Overview

The documentation pages in this section describe the RESTful APIs included with Cascade Gateway products. It is assumed that the reader has practical knowledge of RESTful APIs, so the documentation does not go into detail about what REST is and how to use it. Instead the documentation focuses on what data can be accessed and how to access it.

The following information can be accessed via the API:

- Perform System operations
- Information about system users

Details about REST resources can be found in the Resources section. This overview continues with how to run reports and retrieve data from them.

Authentication

All REST requests must be authenticated. The Authentication section of the Common 1.0 API describes which authentication methods are presently supported. There are also examples that show how to use each of the different authentication methods.

Resources

Flow_Sources: List flow sources

Get a list of flow sources.

GET https://{device}/api/gateway/1.4/flow_sources

Authorization

This request requires authorization.

Response Body

On success, the server returns a response body with the following structure:
[{
  "flows_received_last_min": number,
  "last_min": number,
  "ipaddr": string,
  "versions": string,
  "flow_type": string
}]

Example:
[{
  "flows_received_last_min": 13299,
  "ipaddr": "10.32.4.41",
  "last_min": 1399317542,
  "flow_type": "NetFlow",
  "versions": "9.2"
},
{
  "flows_received_last_min": 374,
  "ipaddr": "10.32.7.31",
  "last_min": 1399317542,
  "flow_type": "NetFlow",
  "versions": "9.2"
},
{
  "flows_received_last_min": 500,
  "ipaddr": "10.32.7.35",
  "last_min": 1399317542,
  "flow_type": "NetFlow",
  "versions": "9.2"
}]

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GatewayStatsFlowSources</td>
<td>&lt;array of &lt;object&gt;&gt;</td>
<td>List of flow sources.</td>
<td></td>
</tr>
<tr>
<td>GatewayStatsFlowSources [GatewayStatsFlowSource].flows_received_last_min</td>
<td>&lt;number&gt;</td>
<td>Flows received (last minute).</td>
<td>Optional</td>
</tr>
<tr>
<td>GatewayStatsFlowSources [GatewayStatsFlowSource].last_min</td>
<td>&lt;number&gt;</td>
<td>Last heard from.</td>
<td>Optional</td>
</tr>
<tr>
<td>GatewayStatsFlowSources [GatewayStatsFlowSource].ipaddr</td>
<td>&lt;string&gt;</td>
<td>IP address.</td>
<td></td>
</tr>
</tbody>
</table>

**Flow_Sources: Get flow source**

Get a flow source by IP Address.

GET https://{device}/api/gateway/1.4/flow_sources/{ipaddr}

**Authorization**

This request requires authorization.

**Response Body**

On success, the server returns a response body with the following structure:

JSON
Example:
```
{
  "flows_received_last_min": 962,
  "ipaddr": "10.32.1.41",
  "last_min": 1392392760,
  "flow_type": "NetFlow",
  "versions": "9.2"
}
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>GatewayStatsFlowSource</td>
<td>object</td>
<td>Flow source.</td>
<td></td>
</tr>
<tr>
<td>GatewayStatsFlowSource</td>
<td>&lt;number&gt;</td>
<td>Flows received (last minute).</td>
<td>Optional</td>
</tr>
<tr>
<td>GatewayStatsFlowSource</td>
<td>&lt;number&gt;</td>
<td>Last heard from.</td>
<td>Optional</td>
</tr>
<tr>
<td>GatewayStatsFlowSource.ipaddr</td>
<td>&lt;string&gt;</td>
<td>IP address.</td>
<td></td>
</tr>
<tr>
<td>GatewayStatsFlowSource.versions</td>
<td>&lt;string&gt;</td>
<td>Versions.</td>
<td></td>
</tr>
<tr>
<td>GatewayStatsFlowSource.flow_type</td>
<td>&lt;string&gt;</td>
<td>Flow type.</td>
<td></td>
</tr>
</tbody>
</table>

**Flow_Sources: Delete flow source**
Delete a flow source by IP Address.

```text
DELETE https://{device}/api/gateway/1.4/flow_sources/{ipaddr}
```

**Authorization**
This request requires authorization.

**Response Body**
On success, the server does not provide any body in the responses.

**Ping: Ping**
Simple test of service availability.

