COMMSCOPE®
RUCKUS ®

Cloudpath Enrollment System REST API v2 User Guide, 5.7

Supporting Cloudpath Software Release 5.7

Copyright, Trademark and Proprietary Rights Information

Contents

Cloudpath REST API v2 Basics

Overview

REST API v2 is a RESTful API designed for external consumer use. It uses Json Web Tokens for authentication, and also requires an API Key. This authentication allows for auditing, and provides greater security than only requiring an API key.

> **NOTE**
> Basic knowledge of REST API v2 and Json format are necessary.

Cloudpath provides REST APIs to enable integration with existing processes or third-party applications. REST APIs allow the system to actively notify external systems (outbound calls) and to be queried and manipulated by external systems (inbound calls).

REST APIs are resource-based, and support the following HTTP verbs:

- GET (to get information about a resource)
- POST (to create a resource)
- PUT (to change the state of a resource)
- DELETE (to delete a resource)

REST APIs also return errors by using HTTP return codes. Refer to Return Codes and Error Response Format on page 10.

Root URI: The root URI is in the form of:

```
<Cloudpath FQDN>/admin/apiv2/{APIKey}
```

Format for Variables

Variables in Cloudpath are in the following format: ${MAC_ADDRESS}, ${USERNAME}, or ${IP_ADDRESS}. See the Enrollment Variables section on the Enrollment Record page in the UI to view available variables, variable syntax, and expected values. Also, navigate to **Configuration > DPSK Pools**, and go to the window to add or edit a pool, and you can view all the available variables from a drop-down list.
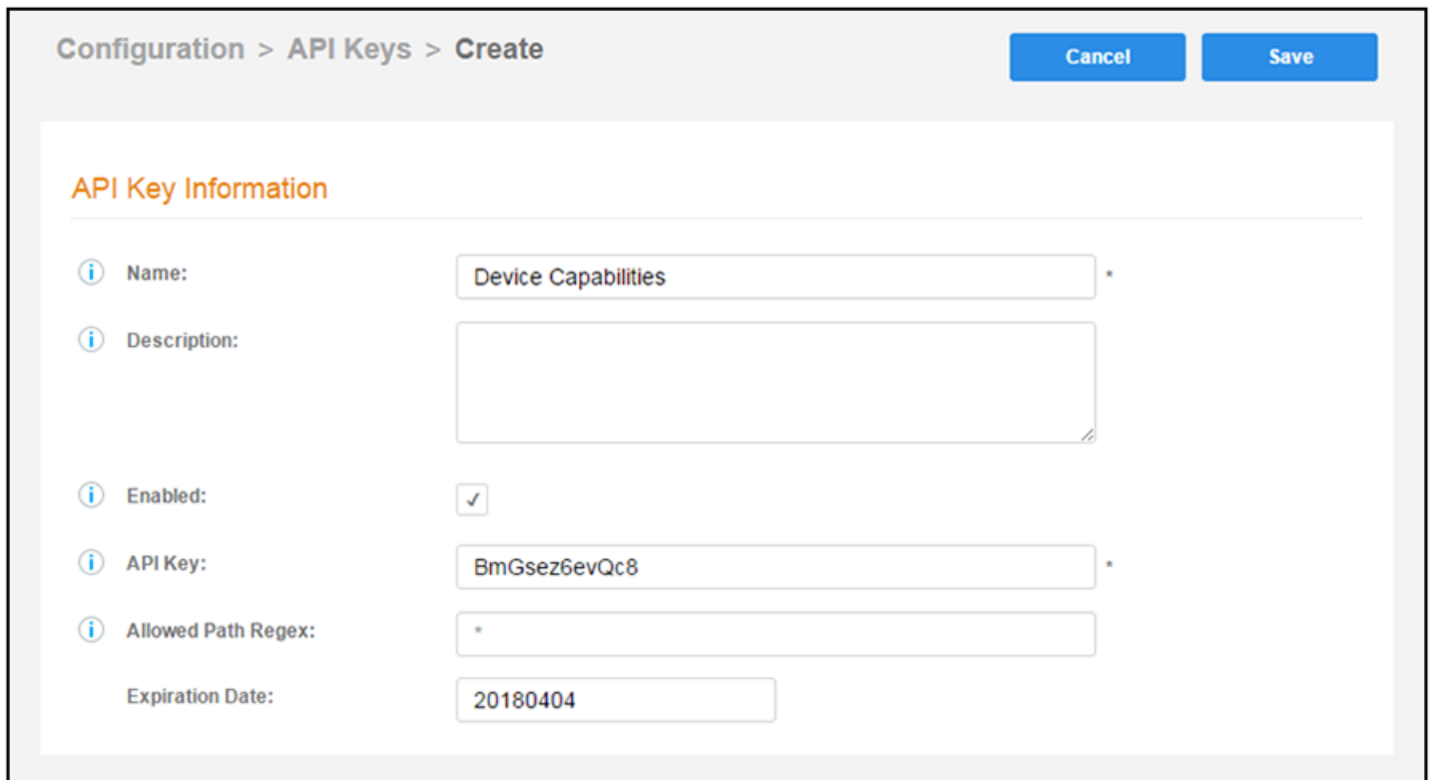
Only variables which are populated at the time of the notification event are sent. MAC addresses and other variables may be populated through events other than onboarding, such as data prompts.

Creating an API Key

For Cloudpath to honor inbound requests from external systems, the external system must authenticate using an API key. The API key is part of the REST API query used to retrieve information from the Cloudpath system. Authentication is a combination of this API key and an authentication with the Token API. To create an API Key, perform the following steps:

1. Go to **Configuration** > **API Keys** .

2. Click **API Keys**

FIGURE 1 Create API Key

Cloudpath provides a generated **APIKey**, or you can specify a custom key. An API key must be URL-safe.

The **Allowed Path Regex** is a regex specifying the paths allowed to be called with this API key. For example,/admin/api/.*/certificate/.* can query anything in the 'certificate' realm. If left blank, there is no restriction.

For the **Expiration Date**, the Cloudpath system defaults to a one-year expiration, but this value can be modified.

Security

The Cloudpath REST v2 APIs use Json Web Tokens (JWT) to secure all endpoints, in addition to using the API key.

The user calls "..../admin/apiv2/*APIKey*/token", with a user name and password, and, if authenticated, a JWT token is returned to the user in the authentication response body.

NOTE

You must include an active JWT token in the "Authorization" HTTP request header of all subsequent requests.

The token has a five-minute timeout and can be refreshed by requesting a new token while the current token is still valid. In addition, the "..../ admin/apiv2/*APIKey*/token" endpoint provides basic information about the token's expiration.

NOTE

If you change the API key in the Cloudpath UI, all tokens become invalid, and new authentications are required. In addition, if you disable the API key, all API requests are rejected, even if the token is still valid and the user has already been authenticated.

The following table lists the properties that can be used in the body of a token request and/or are provided in the response body.

TABLE 1 Resource: Token

Property Name	Type	Editable	Filter or Sort	Optional	Description
userName	String	Yes	Neither	Required only on a PUT call request, but always returned in the response to a PUT or a POST.	The username of the person trying to use the API. The user needs to have the account admin role for the account being requested.
password	String	Yes	Neither	Required only for a POST request.	The password for the user. Will never be returned in the response body.
token	String	No	Neither	Yes (Never needed in request body)	The token returned for this user. Will be ignored on a POST and a PUT.
expireDateTime	String < DateTime>	No	Neither	Yes (Never needed in request body)	Expiration time and date for the token returned to the user, given in iso80601 date/time format.

