Citrix NetScaler - NITRO REST - Getting Started Guide

Citrix® NetScaler® 10.5

Last Updated: March 2014

Document code: June 10 2014 05:05:32

Contents

NITRO API

The NetScaler NITRO protocol allows you to configure and monitor the NetScaler appliance programmatically.

NITRO exposes its functionality through Representational State Transfer (REST) interfaces. Therefore, NITRO applications can be developed in any programming language. Additionally, for applications that must be developed in Java or .NET or Python, NITRO APIs are exposed through relevant libraries that are packaged as separate Software Development Kits (SDKs).

Note: You must have a basic understanding of the NetScaler appliance before using NITRO.

To use the NITRO protocol, the client application needs only the following:

- Access to a NetScaler appliance, version 9.2 or later.
- To use REST interfaces, you must have a system to generate HTTP or HTTPS requests (payload in JSON format) to the NetScaler appliance. You can use any programming language or tool.
- For Java clients, you must have a system where Java Development Kit (JDK) 1.5 or later is available. The JDK can be downloaded from http://www.oracle.com/technetwork/java/javase/downloads/index.html.
- For .NET clients, you must have a system with .NET framework 3.5 or later installed. The .NET framework can be downloaded from http://www.microsoft.com/downloads/en/default.aspx.
- For Python clients, you must have a system with Python 2.7 or above version and the Requests library (available in <NITRO_SDK_HOME>/lib) installed.

Obtaining the NITRO Package

The NITRO package is available as a tar file on the **Downloads** page of the NetScaler appliance's configuration utility. You must download and un-tar the file to a folder on your local system. This folder is referred to as `<NITRO_SDK_HOME>` in this documentation.

The folder contains the NITRO libraries in the `lib` subfolder. The libraries must be added to the client application classpath to access NITRO functionality. The `<NITRO_SDK_HOME>` folder also provides samples and documentation that can help you understand the NITRO SDK.

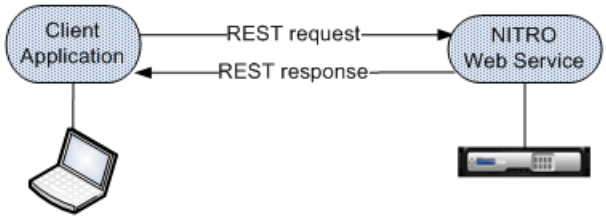
Note:

- The REST package contains only documentation for using the REST interfaces.
- For the Python SDK, the library must be installed on the client path. For installation instructions, read the `<NITRO_SDK_HOME>/README.txt` file.

How NITRO Works

The NITRO infrastructure consists of a client application and the NITRO Web service running on a NetScaler appliance. The communication between the client application and the NITRO web service is based on REST architecture using HTTP or HTTPS.

Figure 1-1. NITRO execution flow



As shown in the above figure, a NITRO request is executed as follows:

1. The client application sends REST request message to the NITRO web service. When using the SDKs, an API call is translated into the appropriate REST request message.
2. The web service processes the REST request message.
3. The NITRO web service returns the corresponding REST response message to the client application. When using the SDKs, the REST response message is translated into the appropriate response for the API call.

To minimize traffic on the NetScaler network, you retrieve the whole state of a resource from the server, make modifications to the state of the resource locally, and then upload it back to the server in one network transaction. For example, to update a load balancing virtual server, you must retrieve the object, update the properties, and then upload the changed object in a single transaction.

Note: Local operations on a resource (changing its properties) do not affect its state on the server until the state of the object is explicitly uploaded.

NITRO APIs are synchronous in nature. This means that the client application waits for a response from the NITRO web service before executing another NITRO API.

REST Web Services

REST (REpresentational State Transfer) is an architectural style based on simple HTTP requests and responses between the client and the server. REST is used to query or

change the state of objects on the server side. In REST, the server side is modeled as a set of entities where each entity is identified by a unique URL. For example, the load balancing virtual server entity is identified by the URL `http://<NSIP>/nitro/v1/config/<lbvserver>/<lbvserver_name>`.

