 403, 404

**GET /nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0**

#### Response

**{**

**"node" : {**

**"id" : "ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0",**

**"nodestate" : "Off",**

**"boot\_source" : "Localdisk",**

**"pending\_boot\_source" : "PXE",**

**"pooling\_group\_id" : "11z23344-0099-7766-5544-33225511",**

**"health\_status" : "OK",**

**"name" : null,**

**"metadata" : {**

**"nic" : [**

**{"mac" : "f1:12:44:55:66:77"},**

**{"mac" : "f2:44:44:44:44:88"}**

**],**

**"mgmt\_mac" : "00:AA:BB:CC:DD:EE",**

**"podid" : "POD1",**

**"rackid" : "Rack2",**

**"slotid" : "3",**

**"board\_serialno" : "2M220100SL"**

**},**

**"node\_properties" : {**

**"cpu\_arch" : "x86\_64",**

**"cpu\_count" : "2",**

**"memory\_size\_gb" : "32",**

**"network" : [**

**{**

**"type" : "ethernet",**

**"speed" : "40000000"**

**}**

**],**

**"memory\_type" : "DDR4",**

**"storage" : [**

**{**

**"type" : "SSD",**

**"volume\_gb" : "40"**

**}**

**]**

**},**

**"created\_at" : "2016-04-20T15:40:00+00:00",**

**"updated\_at" : "2016-04-20T15:40:00+00:00",**

**"links": [**

**{**

**"rel" : "self",**

**"href" : "https://openstack.example.com/v1/nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0"**

**},**

**{**

**"rel" : "boomark",**

**"href" : "https://openstack.example.com/nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0"**

**}**

**]**

**}**

**}**

#### **DELETE** to Release a composed node

Normal response codes: 204

Error response codes: 401,403, 404 ,409

#### 

**DELETE /nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0**

#### Response

**This is no body content for the response of a successful DELETE query**

#### **PUT** Update the attributes of composed node.

Normal response codes: 200

Error response codes: 400, 401, 403 ,404

#### 

**PUT /nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0**

{

"node": {

"name" : "new-server-name"

}

}

## **Storage volume management for node**

The following API is used to manage storage volume attached to a node.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name |  | In | Type | Description |
| URI | /nodes/{node\_id}/storages |  |  |  |
| id | Input parameter | URL | String | UUID of the node |
|  |  |  |  |  |
| storagevolumeAttachments |  | Body | Array | List of the storage volume attachments |
| device |  | Body | String | Name of the device such as , /dev/sdd |
| id |  | Body | String | The UUID of the attachment |
| serverId |  | Body | String | The UUID of the server |
| volumeId |  | Body | String | The UUID of the attached storage volume |

### Operations

#### **GET**

List storage devices attached to the specified node.

Normal response codes: 200

Error response codes: 401, 403 ,404

**GET /nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0/storages**

#### Response

**{**

**"storagevolumeAttachments": [**

**{**

**"device": "/dev/sdd",**

**"id": "a26887c6-c47b-4654-abb5-dfadf7d3f803",**

**"serverId": "ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0",**

**"volumeId": "a26887c6-c47b-4654-abb5-dfadf7d3f803"**

**},**

**{**

**"device": "/dev/sdc",**

**"id": "a26887c6-c47b-4654-abb5-dfadf7d3f804",**

**"serverId": "ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0",**

**"volumeId": "a26887c6-c47b-4654-abb5-dfadf7d3f804"**

**}**

**]**

**}**

#### **POST**

Attach a storage volume to the node.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name |  | In | Type | Description |
| Request |  |  |  |  |
|  |  |  |  |  |
| storagevolumeAttachments |  | Body | String | A Dictionary representation of a storage volume attachment. |
| volumeId |  | Body | String | The UUID of the volume to attach. |
| deviceId |  | Body | String | The UUID of storage device, this mapping to Drive.xml schema |
| id |  | Body | String | The UUID of the attachment |
|  |  |  |  |  |

Normal response codes: 200

Error response codes: 400, 401, 403,404, 409

**POST /nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0/storages**

Content-Type: application/json

{

"storagevolumeAttachment": {

"deviceId": "a26887c6-c47b-4654-abb5-dfadf7d3f803"

}

}

#### Response

{

"storagevolumeAttachment": {

"id": "116887c6-c47b-4654-abb5-dfadf7d44444",

"volumeId": "1226887c-c47b-5446-abb5-33adf7d3f803",

"deviceId": "a26887c6-c47b-4654-abb5-dfadf7d3f803"

}

}

#### Error Response

{

"status": 400,

"error": {

"message": "Error message"

}

}

#### **DELETE**

Detach an attached storage volume from the node.