```text
GET https://{device}/api/gateway/1.4/ping
```

**Authorization**
This request requires authorization.

**Response Body**
On success, the server does not provide any body in the responses.

**System: Start all processes (one module)**
Start all system processes on one module on Enterprise systems. The operation is asynchronous. Use "GET system/{module}/status" to poll for status. The {module} can be either the IP Address or the module name.

```text
POST https://{device}/api/gateway/1.4/system/{module}/start
```

**Authorization**
This request requires authorization.

**Request Body**
Do not provide a request body.
Response Body

On success, the server does not provide any body in the responses.

System: Get status of all processes

Get status of all system processes. On Enterprise systems, get system process statuses on all modules.

GET https://{device}/api/gateway/1.4/system/status

Authorization

This request requires authorization.

Response Body

On success, the server returns a response body with the following structure:

```json
{
    "SystemStatus": [
        {
            "process_id": "25096",
            "process_name": "memmonitor",
            "status": "Running"
        },
        {
            "process_name": "healthd",
            "status": "Stopped"
        },
        {
            "process_id": "25092",
            "process_name": "diskmon",
            "status": "Running"
        },
        {
            "process_id": "25123",
            "process_name": "dispatcher",
            "status": "Running"
        },
        {
            "process_name": "analyzer",
            "status": "Stopped"
        }
    ]
}
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SystemStatus</td>
<td>&lt;array of &lt;object&gt;&gt;</td>
<td>SystemStatus object.</td>
<td></td>
</tr>
<tr>
<td>SystemStatus[SystemProcess].process_id</td>
<td>&lt;string&gt;</td>
<td>Process ID.</td>
<td>Optional</td>
</tr>
<tr>
<td>SystemStatus[SystemProcess].process_name</td>
<td>&lt;string&gt;</td>
<td>Process name.</td>
<td></td>
</tr>
<tr>
<td>SystemStatus[SystemProcess].module_name</td>
<td>&lt;string&gt;</td>
<td>Module name. Available on Enterprise systems only.</td>
<td>Optional</td>
</tr>
<tr>
<td>SystemStatus[SystemProcess].status</td>
<td>&lt;string&gt;</td>
<td>Process status.</td>
<td>Values: Running, Stopped</td>
</tr>
<tr>
<td>SystemStatus[SystemProcess].module_ipaddr</td>
<td>&lt;string&gt;</td>
<td>Module IP address. Available on Enterprise systems only.</td>
<td>Optional</td>
</tr>
</tbody>
</table>

System: Kill all processes

Kill all system processes. The operation is asynchronous. Use "GET system/status" to poll for status. On Enterprise systems, kill system processes on all modules. Warning: this operation can result in data being corrupted.
POST https://{device}/api/gateway/1.4/system/kill

Authorization
This request requires authorization.

Request Body
Do not provide a request body.

Response Body
On success, the server does not provide any body in the responses.

System: Restart all processes
Restart all system processes. The operation is asynchronous. Use "GET system/status" to poll for status. On Enterprise systems, stop system processes on all modules.

POST https://{device}/api/gateway/1.4/system/restart

Authorization
This request requires authorization.

Request Body
Do not provide a request body.

Response Body
On success, the server does not provide any body in the responses.

System: Start all processes
Start all system processes. The operation is asynchronous. Use "GET system/status" to poll for status. On Enterprise systems, start system processes on all modules.

POST https://{device}/api/gateway/1.4/system/start

Authorization
This request requires authorization.

Request Body
Do not provide a request body.

Response Body
On success, the server does not provide any body in the responses.

System: Restart all processes (one module)
Restart all system processes on one module on Enterprise systems. The operation is asynchronous. Use "GET system/{module}/status" to poll for status. The {module} can be either the IP Address or the module name.

POST https://{device}/api/gateway/1.4/system/{module}/restart

Authorization
This request requires authorization.

Request Body
Do not provide a request body.

Response Body
On success, the server does not provide any body in the responses.
System: Stop all processes (one module)

Stop all system processes on one module on Enterprise systems. The operation is asynchronous. Use "GET system/{module}/status" to poll for status. The {module} can be either the IP Address or the module name.

POST https://{device}/api/gateway/1.4/system/{module}/stop

Authorization
This request requires authorization.

Request Body
Do not provide a request body.

Response Body
On success, the server does not provide any body in the responses.

System: Get status of all processes (one module)

Get status of all system processes on one module on Enterprise systems. The {module} can be either the IP Address or the module name.