Each resource also has a state on which the following operations can be performed:

- **Create.** Clients can create new server-side resources on a "container" resource. You can think of container resources as folders, and child resources as files or subfolders. The calling client provides the state for the resource to be created. The state can be specified in the request by using XML or JSON format. The client can also specify the unique URL that will identify the new object. Alternatively, the server can choose and return a unique URL identifying the created object. The HTTP method used for Create requests is POST.
- **Read.** Clients can retrieve the state of a resource by specifying its URL with the HTTP GET method. The response message contains the resource state, expressed in JSON format.
- **Update.** You can update the state of an existing resource by specifying the URL that identifies that object and its new state in JSON or XML, using the PUT HTTP method.
- **Delete.** You can destroy a resource that exists on the server-side by using the DELETE HTTP method and the URL identifying the resource to be removed.

In addition to these four CRUD operations (Create, Read, Update, and Delete), resources can support other operations or actions. These operations use the HTTP POST method, with the URL specifying the operation to be performed and the request body specifying the parameters for that operation.

NetScaler NITRO APIs are categorized depending on the scope and purpose of the APIs into system APIs, feature configuration APIs, and feature statistics APIs.

Note: All NITRO operations are logged in the `/var/log/nitro.log` file on the appliance.

Performing System Level Operations

The first step towards using NITRO is to establish a session with the NetScaler appliance and then authenticate the session by using the NetScaler administrator's credentials. You must specify the username and password in the `login` object. The session ID that is created must be specified in the request header of all further operations in the session.

Note: You must have a user account on the appliance to log on to it. The configuration operations that you can perform are limited by the administrative roles assigned to your account.

To connect to a NetScaler appliance with NSIP address 10.102.29.60 by using the HTTP protocol:

- **URL.** https://10.102.29.60/nitro/v1/config/login/
- **Method.** POST
- **Request.**
  - **Header.**

```
Content-Type:application/vnd.com.citrix.netscaler.login
+json
```

Note: Content types such as `'application/x-www-form-urlencoded'` that were supported in earlier versions of NITRO can also be used. You must make sure that the payload is the same as used in earlier versions. The payloads provided in this documentation are only applicable if the content type is of the form `'application/vnd.com.citrix.netscaler.login +json'`.

  - **Payload.**

```
{
    "login":
    {
        "username":"admin",
        "password":"verysecret"
    }
}
```

- **Response.**
  - **Header.**

```
HTTP/1.0 201 Created
Set-Cookie:
NITRO_AUTH_TOKEN=##87305E9C51B06C848F0942; path=/nitro/v1
```

Note: By default, the connection to the appliance expires after 30 minutes of inactivity. You can modify the timeout period by specifying a new timeout period (in seconds) in the `login` object. For example, to modify the timeout period to 60 minutes, the request payload is:

```
{
    "login":
    {
        "username":"admin",
        "password":"verysecret",
        "timeout":3600
    }
}
```

You can also connect to the appliance to perform a single operation, by specifying the username and password in the request header of the operation. For example, to connect to an appliance while adding a load balancing virtual server:

- **URL.** https://10.102.29.60/nitro/v1/config/lbvserver/
- **Method.** POST
- **Request.**
  - **Header.**

```
X-NITRO-USER:admin
X-NITRO-PASS:verysecret
Content-Type:application/vnd.com.citrix.netscaler.lbvserver
+json
```

  - **Payload.**

```
{
  "lbvserver":
  {
    ...
    ...
    ...
}
```

- **Response.**
  - **Header.**

```
HTTP/1.0 201 Created
```

You can also perform other system-level operations such as enabling NetScaler features and modes, saving and clearing NetScaler configurations, setting the session timeout, setting the severity of the exceptions to be handled, setting the behavior of bulk operations, and disconnecting from the appliance.