Security URIs and Actions

The following table indicates the URIs that you can append to the root URI, along with the actions you can take, and possible return codes you can receive.

TABLE 2 Security URIs and Actions

URI	Action	Description	Return Codes
/token	POST	Authenticates the user and returns a valid security token. The body must include the username and password.	401 - Unauthorized 201 - Created
	PUT	Provides an updated token, without requiring a new login. The body must be empty and the current valid token must be in the "Authorization" header.	422 - Entity cannot be processed (if the body is not empty). 406 - Not Acceptable (token has expired) 401 - Unauthorized

Example of Generating an Authorization Token

URI:

This example uses the following URI:

```
https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/token
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)
- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

Request Headers:

The following table shows the required request header for this URI example:

TABLE 3 Request Header

Key	Value
Content-type	application/json

Request Body:

An example of the request body is shown below (this provides administrative login credentials to the Cloudpath system being used):

```
{"userName":"jeffr@commscope.com",
"password":"test"}
```

Action:

Use the **POST** action to send the request.

Response Body:

Once the request is sent, you will receive the response body that provides the token and its expiration, as shown in the following example:

```
{
    "userName": "jeffr@commscope.com",
    "token":
"eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBhcnJpcy5jb20iLCJyb2xlUGsiOiIyMTAwMCIsImFjY291bnRBZG1pblBrIj
oiMiIsImFwaUtleSI6IkhOV2dNSG93cERCCNCIsImlhdCI6MTU3NTQ4NjExMSwiZXhwIjoxNTc1NDg2NDExfQ.JwBWDgj_K7jP1kDvVU8aq_v
mNulsdx2FD_MUNLAuymQ",
    "expireDateTime": "2019-12-04T12:06:51-07:00[America/Denver]"
}
```

> **NOTE**
> Because each token expires after five minutes, you will need to re-send the request often. You need to use the active token with all other requests, as shown elsewhere in this document.

List Format of Response Body of GET Calls

All GET calls that are not for a specific ID will return a list of items.

The basic Json schema and an example of this schema are shown below.

Json Schema

```
{
  "type": "object",
  "properties": {
    "page": {
      "type": "object",
      "properties": {
```

```
        "pageSize": { "type": "integer" },
        "page": { "type": "integer" },
                "totalCount":{type:"long" }
      },
      "required": [ "pageSize",  "page"]
    },
    "contents": {
      "type": "array",
      "items": [
        {
          "type": "object"
        }
      ]
    }
  },
  "required": [
    "page",
    "contents"
  ]
}
```

Example of Json Response Body From a GET Call That Provides Lists

Each set of brackets ([ ]) indicates a list, and each entry with curly braces ({ }) is an item in that list.

```
{
  "page": {
    "pageSize": 5,
    "page": 3,
    "totalCount":441
  },
  "contents": [
    {
      "guid": "uniqueId",
      "property1": "propertyValue",
      "property2": "property2Value"
    },
    {
      "guid": "differentUniqueId",
      "property1": "propertyValueForItem2",
      "property2": "property2ValueForItem2"
    }
  ]
}
```

For an example of the response body that shows all the DPSKs within a specific DPSK pool, see Example of Using GET to Obtain Listings of Multiple DPSKs Within an Existing DPSK Pool on page 24

For an example of changing the Page properties (the first part of the response body shown above), see Example of Changing Page Properties in a GET Call on page 28

Sorting and Filter/Query

GET calls that are not by ID support paging and sorting/filtering as defined by each resource.

Tables that are provided in this document for the various resources (such as DPSKs) provide information about any query and sort operators that are available for each property. You can also combine a query with a sort, as well as with paging, in any request.

Sorting Syntax

To sort by a property, the query parameter is "sortBy". The property name is listed first, followed by a colon and the requested sort order: "desc" for descending or "asc" for ascending. For a multiple sort-order request, the order is implied by the order of the properties listed. Invalid properties are ignored and will not affect the sort order nor generate an error.

```
https://cloudpathServer/admin/apiv2/apiKey/resource?sortBy=prop1:desc,prop2:asc
```

For an example of a sort, including appending the URI with the sort parameter, see Example of Sorting a GET Call on page 31.

Query Operators

The following query operators are supported:

TABLE 4 Supported Query Operators

Operator	Meaning
like	Does a comparison equivalent to a database query of %val% on strings only.
eq	Does a direct comparison where the two values must be equal.
in	Allows a list of equals to be used, like the "in" operator in a database . This is often used with enumerations, so you can get a list with two or more values in a list of enumerations. Value for the in operator must be: 'val1_val2_val3' where _ is the separator between values.

Query Syntax

To query with a GET of all elements, the query parameter is "filter". This is a list of properties and the operator and value to evaluate, in the format: property(operator:value). Invalid properties will be ignored, and valid properties will be processed. They will be ordered as provided on the property list, with the first property added first to the filter query.

The following example syntax of filtering on a property called *prop1*, using the "like' operator with *value* denoting the value you provide, and also sorting on a property called *prop2*, using the "eq" parameter and a value for *val2*:

```
https://cloudpathServer/admin/apiv2/apiKey/resource?filter=prop1(like:value),prop2(eq:val2)
```

For an example of a query, including appending the URI with the filter parameter, see Example of Filtering a GET Call on page 30.

Return Codes and Error Response Format

The API returns an error response code if something has gone wrong with the request.

The following tables provide the basic components of an error message and the meaning of the various codes.

Basic Parts of an Error Message

TABLE 5 Error Message Components

Property Name	Type	Description
message	String	A message providing additional details about the error.
code	String	The HTTP return code. The response will return this code, but you can also obtain this code from the resource.
uri	String	The URI that made the request that generated the error.

Common Return Codes and Their Meaning

TABLE 6 Codes and Meaning

Code	Meaning
200	"OK" - The response for almost all successful GET requests, and most PUT requests.
201	"Created" - The response for a POST if the resource requested was created.
204	No Content - This will only occur after the deletion of a resource.
400	"Bad Request" - Occurs when the parameters or body required in the method were not correctly provided. This includes missing headers (such as ContentType=application/json) on a POST and PUT.
401	"Not Authorized" - Occurs If the token is invalid (not expired) or has not been provided. This can also occur if the user does not have an Admin role on the account that is associated with the API key. In addition, providing incorrect credentials on the POST **/token** will generate this error.
404-	"Not Found" - Occurs If the item requested or any resource identified in the URI does not exist in the Cloudpath account.
406 -	"Not Acceptable" - Occurs if the token has expired. This error is possible on any endpoint, except the POST on **/token**.
409-	"Conflict" - Occurs when the request itself is valid, but the data in the request causes a conflict with existing data. An common example is if the name is not unique.
415	"Unsupported Media Type" - the REST v2 APIs support only JSON.
4***, 500, 503	Several other errors might occur. The error response should provide additional details as to the issue.

DSPK Resources and URIs

DPSK Pools

You can use REST API v2 calls to obtain information about existing DPSK pools, to create or edit pools, and to delete pools.

> **NOTE**
> This resource does not currently support sorting, paging, or filtering.

The following table lists the properties that can be used in the body of a DPSK-pool-related request and/or are provided in the response body. (In the "Optional" column, if a default value is applicable, it is listed in angle brackets (**< >**).

TABLE 7 Resource: DPSK Pools

Property Name	Type	Editable	Optional in Request/ Response Body?	Description
guid	String	No	Optional on create only, but not on any subsequent uses	The unique identifier for this DPSK pool.
additionalAttributes	List <Attribute>	Yes	Yes	A list of additional attributes to apply to RADIUS configurations. In the **Configuration > DPSK Pools** portion of the UI, if you add or modify a DPSK pool, you can click **+** at the bottom of the Policies section to invoke a drop-down of all the additional attributes that are available. Also, see the "Resource: Additional Attributes" table below for more information. **NOTE** In the UI, you can click the **Variables** arrow to view a list of supported variables.
classid	String	Yes	Yes	The device class ID to use. If not specified, the device is not assigned a class ID.
description	String	Yes	No	A description for this pool.
deviceCountLimit	Int	Yes	No <1>	If you set the enforceDeviceCountLimit parameter to *true*, you can specify the number of devices that are allowed to utilize the DPSK pool you are creating.
displayName	String	Yes	No	The display name for this pool; must be unique.
enabled	Boolean	Yes	No<true>	Indicates if this DPSK pool is enabled. Switching from true to false disables the pool.
enforceDeviceCountLimit	Boolean	Yes	No<false>	If true, each DPSK is assigned a maximum device count as specified in the deviceCountLimit parameter. If false, an unlimited number of devices can utilize the DPSK.
enforceExpirationDate	Boolean	Yes	No <false>	If true, newly generated DPSKs are assigned an expiration date that is based on the creation date and the defined offset.

TABLE 7 Resource: DPSK Pools (continued)

Property Name	Type	Editable	Optional in Request/ Response Body?	Description
expirationDateType	String	Yes	No <END_OF_MONTH>	A Specific date in format YYYYMMDD, or a type of: MINUTES_AFTER_TIME, HOURS_AFTER_TIME, DAYS_AFTER_TIME, WEEKS_AFTER_TIME, MONTHS_AFTER_TIME, QUARTERS_AFTER_TIME , YEARS_AFTER_TIME , END_OF_HOUR , END_OF_DAY , END_OF_WEEK , END_OF_MONTH , END_OF_QUARTER , END_OF_HALF , END_OF_YEAR
expirationDateValue	Int	Yes	No <1>	Will be used if HOURS_AFTER_TIME is provided in the expirationDataType
filterid	String	Yes	Yes	Filter ID for the device in the pool. If not specified, the device is not assigned a filter ID.
phraseDefaultLength	int	Yes	No<12>	The pass phrase default length.
phraseRandomCharactersType	Enum <String>	Yes	No	An enum that indicates the characters to be included in the passphrase. ALPHANUMERIC_UPPER, ALPHANUMERIC_LOWER, ALPHANUMERIC_MIXED, ALPHABETIC_UPPER, ALPHABETIC_LOWER, ALPHABETIC_MIXED, NUMERIC, ASCII
sessionTimeout	Long	Yes	No	The reauthentication period, in seconds, included in the RADIUS reply for successful authentications. If the device stays connected for longer than this period, the WLAN or switch requires that the device be reauthenticated (invisibly to the user). If not specified, sessionTimeout is not applied, which means that the device is not required to reauthenticate as long as it remains associated.
ssidList	List <String>	Yes	No	A list of SSIDs as strings for this DPSK pool.
vlanid	String	Yes	Yes	The VLAN to use. If not specified, the device receives the default VLAN that is configured on the controller. This property value can be one of the DPSK-supported environmental variables.

Resource: Additional Attributes:

The following table lists the additional-attribute properties to apply to RADIUS configurations that can be used in the body of a DPSK-pool-related request and/or are provided in the response body.

> **NOTE**
> In the **Configuration > DPSK Pools** portion of the UI, if you add or modify a DPSK pool, you can click **+** at the bottom of the Policies section to invoke a drop-down of all the additional attributes that are available.

TABLE 8 Resource: Additional Attributes

Property Name	Type	Editable	Filter or Sort	Optional?	Description
name	String	Yes	No	No	The RADIUS attribute to use.
action	String <Enum>	Yes	No	No	The action to take must be one of the following: ADD, ADD_OR_REPLACE, or ADD_MULTIPLE.
value	String	Yes	No	No	The value to substitute.

DPSK Pool URIs and Actions

The following table indicates the URIs that you can append to the root URI, along with the actions you can take, and possible return codes you can receive.

TABLE 9 DPSK Pool URIs and Actions

URI	Action	Description	Return Codes
/dpskPools	GET	Returns the list of all DPSK pools that are in the system.	401 - Unauthorized 200 - OK
	POST	Creates a DPSK pool.	401 - Unauthorized 201 - Created
/dpskPools/{guid}	GET	Returns the specific DPSK pool.	404 - Not Found 401 - Unauthorized 200 - OK
	DELETE	Deletes the pool specified and all its child resources, including DPSKs, devices, and associated enrollments.	500 - Internal Server Error (such as " Pool in use") 401 - Unauthorized 204 - No content 200 - OK
	PUT	Updates aa DPSK pool. See the "Resource: DPSK Pools" and "Additional Attributes" tables above for what properties can be edited.	404 - Not Found 401 - Unauthorized 200 - OK

Example of Using GET for DPSK Pools

This example demonstrates how to obtain information about a DPSK pool that has already been created in the Cloudpath UI. (Instructions on creating DPSK pools in the UI are located in the *Cloudpath Enrollment System Ruckus External Dynamic Pre-Shared Key (eDPSK) Configuration Guide*.) The following figure shows information about a DPSK pool called DPSK Pool 1 that has been created in the UI.

FIGURE 2 DPSK Pool 1 Created in the UI
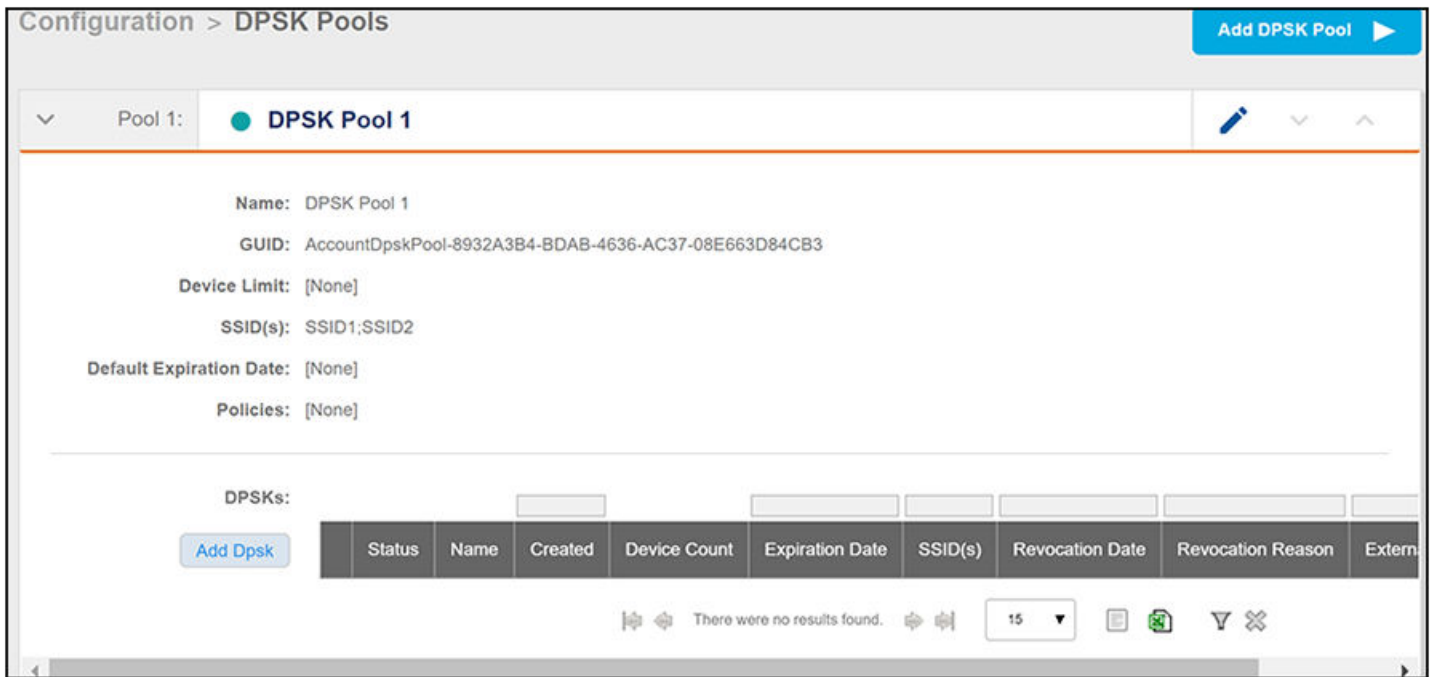
URI:

This example uses the following URI:

```
https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)

- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 10 Request Headers

Key	Value
Content-type	application/json
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2NvdW50QWRtaW5QayI6IjIiLCJhcGlLZXkiOiJaV2NTTGpQZZUdZMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCI6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

None required.

Action:

Use the **GET** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 200 OK" if the GET is successful. The response body provides information about all existing DPSK pools in your Cloudpath system. In this example, DPSK Pool 1 is the only existing pool. You can see that the information in the response body matches the configuration of DPSK Pool 1 as specified in the UI (for example, the "guid" values are the same in each place, the SSID list is the same, and so on). Any values that were not specified in the UI are the default values:

```
{
    "page": {
        "page": 1,
        "pageSize": 1,
        "totalCount": 1
    },
    "contents": [
        {
            "guid": "AccountDpskPool-8932A3B4-BDAB-4636-AC37-08E663D84CB3",
            "displayName": "DPSK Pool 1",
            "description": "",
            "enabled": true,
            "phraseDefaultLength": 12,
            "phraseRandomCharactersType": "ALPHABETIC_LOWER",
            "ssidList": [
                "SSID1",
                "SSID2"
            ],
            "enforceExpirationDate": false,
            "expirationDateType": "MONTHS_AFTER_TIME",
            "expirationDateValue": "1",
            "enforceDeviceCountLimit": false,
            "deviceCountLimit": 1,
            "vlanid": "",
            "filterid": "",
            "classid": "",
            "additionalAttributes": []
        }
    ]
}
```

Example of Using POST to Create a DPSK Pool

This example demonstrates how to create a new DPSK pool in your Cloudpath system by using the REST API POST call.

URI:

This example uses the following URI:

```
https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)
- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 11 Request Headers

Key	Value
Content-type	application/json
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2NvdW50QWRtaW5QayI6IjIiLCJhcGlLZXkiOiJaV2NjTGpQQZUdZMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCI6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

The following is an example of a request body for creating a DPSK pool. You can refer to the preceding Resource tables for a list and descriptions of properties you can use to define your pool. For instance, in this example, a pool called "DPSK Pool 20" is being created with the values provided in the body.

```
{
    "displayName": "DPSK Pool 20",
    "description": "",
    "enabled": true,
    "phraseDefaultLength": 11,
    "phraseRandomCharactersType": "ALPHABETIC_LOWER",
    "ssidList": [
        "SSID19",
        "SSID20"
    ],
    "enforceExpirationDate": false,
    "expirationDateType": "MONTHS_AFTER_TIME",
    "expirationDateValue": "1",
    "enforceDeviceCountLimit": true,
    "deviceCountLimit": 4,
    "vlanid": "777",
    "filterid": "",
    "classid": "",
    "additionalAttributes": [
        {
            "name": "User-Name",
            "action": "ADD_OR_REPLACE",
            "value": "${DPSK.THRIDPARTY_ID}"
        },
        {
            "name": "Callback-Number",
            "action": "ADD",
            "value": "1234"
        }
    ]
}
```

Action:

Use the **POST** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 201 Created" if the POST is successful. You can see that a "guid" (a unique identifier for the newly created pool) is included in the response body.

```
{
    "guid": "AccountDpskPool-37AF7D9B-7A3A-4953-B314-10A4758803DE",
    "displayName": "DPSK Pool 20",
    "description": "",
    "enabled": true,
    "phraseDefaultLength": 11,
    "phraseRandomCharactersType": "ALPHABETIC_LOWER",
    "ssidList": [
        "SSID19",
        "SSID20"
    ],
    "enforceExpirationDate": false,
    "expirationDateType": "MONTHS_AFTER_TIME",
    "expirationDateValue": "1",
    "enforceDeviceCountLimit": true,
    "deviceCountLimit": 4,
    "vlanid": "777",
    "filterid": "",
    "classid": "",
    "additionalAttributes": [
        {
            "name": "User-Name",
            "action": "ADD_OR_REPLACE",
```