GET https://{device}/api/gateway/1.4/system/{module}/status

Authorization
This request requires authorization.

Response Body
On success, the server returns a response body with the following structure:

```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>SystemStatus</td>
<td>&lt;array of</td>
<td>SystemStatus object.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>&lt;object&gt;&gt;</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
```

Example:
```
[
  {
    "process_id": "25096",
    "process_name": "memmonitor",
  },
  {
    "process_name": "healthd",
    "status": "Stopped"
  },
  {
    "process_id": "25092",
    "process_name": "diskmon",
    "status": "Running"
  },
  {
    "process_id": "25123",
    "process_name": "dispatcher",
    "status": "Running"
  },
  {
    "process_name": "analyzer",
    "status": "Stopped"
  }
]
```
### System: Shutdown

Shutdown the system. The operation is asynchronous.

**Authorization**

This request requires authorization.

**Request Body**

Do not provide a request body.

**Response Body**

On success, the server does not provide any body in the responses.

---

### System: Reboot

Reboot the system. The operation is asynchronous.

**Authorization**

This request requires authorization.

**Request Body**

Do not provide a request body.

**Response Body**

On success, the server does not provide any body in the responses.

---

### System: Stop all processes

Stop all system processes. The operation is asynchronous. Use "GET system/status" to poll for status. On Enterprise systems, stop system processes on all modules.

**Authorization**

This request requires authorization.

**Request Body**

Do not provide a request body.

**Response Body**

On success, the server does not provide any body in the responses.

---

### System: Kill all processes (one module)

Kill all system processes on one module on Enterprise systems. The operation is asynchronous. Use "GET system/{module}/status" to poll for status. The {module} can be either the IP Address or the module name. Warning: this operation can result in data being corrupted.
**Authorization**
This request requires authorization.

**Request Body**
Do not provide a request body.

**Response Body**
On success, the server does not provide any body in the responses.

---

**Users: List users**
Get a list of user accounts.

**Authorization**
This request requires authorization.

**Response Body**
On success, the server returns a response body with the following structure:

```json
```
```json
{
  "enabled": string,
  "last_name": string,
  "id": number,
  "last_login": number,
  "authentication_type": string,
  "username": string,
  "authorization_type": string,
  "role": string,
  "first_name": string,
  "last_access": number,
  "view_packet_details": string,
  "last_authentication": number,
  "view_user_information": string,
  "login_timeout": number
}
```

Example:
```json
[
  {
    "username": "admin",
    "last_authentication": 1352313328,
    "first_name": "Jonh",
    "last_name": "Smith",
    "authorization_type": "Local",
    "enabled": true,
    "view_user_information": true,
    "authentication_type": "Local",
    "role": "Administrator",
    "login_timeout": 900,
    "last_login": 1352313328,
    "last_access": 1352313328,
    "id": 123
  },
  {
    "username": "admin2",
    "last_authentication": 1352313328,
    "first_name": "Mark",
    "last_name": "Greg",
    "authorization_type": "Local",
    "enabled": true,
    "view_user_information": true,
    "authentication_type": "Local",
    "role": "Administrator",
    "login_timeout": 900,
    "last_login": 1352313328,
    "last_access": 1352313328,
    "id": 124
  }
]
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Users</td>
<td>&lt;array of &lt;object&gt;&gt;</td>
<td>List of user accounts on the system.</td>
<td></td>
</tr>
<tr>
<td>Users[User]</td>
<td>&lt;object&gt;</td>
<td>User account.</td>
<td>Optional</td>
</tr>
<tr>
<td>Users[User].enabled</td>
<td>&lt;string&gt;</td>
<td>Boolean flag indicating if the user account is enabled.</td>
<td></td>
</tr>
<tr>
<td>Users[User].id</td>
<td>&lt;number&gt;</td>
<td>Numeric ID of the user that the system uses internally and in the API.</td>
<td></td>
</tr>
<tr>
<td>Users[User].last_login</td>
<td>&lt;number&gt;</td>
<td>Time of last login. Unix time (epoch).</td>
<td></td>
</tr>
<tr>
<td>Users[User].authentication_type</td>
<td>&lt;string&gt;</td>
<td>Type of authentication for the user, such as Local or RADIUS.</td>
<td>Values: Local, Remote</td>
</tr>
<tr>
<td>Users[User].username</td>
<td>&lt;string&gt;</td>
<td>User name (short name) that identifies the user to the system, such as 'admin'.</td>
<td></td>
</tr>
<tr>
<td>Users[User].authorization_type</td>
<td>&lt;string&gt;</td>
<td>Type of authorization for the user, such as Local or RADIUS.</td>
<td>Values: Local, Remote</td>
</tr>
<tr>
<td>Users[User].role</td>
<td>&lt;string&gt;</td>
<td>Role of the user. Defines permissions.</td>
<td>Values: Developer, Administrator, Operator, Monitor, Event_Viewer, Dashboard_Viewer</td>
</tr>
<tr>
<td>Users[User].first_name</td>
<td>&lt;string&gt;</td>
<td>First name of the user.</td>
<td></td>
</tr>
<tr>
<td>Users[User].last_access</td>
<td>&lt;number&gt;</td>
<td>Time of last access to the system. Unix time (epoch).</td>
<td></td>
</tr>
<tr>
<td>Users[User].view_packet_details</td>
<td>&lt;string&gt;</td>
<td>Boolean flag indicating if the user has access to packet data.</td>
<td>Optional</td>
</tr>
<tr>
<td>Users[User].last_authentication</td>
<td>&lt;number&gt;</td>
<td>Time of last authentication. Unix time (epoch).</td>
<td></td>
</tr>
</tbody>
</table>
**Users: Re-authenticate user**

Re-authenticate user account. Requires basic authentication.

GET https://{device}/api/gateway/1.4/users/re_authenticate

**Authorization**

This request requires authorization.

**Response Body**

On success, the server does not provide any body in the responses.

**Users: Get user**

User account by user ID.

GET https://{device}/api/gateway/1.4/users/{user_id}

**Authorization**

This request requires authorization.

**Response Body**

On success, the server returns a response body with the following structure:

```
{  
  "enabled": string,  
  "last_name": string,  
  "id": number,  
  "last_login": number,  
  "authentication_type": string,  
  "username": string,  
  "authorization_type": string,  
  "role": string,  
  "first_name": string,  
  "last_access": number,  
  "view_packet_details": string,  
  "last_authentication": number,  
  "view_user_information": string,  
  "login_timeout": number  
}
```

Example:

```
{  
  "enabled": "true",  
  "last_authentication": 1352313328,  
  "first_name": "Jonh",  
  "last_name": "Smith",  
  "authorization_type": "Local",  
  "enabled": true,  
  "view_user_information": true,  
  "authentication_type": "Local",  
  "role": "Administrator",  
  "login_timeout": 900,  
  "last_login": 1352313328,  
  "last_access": 1352313328,  
  "id": 123  
}
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>&lt;object&gt;</td>
<td>User account</td>
<td></td>
</tr>
<tr>
<td>User.enabled</td>
<td>&lt;string&gt;</td>
<td>Boolean flag indicating if the user account is enabled.</td>
<td></td>
</tr>
<tr>
<td>User.last_name</td>
<td>&lt;string&gt;</td>
<td>Last name of the user.</td>
<td></td>
</tr>
</tbody>
</table>
Users: Test RADIUS user

Test a RADIUS user.

POST https://{device}/api/gateway/1.4/users/radius/test_user?password={string}&username={string}

Authorization
This request requires authorization.

Parameters

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>password</td>
<td>&lt;string&gt;</td>
<td>RADIUS password.</td>
<td></td>
</tr>
<tr>
<td>username</td>
<td>&lt;string&gt;</td>
<td>RADIUS username.</td>
<td></td>
</tr>
</tbody>
</table>

Request Body
Provide a request body with the following structure:

```json
{
  "password": "string",
  "username": "string"
}
```