For more information on the REST messages, see the **Configuration** node of the `<NITRO_SDK_HOME>/index.html` file.

Example 1: Enable the load balancing feature

- **URL.** http://10.102.29.60/nitro/v1/config/nsfeature?action=enable
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.nsfeature
+json
```

- **Payload**

```
{
    "nsfeature":
    {
        "feature":
        [
            "LB",
        ]
    }
}
```

Example 2: Save NetScaler configurations

- **URL.** http://10.102.29.60/nitro/v1/config/nsconfig?action=save
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.nsconfig
+json
```

  - **Payload**

```
{
    "nsconfig":{}
}
```

Example 3: Disconnecting from the appliance

- **URL.** https://10.102.29.60/nitro/v1/config/logout/
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.logout
+json
```

  - **Payload**

```
{
    "logout":{}
}
```

Note: Make sure that you have saved the configurations before performing this operation.

Configuring NetScaler Features

A NetScaler appliance has multiple features, and each feature has multiple resources. Each NetScaler resource, depending on the operation to be performed on it, has a unique URL associated with it. URLs for configuration operations have the format `http://<NSIP>/nitro/v1/config/<resource_type>/<resource_name>`. For example, to access the lbvserver named MyFirstLbVServer on a NetScaler with IP 10.102.29.60, the URL is `http://10.102.29.60/nitro/v1/config/lbvserver/MyFirstLbVServer`.

Using NITRO you can perform the following operations:

Create | Retrieve | Update | Delete | Enable/Disable | Unset | Bind/Unbind | Bulk operations

For more information on the REST messages, see the **Configuration** node of the `<NITRO_SDK_HOME>/index.html` file.

Create

To create a new resource (for example, an lbvserver) on the appliance, specify the resource name and other related arguments in the specific resource object. For a lbvserver resource, the object would be an `lbvserver` object.

To create an lbvserver named "MyFirstLbVServer":

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver/
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.lbvserver
+json
```

  - **Payload**

```
{
    "lbvserver":
    {
        "name":"MyFirstLbVServer",
        "servicetype":"http"
    }
}
```

Retrieve

NetScaler resource properties can be retrieved as follows:

- To retrieve details of all resources of a specific type, specify the resource type in the URL.

  URL format: `http://<NSIP>/nitro/v1/config/<resource_type>`

- To retrieve details of a specific resource on the NetScaler appliance, specify the resource name in the URL.

  URL format: `http://<NSIP>/nitro/v1/config/<resource_type>/<resource_name>`

- To retrieve specific details of a resource, specify the resource details that you want to view in the URL.

  URL format: `http://<NSIP>/nitro/v1/config/<resource_type>/<resource_name>?attrs=<attrib1>,<attrib2>`

- To retrieve details of resources on the basis of some filter, specify the filter conditions in the URL.

  URL format: `http://<NSIP>/nitro/v1/config/<resource_type>?filter=<attrib1>:<value>,<attrib2>:<value>`

- If the request is likely to result in a large number of resources, you can divide the results into pages and retrieve them page by page.

  For example, assume that you have a NetScaler that has 53 lbvservers and you want to retrieve all the lbvservers. So, instead of retrieving all 53 in one response, you can configure the results to be divided into pages of 10 lbvservers each (6 pages total), and retrieve them from the NetScaler page by page.

  URL format: `http://<NSIP>/nitro/v1/config/<resource_type>?pageno=<value>&pagesize=<value>`

  You specify the page count with the `pagesize` parameter and the page number that you want to retrieve with the `pageno` parameter.

- To get the number of resources that are likely to be returned by a request, you can use the `count` query string parameter to ask for a count of the resources to be returned, rather than the resources themselves.

