Deployment Type Info API

You can use the Deployment Type Info API to list the current system Deployment Type and edit the Deployment Type.

This section explains the supported API operations for Deployment Type Info and outlines the parameters.

• Deployment Type Info API Parameters, page 1
• Deployment Type Info Get, page 2
• Deployment Type Info Update, page 3
• VM Validation, page 6

Deployment Type Info API Parameters

deploymentType

Defines the type of deployment.
Deployment Type Info Get

Returns the current deployment state and the results of the capacity and system validation tests.

**Syntax**

**URL:** https://<server>/unifiedconfig/config/deploymenttypeinfo

**HTTP Method:** GET

**Response**

See also General Usage.

**Example XML Response**

```xml
<deploymentTypeInfo>
  <systemValidationStatus>
    <!-- same as serviceability API -->
  </systemValidationStatus>
</deploymentTypeInfo>
```
To get the VM Validation Log, instead of getting the Deployment Type as detailed in the preceding table, you specify an additional path.

Get VM Validation returns the log file for the last attempt at VM Validation; that is, the log file for attempting to change the Deployment Type to CCEPACM1 (7).

**URL:** https://<server>/unifiedconfig/config/deploymenttypeinfo

**HTTP Method:** PUT

**Response**
Returns the contents of the log file, if one exists.

If the Deployment Type has not been changed to CCEPACM1, the following message is returned: *No VM validation log exists on this AW. To generate one, attempt to configure the system to CCEPACM1.*

### Deployment Type Info Update

Sets the specified Deployment Type, if the system validation check, capacity check, and VM Validation for that Deployment Type pass and are required.

**Syntax**

**URL:** https://<server>/unifiedconfig/config/deploymenttypeinfo

**HTTP Method:** PUT

**Input/Output Format**

xml

**Response**

See [HTTP Responses](https://<server>/unifiedconfig/config/deploymenttypeinfo).

**Parameters**

See Deployment Type Info API Parameters, on page 1.

**Operation Validation**

**deploymentType**

- Optional field.
- Valid values are:
0 = General
1 = NAM
2 = VRU
3 = NAM Rogger
4 = ICM Router Logger
5 = 8000 Agents Router Logger
6 = 12000 Agents Router Logger
7 = Packaged CCE: CCE-PAC-M1
8 = ICM Rogger
9 = 4000 Agents Rogger
10 = Packaged CCE: CCE-PAC-M1 Lab Only
11 = HCS-CC 1000 Agents
12 = HCS-CC 500 Agents
13 = UCCE 450 Agents Progger

- If an invalid deploymentType is specified, an API error is thrown.
- Switching back to Deployment Type 0 is not allowed because the call processing stops.

changeStamp
- Required field.
- Integers only: start with zero.

Example XML Request Payload (All Deployments Except Packaged CCE: CCE-PAC-M1)

```xml
<deploymentTypeInfo>
  <changeStamp>0</changeStamp>
  <deploymentType>0</deploymentType>
</deploymentTypeInfo>
```

Example XML Request Payload (Packaged CCE: CCE-PAC-M1)

```xml
<deploymentTypeInfo>
  <changeStamp>59</changeStamp>
  <deploymentType>7</deploymentType>
  <vmHosts>
    <vmHost>
      <id>sideA</id>
      <host>"ESXi Side A hostname or IP"</host>
      <userName>"username"</userName>
      <password>"password"</password>
    </vmHost>
    <vmHost>
      <id>sideB</id>
      <host>"ESXi Side B hostname or IP"</host>
      <userName>"username"</userName>
      <password>"password"</password>
    </vmHost>
  </vmHosts>
</deploymentTypeInfo>
```
Example API Error Response if an Invalid Deployment Type Is Specified

```
<apiErrors>
  <apiError>
    <errorData>deploymentType</errorData>
    <errorMessage>Invalid deployment type.</errorMessage>
    <errorType>invalidInput.invalidDeploymentType</errorType>
  </apiError>
</apiErrors>
```