Request : attachment id

Normal response codes: 202

Error response codes: 400,401, 403, 404, 409

**DELETE /nodes/ee1ecc3c-d3dd-f4ff-a6aa-uu7uk9k0/storages/116887c6-c47b-4654-abb5-dfadf7d44444**

#### Success Response

**This is no body content for the response of a successful DELETE query**

#### Error Response

{

"status": 404,

"error": {

"message": "Error message"

}

}

# **Storage**

This section lists Plasma API to support management of RSD pooling storage such as NVMe or SSD drive. OpenStack managed storage is out of scope of this document – it is handled by OpenStack components(Cinder)/Ceph etc.

## **Storage Device Collection**

This call lists the enumerated storage devices.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name |  | In | Type | Description |
| URI | /storages |  |  |  |
|  | filter (optional) | Query | String | Using filter to return node based on criteria |
|  | deviceId | Body | String | The UUID of storage device, this mapping to Drive.xml schema |
|  | pooling\_groud\_id | Body | String | Identifier of Switch (switch cluster) devices |
|  | allocate\_status | Body | String | Status of storage device,  Status : Allocated, Available, Erasing |
|  |  |  |  |  |

### Operations

#### **GET**

List all storage devices

Normal response codes: 200

Error response codes: 401, 403, 404

**GET /storages**

#### Response

**{**

**"storges" : [**

**{**

**"**deviceId**" : "bbfddf09-4d7e-40d5-88a9-8acfb2f88c21",**

**"pooling\_group\_id" : "11z23344-0099-7766-5544-33225511",**

**"allocate\_status" : "allocated",**

**"links" : [**

**{**

**"ref" : "self",**

**"href" : "https://openstack.example.com/v1/storages/bbfddf09-4d7e-40d5-88a9-8acfb2f88c21"**

**},**

**{**

**"ref" : "bookmark",**

**"href" : "https://openstack.example.com/storages/bbfddf09-4d7e-40d5-88a9-8acfb2f88c21"**

**}**

**]**

**},**

**{**

**"**deviceId**" : "4c16a45b-b029-49c4-af84-1abcf458a062",**

**"pooling\_group\_id" : "22zz3344-0099-7766-5544-33225512",**

**"allocate\_status" : "available",**

**"links" : [**

**{**

**"ref" : "self",**

**"href" : "https://openstack.example.com/v1/storages/4c16a45b-b029-49c4-af84-1abcf458a062"**

**},**

**{**

**"ref" : "bookmark",**

**"href" : "https://openstack.example.com/storages/4c16a45b-b029-49c4-af84-1abcf458a062"**

**}**

**]**

**}**

**]**

**}**

## **Storage Device**

This call provides the properties of the specified storage device.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name |  | In | Type | Description |
| URI | /storages/{device\_id} |  |  |  |
|  | id | Query | String | Identifier of the storage device |
|  | capacity\_mb | Body | String | Size of storage devices |
|  | pooling\_group\_id | Body | String | Identifier of Switch (switch cluster) devices |
|  | health\_status | Body | String | Health status of device |
|  |  |  |  |  |

### Operations

#### **GET**

Query storage device properties

**GET /storages/4c16a45b-b029-49c4-af84-1abcf458a062**Content-Type: application/json

#### Response

**{**

**"storage\_device" :**

**{**

**"deviceId" : "4c16a45b-b029-49c4-af84-1abcf458a062",**

**"pooling\_group\_id" : "11z23344-0099-7766-5544-33225511",**

**"health\_status" : "critical",**

**"capacity\_mb" : "1000",**

**"property\_foo1" : "value\_bar1",**

**"property\_foo2" : "value\_bar2"**

**}**

**}**

#### **PUT**

Set device properties such as QoS

Normal response codes: 200

Error response codes: 400,401, 403, 404

**PUT /storages/ffffcccc-dddd-ffff-aaaa-uuuukkkk**

Content-Type: application/json

{

“property\_foo1” : “test1”

}

#### Response

**{**

**“storage\_device” : [  
 {**

**“deviceId” : “ffffcccc-dddd-ffff-aaaa-uuuukkkk”,**

**“capacity\_mb” : “1000”,**

**“pooling\_group\_id” : “11z23344-0099-7766-5544-33225511”,**

**“health\_status” : “Critical”,**

**“property\_foo1” : “test1” ,**

**“property\_foo2” : “value\_bar2”**

**}**

**]**

**}**

#### Error response

**{**

"status": 404,

"error": {

"message": "Error message"

}

**}**

# **OpenStack Flavors**

Flavor creator is a Plasma component that can generate Intel® RSD optimized flavors which can then be inserted into OpenStack installations.