  URL format: `http://<NSIP>/nitro/v1/config/<resource_type>?count=yes`

To retrieve the details of an lbvserver named "MyFirstLbVServer":

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver/MyFirstLbVServer/
- **HTTP Method.** GET
- **Request.**
  - **Header**

    ```
    Cookie:NITRO_AUTH_TOKEN=tokenvalue
    ```

- **Response.**
  - **Header**

    ```
    HTTP/1.0 200 OK
    Content-Type:application/vnd.com.citrix.netscaler.lbvserver+json
    ```

- **Payload**

```
{
    "lbvserver":
    [
        {
            "name":"MyFirstLbVServer",
            "servicetype":"http",
            "insertvserveripport":"OFF",
            "ip":"0.0.0.0",
            "port":80,
            ...
        }
    ]
}
```

Update

To update the details of an existing resource on the NetScaler appliance, specify the resource name, and the arguments to be updated, in the specific resource object.

To change the load balancing method to ROUNDROBIN and update the comment property for a load balancing virtual server named "MyFirstLbVServer":

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver/MyFirstLbVServer/
- **HTTP Method.** PUT
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.lbvserver
+json
```

  - **Payload**

```
{
    "lbvserver":
    {
        "name":"MyFirstLbVServer",
        "lbmethod":"ROUNDROBIN",
        "comment":"Updated comments"
    }
}
```

Delete

To delete a NetScaler resource, specify the resource name in the URL.

To delete a load balancing virtual server named "MyFirstLbVServer":

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver/MyFirstLbVServer
- **HTTP Method.** DELETE

Enable/Disable

To enable a resource on the NetScaler appliance, specify the resource name in the specific resource object.

To enable a load balancing virtual server named "MyFirstLbVServer":

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver?action=enable
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.lbvserver
+json
```

  - **Payload**

```
{
    "lbvserver":
    {
        "name":"MyFirstLbVServer"
    }
}
```

Note: To disable a resource, in the URL specify the action as "disable".

Unset

To unset the value that is set to a parameter, specify the action as "unset" and in the payload, specify the parameters to be unset.

To unset the load balancing method and the comments specified for a load balancing virtual server named "MyFirstLbVServer":

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver?action=unset
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.lbvserver
+json
```

  - **Payload**

```
{
    "lbvserver":
    {
        "name":"MyFirstLbVServer",
        "lbmethod":true,
        "comment":true,
```

```
    }
}
```

Bind/Unbind

To bind a resource to another, specify the name of the two resources and specify the weight for the binding.

To bind a service named "svc_prod" to a load balancing virtual server named "MyFirstLbVServer", by specifying a certain weight for the binding:

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver_service_binding/
- **HTTP Method.** PUT
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.lbvserver_service_binding+json
```

  - **Payload**

```
{
    "lbvserver_service_binding":
    {
        "name":"MyFirstLbVServer",
        "servicename":"svc_prod",
        "weight":111,
    }
}
```

Note: To unbind, specify the arguments in the URL as follows:

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver_service_binding/MyFirstLbVServer?args=servicename:svc_prod
- **HTTP Method.** DELETE

Bulk operations

You can create, retrieve, update, and delete multiple resources simultaneously and thus minimize network traffic. For example, you can add multiple load balancing virtual servers in the same operation. To perform a bulk operation, specify the required parameters in the same request payload.

To account for the failure of some operations within the bulk operation, NITRO allows you to configure one of the following behaviors:

- **Exit.** When the first error is encountered, the execution stops. The commands that were executed before the error are committed.

- **Rollback.** When the first error is encountered, the execution stops. The commands that were executed before the error are rolled back. Rollback is only supported for add and bind commands.
- **Continue.** All the commands in the list are executed even if some commands fail.

You must specify the behavior of the bulk operation in the request header using the X-NITRO-ONERROR parameter.