```
            "value": "${DPSK_THIRDPARTY_ID}"
        },
        {
            "name": "Callback-Number",
            "action": "ADD",
            "value": "1234"
        }
    ]
}
```

Verifying the Newly Created DPSK Pool in the Cloudpath UI:

You can go to the **Configuration > DPSK Pools** area of the UI, and expand the pool to confirm that the pool was created correctly. The values should match those in the Response body from the POST. For example, in this expanded view, you can see that the GUID matches the one provided in the Response body.

FIGURE 3 Expanded View in UI of Pool Created With an API Post Call
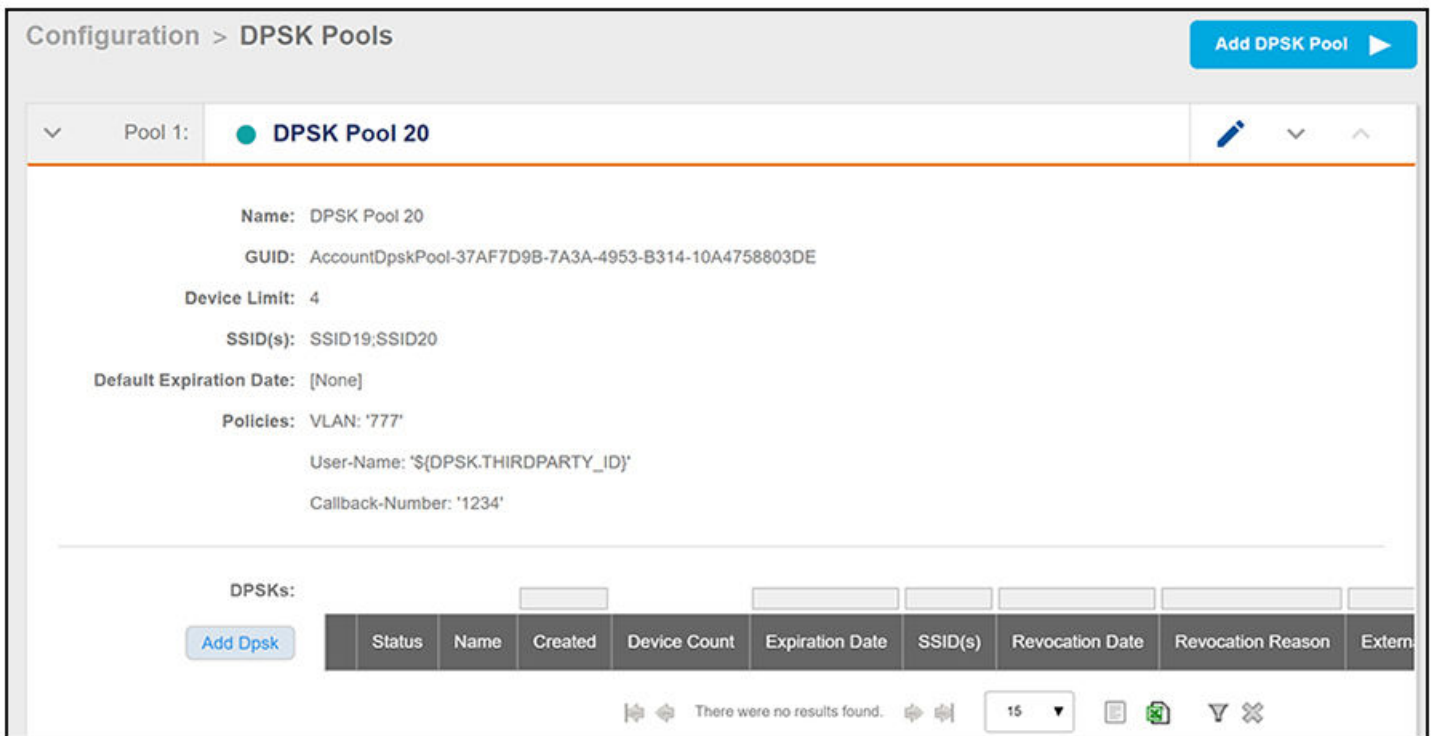
If you go to the Edit view of the pool, you see additional pool values. Again, they should match the values you assigned to the properties you used in your POST.

FIGURE 4 Edit View of Pool Created With an API Post Call

DPSKs

You can use REST API v2 calls to obtain information about existing DPSKs, to create or edit DPSKs, and to delete DPSKs.

The following table lists the properties that can be used in the body of a DPSK-related request and/or are provided in the response body. (In the "Optional" column, if a default value is applicable, it is listed in angle brackets (**< >**).

TABLE 12 Resource: DPSKs

Property Name	Type	Editable	Sort/Filter	Optional in Request/Response Body?	Description
guid	String	No	Neither	Optional on create only, but not on any subsequent uses	The unique identifier for this PSK.
name	String	Yes	sortBy, like	No	The reference name of the item.
thirdPartyId	String	Yes	sortBy , like	No	A string that can be used by third-party consumers. This can be referenced using attribute variables in RADIUS attributes as: DPSK_THIRD_PARTY_ID
passphrase	String	Yes	Neither	Optional on create only, but not on any subsequent uses	The pre-shared key for this PSK, which will be automatically assigned if not provided in a POST. This string will always be returned with the resource.
status	Enum <String>	Yes	in, eq	No	The status of the PSK. Options are: ACTIVE, EXPIRED, or REVOKED. A change to REVOKED or ACTIVE can be requested but only for a DPSK whose state is not EXPIRED.
ssidList	List <String>	Yes	Neither	No	If not null, this is the list of SSIDs on which the DPSK can be used. This value must be a subset of the SSIDs assigned to the corresponding DPSK pool. If the ssidList property is not specified, the DPSK can be used on any SSID contained in the "ssids" field of the corresponding DPSK pool.
expirationDateTime	String < DateTime>	Yes	sortBy	Yes	The DPSK (and the devices registered to the DPSK) will no longer be usable after the date and time specified. ISO 8601 formatted date is used for this property.
useExpirationDate	boolean	Yes	Neither	Yes <true>	Indicates if expiration date time should be used.
useDeviceCountLimit	boolean	Yes	Neither	Yes <false>	Indicates that the system will enforce the device count limit for this specific DPSK.
deviceCountLimit	Long	Yes	Neither	Yes	If specified, this DPSK will have its own device count limit enforced independently of the corresponding DPSK pool.
revokedDateTime	String < DateTime>	No	Neither	Yes	The date time (in ISO 8601 date time format) that the DPSK was revoked.
revocationReason	String	Yes	Neither	Yes	If the status is changed to REVOKED, this is the "reason for the revocation."
vlanId	String	Yes	Neither	No	If specified, this VLAN ID will override the VLAN ID specified within the corresponding DPSK pool.
overrideReauthentication	String	Yes	Neither	Yes	If specified, this Reauthentication value will override the value specified within the DPSK pool.

DPSK URIs and Actions

The following table indicates the URIs that you can append to the root URI, along with the actions you can take, and possible return codes you can receive.

TABLE 13 DPSK URIs and Actions

URI	Action	Description	Return Codes
dpskPools /{poolGuid}/dpsks	GET	Returns the list of PSKs that are part of the identified pool.	200 - OK 404 -Not Found
	POST	Creates a PSK in the identified pool.	201- Created 404 - Not Found
/dpskPools/{poolGuid}/dpsks/{guid}	GET	Returns the specified PSK.	200 - OK 404 - Not Found
	DELETE	Deletes the specified PSK.	200 - OK 204 - No content
	PUT	Updates a PSK. See the "Resource: DPSKs" table above for what properties can be edited.	200 - OK 404 -Not Found

Example of Using PUT to Edit a DPSK Within an Existing DPSK Pool

This example demonstrates how to edit a DPSK that has already been created either by using the UI or by using a REST API POST call. In Example of Using POST to Create a DPSK Pool on page 17, an example shows the creation of a pool called DPSK Pool 20. Suppose that a DPSK called DPSK15 was then created within that pool. The following figure shows the properties of DPSK15 from the UI:

FIGURE 5 DPSK15 Information in UI Before API PUT Call
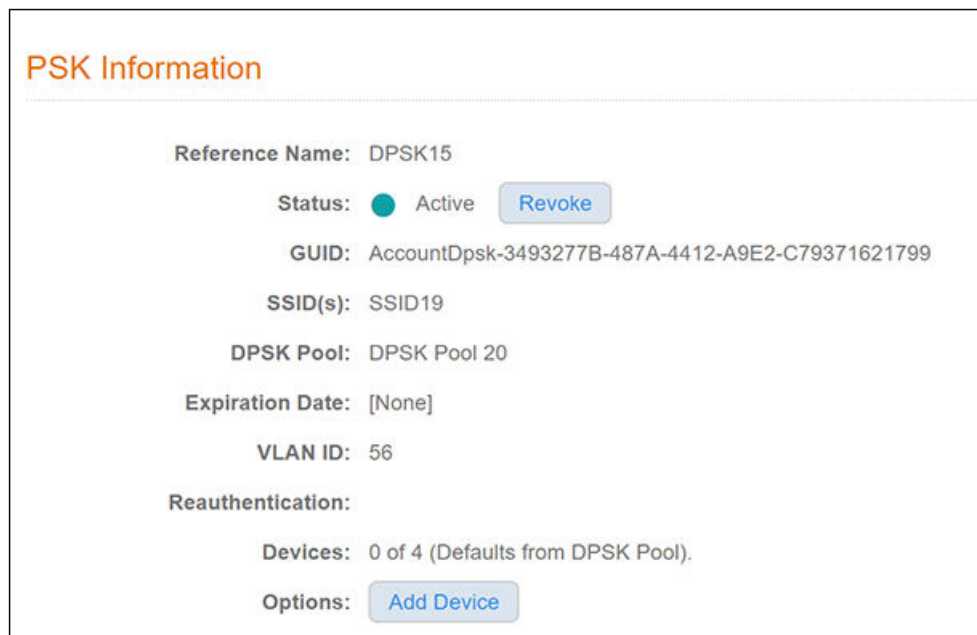
URI:

This example uses the following URI:

```
https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools/AccountDpskPool-37AF7D9B-7A3A-4953-
B314-10A4758803DE/dpsks/AccountDpsk-3493277B-487A-4412-A9E2-C79371621799
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)

- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

- AccountDpskPool-37AF7D9B-7A3A-4953-B314-10A4758803DE is the GUID of the DPSK pool to which this DPSK belongs

- AccountDpsk-3493277B-487A-4412-A9E2-C79371621799 is the GUID of this specific DPSK that will be edited (see the figure above for this GUID)

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 14 Request Headers

Key	Value
Content-type	application/json
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2vdW50QWRraW5Qayl6IjIiLCJhcGlLZXkiOiJaV2NjTGpQZUddZMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCl6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

The following is an example of a request body for editing the DPSK called DPSK.

> **NOTE**
> To create the body of a PUT request, you can start by first doing a GET request on the DPSK that you want to edit. After a successful GET request, copy and paste the entire response body into the request body of your GET. Then, edit the values you want to change. In the example shown below, the ssidList has been changed to null, and the vlanID has been changed to 44.

.

```
{
    "guid": "AccountDpsk-3493277B-487A-4412-A9E2-C79371621799",
    "name": "DPSK15",
    "passphrase": "nnyvznlmjzz",
    "status": "ACTIVE",
    "ssidList": [],
    "expirationDateTime": "2019-12-06T00:00-07:00[America/Denver]",
    "useDeviceCountLimit": false,
    "deviceCountLimit": 0,
    "vlanid": "44"
}
```

Action:

Use the **PUT** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 200 OK" if the PUT is successful. You can see that the response body below matches what was requested in the request body above.

```
{
    "guid": "AccountDpsk-3493277B-487A-4412-A9E2-C79371621799",
    "name": "DPSK15",
    "passphrase": "nnyvznlmjzz",
    "status": "ACTIVE",
    "ssidList": [],
    "expirationDateTime": "2019-12-06T00:00-07:00[America/Denver]",
    "useDeviceCountLimit": false,
    "deviceCountLimit": 0,
```

```
        "vlanid": "44"
    }
```

Verifying the Edited DPSK in the Cloudpath UI:

You can go to the **Configuration > DPSK Pools** area of the UI, and click the view icon of the desired DPSK in the "DPSKs" section of the window to confirm that the DPSK was edited correctly. The values should match those in the Response body from the PUT. For example, you can see that the SSID list no longer has any restrictions, and that the VLAN ID has been changed to 44.

FIGURE 6 DPSK15 Information in UI After API PUT Call
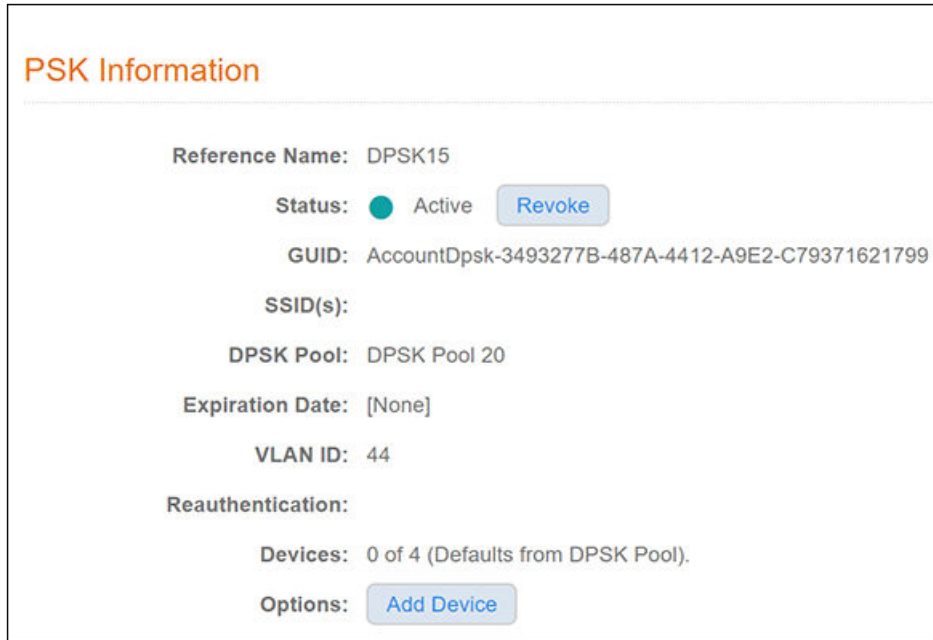
Example of Using GET to Obtain Listings of Multiple DPSKs Within an Existing DPSK Pool

This example demonstrates how to use a GET call to obtain a full list of DPSKs. In this example, six DPSKs have been created in the pool called DPSK Pool 20, as shown in the following figure:

FIGURE 7 Listing of Multiple DPSKs Within a Pool

URI:

This example uses the following URI:

```
https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools/AccountDpskPool-37AF7D9B-7A3A-4953-
B314-10A4758803DE/dpsks/
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)

- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

- AccountDpskPool-37AF7D9B-7A3A-4953-B314-10A4758803DE is the GUID of the DPSK pool from which you want a listing of all its DPSKs and their respective properties.

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 15 Request Headers

Key	Value
Content-type	application/json
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2NvdW50QWRtaW5QayI6IjIiLCJhcGlLZXkiOiJaV2NjTGpQZUdUdMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCI6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

No request body is needed for this request. All the necessary information is in the URI.

Action:
Use the **GET** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 200 OK" if the GET is successful. The response body below lists information for each of the six DPSKs within the corresponding pool. The first part of the output shows the page information: All the information is on page 1, the page size is currently 6 (meaning there are six items on page 1), and the total number of DPSKs listed is 6.

```
{
    "page": {
        "page": 1,
        "pageSize": 6,
        "totalCount": 6
    },
    "contents": [
        {
            "guid": "AccountDpsk-C9302B8E-191A-4D48-AA88-ED0BE48E961B",
            "name": "DPSK33",
            "passphrase": "zhcpeclpjzb",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-7D1DE55E-D00D-4A01-95D0-D53835D7985B",
            "name": "DPSK32",
            "passphrase": "yhxicugernd",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-DA0B4E45-13A3-4A8E-B822-CD9FDCE6583A",
            "name": "DPSK31",
            "passphrase": "wahzqqdhqvd",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-8843A2A6-1052-431F-ADBD-FE05F5F42C7F",
            "name": "DPSK30",
            "passphrase": "ewlpxpkcizg",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-ACEDBBAC-B68C-4BF6-8ADF-51C68EB19874",
            "name": "DPSK23",
            "passphrase": "lytwupslyjq",
            "status": "ACTIVE",
            "ssidList": [],
```