## **Flavor Creator**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name |  | In | Type | Description |
| URI | /flavors |  |  |  |
|  |  |  |  |  |

### Operations

#### **GET**

No operation on /GET

**GET /flavors**

#### Response

**{**

**}**

## **Flavor creator - Generate flavors**

This API call generates OpenStack flavors based on the specified criterion. If the criterion provided is not supported an error message that the operation is not supported shall be returned.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name |  | In | Type | Description |
| URI | /flavors |  |  |  |
| criteria | Input | Request Body | String | Criterion to be used to generate the flavors. Supported criterion determined from output of /criteria |

#### **POST**

Normal response codes: 200

Error response codes: 400, 401, 403, 404, 409

**POST /flavors**

Content-Type: application/json

{

"criteria": “criterion1, criterion2..”

}

#### Response

**{**

**[**

**[**

**"[{\"flavor\": {\"disk\": 17519, \"vcpus\": 4, \"ram\": 16, \"name\": \"S\_irsd-Systems:Rack1-Block1-Sled1-Node1\_Sled:Rack1-Block1-Sled1\_Enclosure:Rack1-Block1\_Rack:Rack1\_\", \"id\": \"1e3fda00-f3cc-46f7-9fd2-7e384ab81770\"}}, {\"extra\_specs\": {\"Rack\": \"Rack1\", \"Systems\": \"Rack1-Block1-Sled1-Node1\", \"Sled\": \"Rack1-Block1-Sled1\", \"Enclosure\": \"Rack1-Block1\"}}]",**

**"[{\"flavor\": {\"disk\": 70076, \"vcpus\": 16, \"ram\": 64, \"name\": \"L\_irsd-Systems:Rack1-Block1-Sled1-Node1\_Sled:Rack1-Block1-Sled1\_Enclosure:Rack1-Block1\_Rack:Rack1\_\", \"id\": \"a4ccdf1b-7df1-4499-85c0-39aa503716e8\"}}, {\"extra\_specs\": {\"Rack\": \"Rack1\", \"Systems\": \"Rack1-Block1-Sled1-Node1\", \"Sled\": \"Rack1-Block1-Sled1\", \"Enclosure\": \"Rack1-Block1\"}}]",**

**"[{\"flavor\": {\"disk\": 35038, \"vcpus\": 8, \"ram\": 32, \"name\": \"M\_irsd-Systems:Rack1-Block1-Sled1-Node1\_Sled:Rack1-Block1-Sled1\_Enclosure:Rack1-Block1\_Rack:Rack1\_\", \"id\": \"c40f7fa8-7699-450c-abe2-3194c6aaa76c\"}}, {\"extra\_specs\": {\"Rack\": \"Rack1\", \"Systems\": \"Rack1-Block1-Sled1-Node1\", \"Sled\": \"Rack1-Block1-Sled1\", \"Enclosure\": \"Rack1-Block1\"}}]"**

**],**

**}**

#### Error reponse

**{**

"status": 404,

"error": {

"message": "Error message"

}

**}**

## **Flavor Creator – Supported criteria**

Provide a list of criteria which are supported by the flavor creator. One or more criteria are used to create the flavors; example of such criteria include:

* CPU model/features
* Location hierarchy
* Platform features

No input parameters are necessary for this call.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name |  | In | Type | Description |
| URI | /flavor/criteria |  |  |  |
|  |  |  |  |  |

### Operations

#### **GET**

List all supported flavor generation criteria along with their descriptions.

Normal response codes: 200

Error response codes: 401, 403

**GET /flavor/criteria**

#### Response

**{**

**{**

**"critera": [**

**{**

**"name": "default",**

**"description: "This generates 3 flavors S,M,L based on node asset tag"**

**},**

**{**

**"name": "example",**

**"description": "This is a dummy criteria for demo purpose"**

**}**

**]**

**}**

**}**

#### **DELETE**

Delete a critiera

Not applicable. Supported criteria are determined based on installed plugins.