To add two load balancing virtual servers in one operation and continue if one command fails:

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver/
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.lbvserver_list+json
X-NITRO-ONERROR:continue
```

  - **Payload**

```
{
    "lbvserver":
    [
        {
            "name":"new_lbvserver1",
            "servicetype":"http"
        },

        {
            "name":"new_lbvserver2",
            "servicetype":"http"
        }
    ]
}
```

- **Response**
  - **Header**

```
HTTP/1.0 207 Multi Status
```

  - **Payload**

```
{
    "errorcode":273,
    "message":"Resource already exists",
    "severity":"ERROR",
    "response":
    [
        {
```

```
            "errorcode": 0,
            "message": "Done",
            "severity": "NONE"
        },
        {
            "errorcode": 273,
            "message":"Resource already exists",
            "severity": "ERROR"
        }
    ]
}
```

Binding NetScaler Resources

NetScaler resources form relationships with each other through the process of binding. This is how services are associated with an lbvserver (by binding them to it), or how various policies are bound to an lbvserver. Each binding relationship is represented by its own object. A binding resource has properties representing the name of each NetScaler resource in the binding relationship. It can also have other properties related to that relationship (for example, the weight of the binding between an lbvserver resource and a service resource).

Note: Unlike for NetScaler entities, you use a PUT HTTP method, instead of POST, for adding new binding resources.

For more information on the REST messages, see the **Configuration** node of the `<NITRO_SDK_HOME>/index.html` file.

To bind a service to a load balancing virtual server named "MyFirstLbVServer" and specify a weight for the binding:

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver_service_binding/MyFirstLbVServer?action=bind
- **HTTP Method.** PUT
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.lbvserver_service_binding+json
```

  - **Payload**

```
{
    "lbvserver_service_binding":
    {
        "servicename":"svc_prod",
        "weight":20,
        "name":"MyFirstLbVServer"
    }
}
```

To retrieve list of all the services bound to a virtual server "lbv1":

- **URL.** http://10.102.29.60/nitro/v1/config/lbvserver_service_binding/lbv1?attrs=servicename
- **HTTP Method.** GET

For more information on retrieving information, see the "Retrieving properties of a resource" section in Configuring NetScaler Features.

Globally Bind Resources

Some NetScaler resources can be bound globally to affect the whole system. For example, if a compression policy is bound to an lbvserver, the policy affects only the traffic on that lbvserver. However, if bound globally, it can affect any traffic on the appliance, regardless of which virtual servers handle the traffic.

The names of NITRO resources that can be used to bind resources globally have the pattern `<featurename>global_<resourcetype>_binding`. For example, the object aaaglobal_preauthenticationpolicy_binding is used to bind preauthentication policies globally.

To bind the policy named preautpol1 globally at priority 200:

- **URL.** http://10.102.29.60/nitro/v1/config/aaaglobal_aaapreauthenticationpolicy_binding?action=bind
- **HTTP Method.** PUT
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.aaaglobal_aaapreauthenticationpoli
cy_binding+json
```

  - **Payload**

```
{
    "aaaglobal_aaapreauthenticationpolicy_binding":
    {
        "policy":"preautpol1",
        "priority":200
    }
}
```

Configuring a NetScaler Cluster

You can use NITRO to add or create and manage a NetScaler cluster.

Cluster Instance Operations

All operations on a cluster instance must be performed on the `clusterinstance` object.

To create a cluster instance with ID 1:

- **URL.** http://10.102.29.60/nitro/v1/config/clusterinstance/
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.clusterinstance+json
```

  - **Payload**

```
{
    "clusterinstance":
    {
        "clid":1,
        "preemption":"ENABLED"
    }
}
```

Cluster Node Operations

All operations on a cluster node must be performed on the `clusternode` object.

To add a cluster node with NSIP address 10.102.29.60:

- **URL.** http://10.102.29.60/nitro/v1/config/clusternode/
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.clusternode+json
```

  - **Payload**

```
{
    "clusternode":
    {
        "nodeid":1,
        "ipaddress":"10.102.29.60",
        "state":"ACTIVE",
        "backplane":"1/1/2"
    }
}
```

Add a Cluster IP Address

To define a cluster IP address, specify the required parameters in the `nsip` object.

To configure a cluster IP address on NetScaler appliance with IP address 10.102.29.60:

- **URL.** http://10.102.29.60/nitro/v1/config/nsip/
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.nsip+json
```

  - **Payload**

```
{
    "nsip":
    {
        "ipaddress":"10.102.29.61",
        "netmask":"255.255.255.255",
        "type":"CLIP"
    }
}
```

Add a Spotted IP Address

To configure an IP address as spotted, specify the required parameters in the `nsip` object. This configuration must be done on the cluster IP address.