```
                "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
                "useDeviceCountLimit": false,
                "deviceCountLimit": 0,
                "vlanid": ""
            },
            {
                "guid": "AccountDpsk-3493277B-487A-4412-A9E2-C79371621799",
                "name": "DPSK15",
                "passphrase": "nnyvznlmjzz",
                "status": "ACTIVE",
                "ssidList": [],
                "expirationDateTime": "2019-12-06T00:00-07:00[America/Denver]",
                "useDeviceCountLimit": false,
                "deviceCountLimit": 0,
                "vlanid": "44"
            }
        ]
    }
```

Example of Creating a DPSK Into an Existing DPSK Pool With a POST call

This example shows how to use a POST to create a new DPSK in an existing pool.

URI:

This example uses the following URI:

```
    https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools/AccountDpskPool-37AF7D9B-7A3A-4953-
    B314-10A4758803DE/dpsks
```

where:

- • https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)

- • BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

- • AccountDpskPool-37AF7D9B-7A3A-4953-B314-10A4758803DE is the GUID of the pool to which you are adding the new DPSK

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 16 Request Headers

Key	Value
Content-type	application/json
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2NvdW50QWRtaW5QayI6IjIiLCJhcGlLZXkiOiJaV2NjTGppQZUdZMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCI6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

The following is an example of a request body for creating a DPSK. You can refer to the preceding Resource tables for a list and descriptions of properties you can use to define your pool. For instance, in this example, a pool called "dpsk51" is being created with the values provided in the body.

```
    {
        "name":"dpsk51",
        "passphrase":"qsrmxldiecca",
        "status":"ACTIVE",
        "ssidList":[],
        "expirationDateTime":"2019-12-21T00:00-07:00[America/Denver]",
        "useDeviceCountLimit":true,
        "deviceCountLimit":7,
```

```
        "vlanid":"2",
        "overrideReauthentication":86400
    }
```

Action:

Use the **POST** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 201 Created" if the POST is successful. You can see that a "guid" (a unique identifier for the newly created DPSK) is included in the response body.

```
{
    "guid": "AccountDpsk-82DB703A-E43A-4972-A04F-8634BEB25A1D",
    "name": "dpsk51",
    "passphrase": "qsrmxldiecca",
    "status": "ACTIVE",
    "ssidList": [],
    "expirationDateTime": "2019-12-21T00:00-07:00[America/Denver]",
    "useDeviceCountLimit": true,
    "deviceCountLimit": 7,
    "vlanid": "2",
    "overrideReauthentication": 86400
}
```

You can go to the **Configuration > DPSK Pools** area of the UI, and expand the pool to confirm that the DPSK was created correctly.

Example of Changing Page Properties in a GET Call

This example demonstrates how to change the page properties to display the response body of a GET call (where you are getting a complete list of items) as you choose. In the previous GET example, six DPSKs were listed on one page. The page information was displayed as the following:

```
"page": {
        "page": 1,
        "pageSize": 6,
        "totalCount": 6
```

URI:

This example uses the following URI:

```
 https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools/AccountDpskPool-37AF7D9B-7A3A-4953-
B314-10A4758803DE/dpsks/?pageSize=3&page=2
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)
- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI
- AccountDpskPool-37AF7D9B-7A3A-4953-B314-10A4758803DE is the GUID of the DPSK pool from which you got a listing of all its DPSKs and their respective properties.
- ?pageSize=3&page=2 denotes that you want three items per page in the GET response body, and you want only the page-2 information.

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 17 Request Headers

Key	Value
Content-type	application/json

TABLE 17 Request Headers (continued)

Key	Value
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2NvdW50QWRtaW5Qayl6IjIiLCJhcGlLZXkiOiJaV2NjTGppQZUdZMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCl6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

No request body is needed

Action:

Use the **GET** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 200 OK" if the GET is successful. The response body below lists the information based on how you used the page filters. As shown below, the page information reflects how it was specified in the URI, and the three Page 2 DPSKs are listed:

```
{
    "page": {
        "page": 2,
        "pageSize": 3,
        "totalCount": 6
    },
    "contents": [
        {
            "guid": "AccountDpsk-8843A2A6-1052-431F-ADBD-FE05F5F42C7F",
            "name": "DPSK30",
            "passphrase": "ewlpxpkcizg",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-ACEDBBAC-B68C-4BF6-8ADF-51C68EB19874",
            "name": "DPSK23",
            "passphrase": "lytwupslyjq",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-3493277B-487A-4412-A9E2-C79371621799",
            "name": "DPSK15",
            "passphrase": "nnyvznlmjzz",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-06T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": "44"
        }
    ]
}
```

Example of Filtering a GET Call

This example demonstrates how to filter a GET call by using a filter parameter. For a list of properties that support filters, and the filter operators that can be used on these properties, refer to Table 12. Suppose you want to use "like" with the value of "3" to filter on only the DPSKs that have a "3" as part of their *name* property. You would add the following string to the URI: ?filter=name(like:3)

URI:

This example uses the following URI:

```
https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools/AccountDpskPool-37AF7D9B-7A3A-4953-
B314-10A4758803DE/dpsks/?filter=name(like:3)
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)

- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

- AccountDpskPool-37AF7D9B-7A3A-4953-B314-10A4758803DE is the GUID of the DPSK pool from which you want a listing of the DPSKs that match the filter criteria that you append to the URI.

- ?filter=name(like:3) is the filter criteria that denotes that you want to display only those DPSKs that contain the numeral "3" in their name.

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 18 Request Headers

Key	Value
Content-type	application/json
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2NvdW50QWRtaW5QayI6IjIiLCJhcGlLZXkiOiJaV2NjTGppQZUdZMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCI6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

No request body is needed

Action:

Use the **GET** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 200 OK" if the GET is successful. The response body below lists the information based on how you used the filters. As shown below, all the DPSKs listed contain the numeral "3" in their name:

```
{
    "page": {
        "page": 1,
        "pageSize": 5,
        "totalCount": 5
    },
    "contents": [
        {
            "guid": "AccountDpsk-C9302B8E-191A-4D48-AA88-ED0BE48E961B",
            "name": "DPSK33",
            "passphrase": "zhcpeclpjzb",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
```

```
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-7D1DE55E-D00D-4A01-95D0-D53835D7985B",
            "name": "DPSK32",
            "passphrase": "yhxicugernd",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-DA0B4E45-13A3-4A8E-B822-CD9FDCE6583A",
            "name": "DPSK31",
            "passphrase": "wahzqqdhqvd",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-8843A2A6-1052-431F-ADBD-FE05F5F42C7F",
            "name": "DPSK30",
            "passphrase": "ewlpxpkcizg",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-ACEDBBAC-B68C-4BF6-8ADF-51C68EB19874",
            "name": "DPSK23",
            "passphrase": "lytwupslyjq",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        }
    ]
}
```

Example of Sorting a GET Call

This example demonstrates how to sort a GET call by using the *sortBy* parameter. For a list of properties that support sorting, refer to Table 12. Suppose you want to sort on the name of the DPSK, in descending order. You would add the following string to the URI: ?sortBy=name:desc

URI:

This example uses the following URI:

```
https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools/AccountDpskPool-37AF7D9B-7A3A-4953-
B314-10A4758803DE/dpsks/?sortBy=name:desc
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)

- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

- AccountDpskPool-37AF7D9B-7A3A-4953-B314-10A4758803DE is the GUID of the DPSK pool from which you want to sort the DPSKs by adding sorting criteria to the URI.

- ?sortBy=name:desc is the sorting criteria that denotes that you want to display DPSKs by name, in descending order.