Example XML Error Response if Capacity Check Fails

The following example lists a single check that failed. However, if there are multiple capacity check failures, an API error is returned in this list for each of those failures:

```
<apiErrors>
  <apiError>
    <errorMessage>You have reached capacity for Reason Code. The limit is 100</errorMessage>
    <errorType>capacityLimit.maxItems</errorType>
  </apiError>
</apiErrors>
```

Example XML Error Response if System Validation Check Fails

Similar to capacity check errors, system validation check errors are returned as individual API errors. The example below shows a multiple rules failure.

```
<apiErrors>
  <apiError>
    <errorMessage>System validation check failed for rule TYPE10_NETWORK_VRU_MAP_COUNT. Min = 4, Max = 4, Actual = 0</errorMessage>
    <errorType>deploymentTypeInfo.invalidSystem</errorType>
  </apiError>
  <apiError>
    <errorMessage>System validation check failed for rule UCM_PIM_COUNT. Min = 1, Max = 1, Actual = 2</errorMessage>
    <errorType>deploymentTypeInfo.invalidSystem</errorType>
  </apiError>
  <apiError>
    <errorMessage>System validation check failed for rule PG_COUNT. Min = 1, Max = 2, Actual = 4</errorMessage>
    <errorType>deploymentTypeInfo.invalidSystem</errorType>
  </apiError>
  <apiError>
    <errorMessage>System validation check failed for rule SERVICE_MEMBER_COUNT. Min = 0, Max = 0, Actual = 1</errorMessage>
    <errorType>deploymentTypeInfo.invalidSystem</errorType>
  </apiError>
</apiErrors>
```
VM Validation

Overview

VM validation is required for CCE-PAC-M1. When attempting to change the Deployment Type to Packaged CCE: CCE-PAC-M1, the virtual machine infrastructure is validated to ensure the systems and resources are correctly configured. The steps that occur in the validation are as follows:

1. User provides hostname or IP, userName, and password for the ESXi servers that are configured for SideA and SideB.
2. ESXi server properties are validated for each host.
3. A list of the Virtual Machines present on each host are retrieved and matched to the defined profiles. **Note:** since no naming convention is defined, the profile list is literally searched for a match to the VM properties.
4. A log file is written to the HD of AW with the contents of the validation attempt.
5. If all ESXi properties match, and all required profiles match a Virtual Machine, and no additional Virtual Machines are present, the deployment change is allowed to proceed.
6. If any of the preceding conditions fail, an error is reported back to the user.

The validation library is deployment aware and utilizes the vm_validation.xml spring configuration file to define the ESXi server properties and VMs that are validated. The IDs in the file must match those in the vmHost xml; that is, sideA and sideB. See the example XML in Deployment Type Info Update, on page 3.

**Table 1: VM Checks**

<table>
<thead>
<tr>
<th>#</th>
<th>Requirement</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>VM - Number of CPUs</td>
<td>Required (Exact Match)</td>
</tr>
<tr>
<td>2</td>
<td>CPU Reservation</td>
<td>Required (Exact Match)</td>
</tr>
<tr>
<td>3</td>
<td>VM - Exact Memory</td>
<td>Required (Exact Match)</td>
</tr>
<tr>
<td>4</td>
<td>VM - Exact Disk Size(s)</td>
<td>Required (Exact Match - Order Independent)</td>
</tr>
<tr>
<td>5</td>
<td>VM - Exact Number of Disks</td>
<td>Required (Exact Match)</td>
</tr>
<tr>
<td>6</td>
<td>VM - VMWare Tools</td>
<td>Collect and Log Only</td>
</tr>
<tr>
<td>7</td>
<td>Host - Vendor</td>
<td>Required (Exact Match)</td>
</tr>
<tr>
<td>8</td>
<td>Host - BIOS</td>
<td>Required (Major Version Only; that is: C260.xxxx)</td>
</tr>
<tr>
<td>9</td>
<td>Host - ESXi Version</td>
<td>Required (Exact Match)</td>
</tr>
<tr>
<td>10</td>
<td>Host - Total Number of VMs</td>
<td>Required (Exact Match)</td>
</tr>
</tbody>
</table>
## Errors

The following is a list of errors returned if the host information is incorrect or the VMs are not in a valid layout.