To configure a spotted SNIP address on a node with ID 1:

- **URL.** http://10.102.29.60/nitro/v1/config/nsip/
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.nsip+json
```

  - **Payload**

```
{
    "nsip":
    {
        "ipaddress":"10.102.29.77",
        "netmask":"255.255.255.0",
        "type":"SNIP",
        "ownernode":1
    }
}
```

Join NetScaler Appliance to Cluster

To join an appliance to a cluster, specify the required parameters in the `cluster` object.

To join a NetScaler appliance to a cluster:

- **URL.** http://10.102.29.60/nitro/v1/config/cluster/
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/vnd.com.citrix.netscaler.cluster
+json
```

  - **Payload**

```
{
    "cluster":
    {
        "clip":"10.102.29.61",
        "password":"verysecret"
    }
}
```

Linkset Operations

To configure a linkset, do the following:

1. Create a linkset by specifying the required parameters in the `linkset` object.

   To add a linkset LS/1:

   - **URL.** http://10.102.29.60/nitro/v1/config/linkset/
   - **HTTP Method.** POST
   - **Request.**
     - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.linkset+json
```

     - **Payload**

```
{
    "linkset":
    {
        "id":"LS/1"
    }
}
```

2. Bind the required interfaces to the linkset by specifying the interfaces in the `linkset_interface_binding` object.

   To bind interfaces 1/1/2 and 2/1/2 to linkset LS/1:

- **URL.** http://10.102.29.60/nitro/v1/config/linkset_interface_binding/ LS%2F1?action=bind
- **HTTP Method.** PUT
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.linkset_interface_binding+json
```

  - **Payload**

```
{
    "linkset_interface_binding":
    {
        "id":"LS/1",
        "ifnum":"1/1/2 2/1/2"
    }
}
```

Retrieving Feature Statistics

The NetScaler appliance collects statistics about the usage of its features and the corresponding resources. NITRO can retrieve these statistics.

- URL to get statistics of a feature must have the format `http://<NSIP>/nitro/v1/stat/<feature_name>`.
- URL to get the statistics of a resource must have the format: `http://<NSIP>/nitro/v1/stat/<resource_type>/<resource_name>`.

For more information on the REST messages, see the **Statistics** node of the `<NITRO_SDK_HOME>/index.html` file.

To get the statistics of a lbvserver named "MyFirstLbVServer":

- **URL.** http://10.102.29.60/nitro/v1/stat/lbvserver/MyFirstLbVServer
- **HTTP Method.** GET
- **Request.**
  - **Header.**

```
Content-Type:application/vnd.com.citrix.netscaler.lbvserver
+json
```

- **Response.**
  - **Header**

```
HTTP/1.0 200 OK
```

- **Payload**

```
{
    "lbvserver":
    [
        {
            "name":"MyFirstLbVServer",
            "establishedconn":0,
            "vslbhealth":0,
            "primaryipaddress":"0.0.0.0",
            ...
        }
    ]
}
```

Note: Not all NetScaler features and resources have statistic objects associated with them.

Managing AppExpert Applications

To export an AppExpert application, specify the parameters needed for the export operation in the `apptemplateinfo` object. Optionally, you can specify basic information about the AppExpert application template, such as the author of the configuration, a summary of the template functionality, and the template version number, in the `template_info` object. This information is stored as part of the template file that is created.