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 19 Request Headers

Key	Value
Content-type	application/json
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2NvdW50QWRtaW5QayI6Ij IiLCJhcGlLZXkiOiJaV2NjTGpQZZUdZMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCI6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

No request body is needed

Action:

Use the **GET** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 200 OK" if the GET is successful. The response body below lists the information based on how you used the *sortBy* parameter. As shown below, the DPSKs have been sorting by name in descending order.

```
{
    "page": {
        "page": 1,
        "pageSize": 6,
        "totalCount": 6
    },
    "contents": [
        {
            "guid": "AccountDpsk-C9302B8E-191A-4D48-AA88-ED0BE48E961B",
            "name": "DPSK33",
            "passphrase": "zhcpeclpjzb",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-7D1DE55E-D00D-4A01-95D0-D53835D7985B",
            "name": "DPSK32",
            "passphrase": "yhxicugernd",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
            "useDeviceCountLimit": false,
            "deviceCountLimit": 0,
            "vlanid": ""
        },
        {
            "guid": "AccountDpsk-DA0B4E45-13A3-4A8E-B822-CD9FDCE6583A",
            "name": "DPSK31",
            "passphrase": "wahzqqdhqvd",
            "status": "ACTIVE",
            "ssidList": [],
            "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
```

20

```
                    "useDeviceCountLimit": false,
                    "deviceCountLimit": 0,
                    "vlanid": ""
                },
                {
                    "guid": "AccountDpsk-8843A2A6-1052-431F-ADBD-FE05F5F42C7F",
                    "name": "DPSK30",
                    "passphrase": "ewlpxpkcizg",
                    "status": "ACTIVE",
                    "ssidList": [],
                    "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
                    "useDeviceCountLimit": false,
                    "deviceCountLimit": 0,
                    "vlanid": ""
                },
                {
                    "guid": "AccountDpsk-ACEDBBAC-B68C-4BF6-8ADF-51C68EB19874",
                    "name": "DPSK23",
                    "passphrase": "lytwupslyjq",
                    "status": "ACTIVE",
                    "ssidList": [],
                    "expirationDateTime": "2019-12-07T00:00-07:00[America/Denver]",
                    "useDeviceCountLimit": false,
                    "deviceCountLimit": 0,
                    "vlanid": ""
                },
                {
                    "guid": "AccountDpsk-3493277B-487A-4412-A9E2-C79371621799",
                    "name": "DPSK15",
                    "passphrase": "nnyvznlmjzz",
                    "status": "ACTIVE",
                    "ssidList": [],
                    "expirationDateTime": "2019-12-06T00:00-07:00[America/Denver]",
                    "useDeviceCountLimit": false,
                    "deviceCountLimit": 0,
                    "vlanid": "44"
                }
            ]
        }
```

DPSK Devices

You can use REST API v2 calls to obtain information about all devices that are connected to a specific DPSK.

The following table lists the properties that can be used in the body of a DPSK device-related request and/or are provided in the response body.

TABLE 20 Resource: DPSK Devices

Property Name	Type	Editable	Filter or Sort	Optional in Request/ Response Body?	Description
guid	String	No	Neither	Yes (No device guid needs to be given because the GET call on devices is only for *all* devices associated with a specific DPSK.)	The unique identifier for this device
displayName	String	No	sortBy, like	Yes	The display name for this device
macAddress	String	No	sortBy, like	No	The MAC address for this device
pmk	String	No	Neither	No	The pmk assigned to this device.
ssid	String	No	sortBy, like	No	The SSID this device is connected to

TABLE 20 Resource: DPSK Devices (continued)

Property Name	Type	Editable	Filter or Sort	Optional in Request/ Response Body?	Description
createdDate	String <date>	No	sortBy	No	The date that this device was created

DPSK Devices URIs and Actions

The following table indicates the URIs that you can append to the root URI, along with the action you can take, and possible return codes you can receive.

TABLE 21 DPSK Devices URIs and Actions

URI	Action	Description	Return Codes
/dpskPools/{poolId}/dpsks / {dpskGuid}/devices	GET	Returns the list of devices that are on the specified DPSK.	200 - OK 401 - Unauthorized 404 -Not Found
	POST	Creates a device in the identified dpsks.	200 - OK 401 - Unauthorized 404 -Not Found
/dpskPools/{poolId}/dpsks/{dpskId}/ devices/{Guid}	GET	Returns the specified device.	200 - OK 401 - Unauthorized 404 -Not Found
	PUT	Updates the specified device.	200 - OK 401 - Unauthorized 404 -Not Found
	DELETE	Deletes the specified device.	200 - OK 204 - No content 401 - Unauthorized 404 -Not Found

Example of Using GET to Obtain Information About Devices Connected to a Specific DPSK

This example demonstrates how to use a GET call to obtain a list of devices connected to a DPSK. In this example, two devices exist on DPSK15, as shown at the bottom of the following figure:

FIGURE 8 DPSK Device Information

URI:

This example uses the following URI:

```
https://Jeff243.cloudpath.net/admin/apiv2/BmGsez6evQc8/dpskPools/AccountDpskPool-37AF7D9B-7A3A-4953-
B314-10A4758803DE/dpsks/AccountDpsk-3493277B-487A-4412-A9E2-C79371621799
/devices
```

where:

- https://Jeff243.cloudpath.net/admin/apiv2 is the root URI of the Cloudpath system being used (must include https, as shown)

- BmGsez6evQc8 is an example of an API key generated from the **Configuration > API Keys** section of the Cloudpath UI

- AccountDpskPool-37AF7D9B-7A3A-4953-B314-10A4758803DE is the GUID of the DPSK pool that the DPSK belongs to.

- AccountDpsk-3493277B-487A-4412-A9E2-C79371621799 is the GUID of DPSK15, which is the DPSK whose devices will be displayed upon a successful GET request.

Request Headers:

The following table shows the required request headers for this URI example, with an example value of an active token that must be pasted into the Authorization header value:

TABLE 22 Request Headers

Key	Value
Content-type	application/json
Authorization	eyJhbGciOiJIUzI1NiJ9.eyJzdWIiOiJqZWZmLnJob2RpZUBjb21tc2NvcGUuY29tIiwicm9sZVBrIjoiMjEwMDAiLCJhY2vdW50QWRtaW5QayI6IjIiLCJhcGlLZXkiOiJaV2NZTGpppQZUdZMGUiLCJpYXQiOjE1NzU1NzUxNTcsImV4cCI6MTU3NTU3NTQ1N30.2q4KTyDUevkAAz7mJyp529Be1-ip3EC8P06Xarmz5ao

Request Body:

No request body is needed for this request. All the necessary information is in the URI.

Action:

Use the **GET** action to send the request.

Response Body:

Once the request is sent, you will receive a "Status: 200 OK" if the GET is successful. The response body below lists information for the two devices connected to the DPSK. You can check the output against what is shown in the UI (see figure above).

```
{
    "page": {
        "page": 1,
        "pageSize": 2,
        "totalCount": 2
    },
    "contents": [
        {
            "guid": "AccountDpskDevice-A6427B1D-715A-4855-B0C2-A555764F7DFD",
            "displayName": "Device 2",
            "macAddress": "112233445567",
            "pmk": "",
            "ssid": "Jeff eDPSK2",
            "createdDate": "2019-12-06T23:58:03-07:00[America/Denver]"
        },
        {
            "guid": "AccountDpskDevice-497B674B-4957-4431-BBE6-62B4145DD6F1",
            "displayName": "Device 1",
            "macAddress": "112233445566",
            "pmk": "",
            "ssid": "Jeff eDPSK",
            "createdDate": "2019-12-06T23:57:21-07:00[America/Denver]"
        }
    ]
}
```