### Example API Error Response

**Error 1:** Missing VM Host - Cannot match profile sideC and cannot find host info for Side A and B.

```xml
<apiErrors>
  <apiError>
    <errorData>deploymentType</errorData>
    <errorMessage>Unable to match host to profile. Host ID: sideC Required Profiles: [sideA, sideB]</errorMessage>
    <errorType>deploymentTypeInfo.invalidSystem.missingVMHostInfo</errorType>
  </apiError>
</apiErrors>
```

**Error 2:** Cannot connect to VM Host - Invalid host (ip), userName, or password provided for side A

```xml
<apiErrors>
  <apiError>
    <errorData>deploymentType</errorData>
    <errorDetail xsi:type="vmHostErrorDetail">
      <hosts>
        <hostInfo>
          <host>10.86.141.10</host>
          <id>sideA</id>
          <userName>root2</userName>
        </hostInfo>
      </hosts>
    </errorDetail>
    <errorMessage>Unable to connect to host(s): [[Id: sideA Host: 10.86.141.10]]</errorMessage>
    <errorType>deploymentTypeInfo.invalidSystem.cannotConnectToVMHost</errorType>
  </apiError>
</apiErrors>
```

**Error 3:** Invalid VM Layout - Side A has valid ESX Host properties, but an invalid layout, Side B has invalid ESX Host properties, but a valid layout

**Error 3** generated a log file retrievable at the URL [https://<server>/unifiedconfig/config/deploymenttypeinfo/vmvalidation/log](https://<server>/unifiedconfig/config/deploymenttypeinfo/vmvalidation/log) (example below)

```xml
<apiErrors>
  <apiError>
    <errorData>deploymentType</errorData>
    <errorDetail xsi:type="invalidVMLayoutErrorDetail">
      <hostPropertiesValid>true</hostPropertiesValid>
      <id>sideA</id>
      <vmLayoutValid>false</vmLayoutValid>
    </errorDetail>
    <errorMessage>The virtual machine host properties or layout is invalid.</errorMessage>
    <errorType>deploymentTypeInfo.invalidSystem.invalidVMLayout</errorType>
  </apiError>
</apiErrors>
```
<apiError>
  <errorData>deploymentType</errorData>
  <errorDetail xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:type="invalidVMLayoutErrorDetail">
    <hostPropertiesValid>false</hostPropertiesValid>
    <id>sideB</id>
    <vmLayoutValid>true</vmLayoutValid>
  </errorDetail>
  <errorMessage>The virtual machine host properties or layout is invalid.</errorMessage>
  <errorType>deploymentTypeInfo.invalidSystem.invalidVMLayout</errorType>
</apiError>

Log:
VM Validation Results: Wed May 23 11:25:09 EDT 2012
Overall: true
Valid Systems: 2 of 2
Summary:
ESX Server: sideA
ESX Server Properties Valid: true
VM Layout Valid: true
Server Result:
Required Version: 5.0.0
Required Min CPU Cores: 20
Required Min Memory (MB): 95000
Required HD(s) (GB): [1392, 1949, 273]
Required Bios <Major version>: C260
Required Vendor: Cisco Systems Inc
Found Version: 5.0.0
Found CPU Cores: 20
Found Memory (MB): 98185
Found HD(s) (GB): [1392, 273, 1949]
Found Bios: C260.1.4.2b.0.102620111637
Found Vendor: Cisco Systems Inc

Virtual Machines Matching Defined Profiles:
VM: BB-CCE-DataSvr-A
Profile: Unified CCE Data Server
OS: Microsoft Windows Server 2008 R2 (64-bit)
CPU Cores: 4
Reservation: 5100
RAM (MB): 8192
HD(s) (GB): [80, 750, 500]
VMware Tools Version: 8384

Required Profiles without Matching Virtual Machines: None
Optional Profiles without Matching Virtual Machines: None
Virtual Machines without Matching Profiles: None