To export an AppExpert application named "MyApp1":

- **URL.** http://10.102.29.60/nitro/v1/config/apptemplateinfo?action=export
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.apptemplateinfo+json
```

  - **Payload**

```
{
    "apptemplateinfo":
    {
       "appname":"MyApp1",
       "apptemplatefilename":"BizAp.xml",
       "template_info":
       {
           "templateversion_major":"2",
           "templateversion_minor":"1",
           "author":"XYZ",
           "introduction":"Intro",
```

```
            "summary":"Summary"
        },
    }
}
```

To import an AppExpert application, specify the parameters needed for the import operation in the `apptemplateinfo` object.

To import an AppExpert application named "MyApp1":

- **URL.** http://10.102.29.60/nitro/v1/config/apptemplateinfo?action=import
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.apptemplateinfo+json
X-NITRO-ONERROR:rollback
```

  - **Payload**

```
{
   "apptemplateinfo":
   {
       "apptemplatefilename":"BizAp.xml",
       "deploymentfilename":"BizAp_deployment.xml",
       "appname":"MyApp1"
   }
}
```

To import an AppExpert application by specifying different deployment settings:

- **URL.** http://10.102.29.60/nitro/v1/config/apptemplateinfo?action=import
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Cookie:NITRO_AUTH_TOKEN=tokenvalue
Content-Type:application/
vnd.com.citrix.netscaler.apptemplateinfo+json
X-NITRO-ONERROR:rollback
```

  - **Payload**

```
{
   "apptemplateinfo":
   {
      "apptemplatefilename":"BizAp.xml",
      "appname":"Myapp2"
       "deploymentinfo":
       {
```

```
            "appendpoint":
            [
                {
                    "ipv46":"11.2.3.8",
                    "port":80,
                    "servicetype":"HTTP"
                }
            ],
            "service":
            [
                {
                    "ip":"12.3.3.15",
                    "port":80,
                    "servicetype":"SSL"
                },
                {
                    "ip":"14.5.5.16",
                    "port":443,
                    "servicetype":"SSL"
                }
            ],
        }
    }
}
```

Performing File Operations

NetScaler operations such as configuring SSL certificates requires the input files to be available locally on the NetScaler appliance. NITRO allows you to perform file operations such as uploading file to the NetScaler, retrieving a list of files and the file content from the NetScaler, and also delete files from the NetScaler. These operations can be performed for files of type: txt, cert, req, xml, and key.

Note: NITRO file operations can be performed only on files of size less than or equal to 2 MB.

Uploading a File to the NetScaler

To upload a file to the NetScaler, specify a name for the file, the location where the file must be created on the NetScaler, and the content of the file.

- **URL.** http://<netscaler-ip-address>/nitro/v1/config/systemfile
- **HTTP Method.** POST
- **Request.**
  - **Header**

```
Content-Type:application/
vnd.com.citrix.netscaler.systemfile+json
```

- **Payload**

```
{
    "systemfile":
    {
        "filename": "cert1.crt",
        "filelocation": "/nsconfig/ssl/",
        "filecontent":"VGhpcyBpcyBteSBmaWxl",
        "fileencoding": "BASE64"
    }
}
```

Retrieving the Files from a NetScaler Directory

Specify the directory path from which you want to get a list of files.

- **URL.** http://<netscaler-ip-address>/nitro/v1/config/systemfile?args=filelocation:<path>
- **HTTP Method.** GET
- **Response**
  - **Header**

```
HTTP/1.0 200 OK
Accept:application/vnd.com.citrix.netscaler.systemfile_list+json
```

  - **Payload**

```
{
    "errorcode": 0,
    "message": "Done",
    "severity": "NONE",
    "systemfile":
    [
        {
            "filename": "ns-root.key",
            "filelocation": "\/nsconfig\/ssl",
            "fileaccesstime": "Tue Jan 14 19:27:01 2014",
            "filemodifiedtime": "Tue Nov  5 17:16:00 2013"
        },
        {
            "filename": "ns-root.req",
            "filelocation": "\/nsconfig\/ssl",
            "fileaccesstime": "Tue Jan 14 19:27:01 2014",
            "filemodifiedtime": "Tue Nov  5 17:16:00 2013"
        }
    ]
}
```

Retrieving Contents of a Specific File

To retrieve the contents of a file, specify the filename and its location.

- **URL.** http://<netscaler-ip-address>/nitro/v1/config/systemfile/<filename>?args=filelocation:<path>

- **HTTP Method.** GET
- **Response.**
  - **Header**