Response Body
On success, the server returns a response body with the following structure:

```json
```
Example:

```json
{
  "error_message": 
  "authenticated": true,
  "server_type": 2,
  "permission_id": 
  "permission": 
  "role_id": 0,
  "role": 
  "authorized": false,
  "server_ip": "10.38.8.112:1812",
  "attributes": [
    {
      "25": "operatorClass"
    },
    {
      "25": "monitorClass"
    },
    {
      "25": "eventviewerClass"
    },
    {
      "17164": "unMappedRole"
    },
    {
      "17164": "monitorCascade"
    },
    {
      "17164": "eventviewerCascade"
    },
    {
      "17164": "dashboardCascade"
    },
    {
      "25": "DBAccess"
    },
    {
      "25": "dashboardClass"
    },
    {
      "17164": "AbC10-!@#$%^&*()_+{}[]:;<>?/.'z"
    },
    {
      "17164": "operatorCascade"
    },
    {
      "LOGIN_SERVER": "10.38.8.112:1812"
    },
    {
      "25": "adminClass1"
    },
    {
      "25": "unMappedClass"
    },
    {
      "25": "eventviewerClass"
    },
    {
      "17164": "adminCascade"
    }
  ],
  "details": 
}
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>role_id</td>
<td>number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>error_message</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>permission</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>server_type</td>
<td>number</td>
<td></td>
<td></td>
</tr>
<tr>
<td>role</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>details</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>permission_id</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>server_ip</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>authenticated</td>
<td>string</td>
<td></td>
<td></td>
</tr>
<tr>
<td>attributes</td>
<td>[ ]</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Property Name: The name of the property or attribute.
Type: The data type of the property or attribute.
Description: A description of the property or attribute.
Notes: Additional notes or comments about the property or attribute.
RemoteTestUserResponse object.

- **role_id**: <number> - Matched role ID.
- **error_message**: <string> - Error message.
- **permission**: <string> - Matched permission name.
- **server_type**: <number> - Indicates the type of the server being tested: RADIUS(2) or TACACS+(3).
- **role**: <string> - Matched role name.
- **details**: <string> - Remote user test details.
- **permission_id**: <string> - Matched permission ID.
- **server_ip**: <string> - Remote Server IP address.
- **authenticated**: <string> - Flag indicating if the remote user was authenticated.
- **attributes**: <array of <object>> - Attributes of Remote Test User Response. Optional
  - **attributes**: <object> - Remote attribute. Optional
  - **attributes**: <RemoteAttributes> - Remote attribute value. Optional
- **authorized**: <string> - Flag indicating if the remote user was authorized (as Administrator, Monitor, etc).

### Users: Test TACACS+ server
Test the connection to a TACACS+ server.

GET https://{device}/api/gateway/1.4/users/tacacs/test_server

**Authorization**
This request requires authorization.

**Parameters**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>&lt;string&gt;</td>
<td>TACACS+ server identifier, example server=IP:PORT.</td>
<td></td>
</tr>
</tbody>
</table>

**Response Body**
On success, the server returns a response body with the following structure:

```json
{
  "success": string,
  "message": string
}
```

Example:
```json
{
  "message": "Connection attempt succeeded",
  "success": true
}
```

RemoteTestServerResponse object.

- **success**: <string> - Flag indicating if the remote server test was successful.
- **message**: <string> - Response message.

### Users: Test TACACS+ user
Test a TACACS+ user.

POST https://{device}/api/gateway/1.4/users/tacacs/test_user?password={string}&username={string}

**Authorization**
This request requires authorization.
Parameters

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>password</td>
<td>&lt;string&gt;</td>
<td>TACACS+ password.</td>
<td></td>
</tr>
<tr>
<td>username</td>
<td>&lt;string&gt;</td>
<td>TACACS+ username.</td>
<td></td>
</tr>
</tbody>
</table>

Request Body

Provide a request body with the following structure:

```json
{
  "password": string,
  "username": string
}
```

Example:
```
{
  "username": "testusername",
  "password": "testpassword"
}
```

Response Body

On success, the server returns a response body with the following structure:

