Common Tenant Tasks

Common Tenant Tasks

Tenants Overview

- A tenant contains policies that enable qualified users domain-based access control. Qualified users can access privileges such as tenant administration and networking administration.
- A user requires read/write privileges for accessing and configuring policies in a domain. A tenant user can have specific privileges into one or more domains.
- In a multitenancy environment, a tenant provides group user access privileges so that resources are isolated from one another (such as for endpoint groups and networking). These privileges also enable different users to manage different tenants.

Tenant Creation

A tenant contains primary elements such as filters, contracts, bridge domains, and application profiles that you can create after you first create a tenant.

Adding a Tenant

A tenant is a policy owner in the virtual fabric. A tenant can be either a private or a shared entity. For example, you can create a securely partitioned private tenant or a tenant with contexts and bridge domains shared by other tenants. A shared type of tenant is typically named common, default, or infra.

In the management information model, a tenant is represented by a managed object (MO) of class fv:Tenant. According to the *Cisco APIC Management Information Model Reference*, an object of the fv:Tenant class is a child of the policy resolution universe (uni) class and has a distinguished name (DN) format of `uni/tn-[`*`name`*`]`.

Note You can only add one tenant at a time.

The following examples show how to add a new tenant named ExampleCorp using XML and JSON.

Example: Using the JSON API to Add a Tenant

To create a new tenant, you must specify the class and sufficient naming information, either in the message body or in the URI.

To create a new tenant using the JSON API, send this HTTP POST message:

```
POST https://apic-ip-address/api/mo/uni.json

{
  "fvTenant" : {
    "attributes" : {
      "name" : "ExampleCorp"
    }
  }
}
```

Alternatively, you can name the tenant in the URI, as in this example:

```
POST https://apic-ip-address/api/mo/uni/tn-ExampleCorp.json

{
  "fvTenant" : {
    "attributes" : {
    }
  }
}
```

If a response is requested (by appending `?rsp-subtree=modified` to the POST URI), a successful operation returns the following response body:

```
{
  "imdata" :
  [{
      "fvTenant" : {
        "attributes" : {
          "instanceId" : "0:0",
          "childAction" : "deleteNonPresent",
          "dn" : "uni/tn-ExampleCorp",
          "lcOwn" : "local",
          "name" : "ExampleCorp",
          "replTs" : "never",
          "rn" : "",
          "status" : "created"
        }
      }
    }
  ]
}
```

To delete the tenant, send this HTTP DELETE message:

```
DELETE https://apic-ip-address/api/mo/uni/tn-ExampleCorp.json
```

Alternatively, you can send an HTTP POST message with sufficient naming information and with `"status" : "deleted"` in the fv:Tenant attributes, as in this example:

```
POST https://apic-ip-address/api/mo/uni.json

{
  "fvTenant" : {
    "attributes" : {
      "name" : "ExampleCorp",
      "status" : "deleted"
    }
  }
}
```

Example: Using the XML API to Add a Tenant

To create a new tenant, you must specify the class and sufficient naming information, either in the message body or in the URI.

To create a new tenant named ExampleCorp using the XML API, send this HTTP POST message:

```
POST https://apic-ip-address/api/mo/uni.xml

<fvTenant name="ExampleCorp"/>
```

Alternatively, you can name the tenant in the URI, as in this example:

```
POST https://apic-ip-address/api/mo/uni/tn-ExampleCorp.xml

<fvTenant />
```

If a response is requested (by appending `?rsp-subtree=modified` to the POST URI), a successful operation returns the following response body:

```
<imdata>
    <fvTenant
        instanceId="0:0"
        childAction="deleteNonPresent"
        dn="uni/tn-ExampleCorp"
        lcOwn="local"
        name="ExampleCorp"
        replTs="never"
        rn=""
        status="created"
    />
</imdata>
```

To delete the tenant, send this HTTP DELETE message:

```
DELETE https://apic-ip-address/api/mo/uni/tn-ExampleCorp.xml
```

Alternatively, you can send an HTTP POST message with sufficient naming information and with `status="deleted"` in the fv:Tenant attributes, as in this example:

```
POST https://apic-ip-address/api/mo/uni.xml

<fvTenant name="ExampleCorp" status="deleted"/>
```