```
HTTP/1.0 200 OK
Accept:application/vnd.com.citrix.netscaler.systemfile+json
```

  - **Payload**

```
{
    "errorcode": 0,
    "message": "Done",
    "severity": "NONE",
    "systemfile":
    [
        {
            "filename": "ns-root.key",
            "filelocation": "\/nsconfig\/ssl",
            "filecontent":
"LS0tLS1CRUdJTiBSU0EgUFJJVkFU0tLQo=",
            "fileencoding": "BASE64",
            "fileaccesstime": "Tue Jan 14 19:27:01 2014",
            "filemodifiedtime": "Tue Nov  5 17:16:00 2013"
        }
    ]
}
```

Deleting a File from the NetScaler

To delete a file from the NetScaler, specify the filename along with the full file path.

- **URL.** http://<netscaler-ip-address>/nitro/v1/config/systemfile/<filename>?args=filelocation:<path>
- **HTTP Method.** DELETE
- **Response.**
  - **Header**

```
HTTP/1.0 200 OK
```

  - **Payload**

```
{
    "errorcode": 0,
    "message": "Done",
    "severity": "NONE"
}
```

Handling Exceptions

The response header provides the status of an operation by using HTTP status codes and the response payload provides the requested resource object (for GET method) and

error details (for unsuccessful operation). NITRO does not provide a response payload for successful POST, PUT and DELETE methods. For successful GET method, the response payload consists only the requested resource object.

The following table provides the HTTP status codes:

Status	HTTP Status Code	Description
Success	200 OK	Request successfully executed.
	201 CREATED	Entity created.
Failure	400 Bad Request	Incorrect request provided.
	401 unauthorized	Not provided login credentials.
	403 forbidden	User is unauthorized
	404 Not Found	User is trying to access a resource not present in the NetScaler.
	405 Method Not Allowed	User is trying to access request methods not supported by NITRO.
	406 Not Acceptable	None of the values supplied by the user in the Accept header can be satisfied by the server.
	409 Conflict	The resource already exists on the NetScaler.
	503 Service Unavailable	The service is not available.
	599	NetScaler specific error code.
Warning	209 X-NITRO-WARNING	Warnings are captured by specifying the login URL as `http://<nsip>/nitro/v1/config/login/?warning=yes`.
Combination of success and failure (for bulk operation with X-NITRO-ONERROR set as continue)	207 Multi Status	Some commands are executed successfully and some have failed.

Note: The content-type in the response header of an unsuccessful operation, consists of error MIME type instead of resource MIME type.

For a more detailed description of the error codes, see the API reference available in the `<NITRO_SDK_HOME>/doc` folder.

Unsupported NetScaler Operations

Some NetScaler operations that are available through the command line interface and through the configuration utility, are not available through NITRO APIs. The following list provides the NetScaler operations not supported by NITRO:

- install API
- diff API on nsconfig resource
- UI-internal APIs (update, unset, and get)
- show ns info
- Application firewall APIs:
  - `importwsdl`
  - `importcustom`
  - `importxmlschema`
  - `importxmlerrorpage`
  - `importhtmlerrorpage`
  - `rmwsdl`
  - `rmcustom`
  - `rmxmlschema`
  - `rmxmlerrorpage`
  - `rmhtmlerrorpage`
- CLI-specific APIs:
  - `ping`
  - `ping6`
  - `traceroute`
  - `traceroute6`
  - `nstrace`
  - `scp`
  - `configaudit`
  - `show defaults`
  - `show permission`
  - `batch`

- source