```json
```

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>RemoteTestUserRequest</td>
<td>&lt;object&gt;</td>
<td>RemoteTestUserRequest object.</td>
<td></td>
</tr>
<tr>
<td>RemoteTestUserRequest.password</td>
<td>&lt;string&gt;</td>
<td>password.</td>
<td></td>
</tr>
<tr>
<td>RemoteTestUserRequest.username</td>
<td>&lt;string&gt;</td>
<td>user name.</td>
<td></td>
</tr>
</tbody>
</table>
Example:
{
  "error_message": "",
  "authenticated": true,
  "server_type": 2,
  "permission_id": "",
  "permission": "",
  "role_id": 0,
  "role": "",
  "authorized": false,
  "server_ip": "10.38.8.112:1812",
  "attributes": [
    {
      "25": "operatorClass"
    },
    {
      "25": "monitorClass"
    },
    {
      "25": "eventviewerClass"
    },
    {
      "17164": "unMappedRole"
    },
    {
      "17164": "monitorCascade"
    },
    {
      "17164": "eventviewerCascade"
    },
    {
      "17164": "dashboardCascade"
    },
    {
      "25": "DBAccess"
    },
    {
      "25": "dashboardClass"
    },
    {
      "17164": "AbC10-!@#$%^&*()_+{}][;<>?/.'z"
    },
    {
      "17164": "operatorCascade"
    },
    {
      "LOGIN_SERVER": "10.38.8.112:1812"
    },
    {
      "25": "adminClass1"
    },
    {
      "25": "unMappedClass"
    },
    {
      "25": "eventviewerClass"
    },
    {
      "17164": "adminCascade"
    },
  ],
  "details": "Using 10.38.8.112:1812 - Unable to match a role."
}

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>


**Users: Test RADIUS server**

Test the connection to a RADIUS server.

GET https://{device}/api/gateway/1.4/users/radius/test_server

**Authorization**

This request requires authorization.

**Parameters**

<table>
<thead>
<tr>
<th>Property Name</th>
<th>Type</th>
<th>Description</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>server</td>
<td>&lt;string&gt;</td>
<td>RADIUS server identifier, example server=IP:PORT.</td>
<td></td>
</tr>
</tbody>
</table>

**Response Body**

On success, the server returns a response body with the following structure:

```
{
  "success": string,
  "message": string
}
```

Example:

```
{
  "message": "Connection attempt succeeded",
  "success": true
}
```

**Error Codes**

In the event that an error occurs while processing a request, the server will respond with appropriate HTTP status code and additional information in the response body:

```
{
  "error_id": "{error identifier}"
}
```
The table below lists the possible errors and the associated HTTP status codes that may returned.

<table>
<thead>
<tr>
<th>Error ID</th>
<th>HTTP Status</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>INTERNAL_ERROR</td>
<td>500</td>
<td>Internal server error.</td>
</tr>
<tr>
<td>AUTH_REQUIRED</td>
<td>401</td>
<td>The requested resource requires authentication.</td>
</tr>
<tr>
<td>AUTH_INVALID_CREDENTIALS</td>
<td>401</td>
<td>Invalid username and/or password.</td>
</tr>
<tr>
<td>AUTH_INVALID_SESSION</td>
<td>401</td>
<td>Session ID is invalid.</td>
</tr>
<tr>
<td>AUTH_EXPIRED_PASSWORD</td>
<td>403</td>
<td>The password must be changed. Access only to password change resources.</td>
</tr>
<tr>
<td>AUTH_DISABLED_ACCOUNT</td>
<td>403</td>
<td>Account is either temporarily or permanently disabled.</td>
</tr>
<tr>
<td>AUTH_FORBIDDEN</td>
<td>403</td>
<td>User is not authorized to access the requested resource.</td>
</tr>
<tr>
<td>AUTH_INVALID_TOKEN</td>
<td>401</td>
<td>OAuth access token is invalid.</td>
</tr>
<tr>
<td>AUTH_EXPIRED_TOKEN</td>
<td>401</td>
<td>OAuth access token is expired.</td>
</tr>
<tr>
<td>AUTH_INVALID_CODE</td>
<td>401</td>
<td>OAuth access code is invalid.</td>
</tr>
<tr>
<td>AUTH_EXPIRED_CODE</td>
<td>401</td>
<td>OAuth access code is expired.</td>
</tr>
<tr>
<td>RESOURCE_NOT_FOUND</td>
<td>404</td>
<td>Requested resource was not found.</td>
</tr>
<tr>
<td>HTTP_INVALID_METHOD</td>
<td>405</td>
<td>Requested method is not available for this resource.</td>
</tr>
<tr>
<td>HTTP_INVALID_HEADER</td>
<td>400</td>
<td>An HTTP header was malformed.</td>
</tr>
<tr>
<td>REQUEST_INVALID_INPUT</td>
<td>400</td>
<td>Malformed input structure.</td>
</tr>
<tr>
<td>URI_INVALID_PARAMETER</td>
<td>400</td>
<td>URI parameter is not supported or malformed.</td>
</tr>
<tr>
<td>URI_MISSING_PARAMETER</td>
<td>400</td>
<td>Missing required parameter.</td>
</tr>
</tbody>
</table>