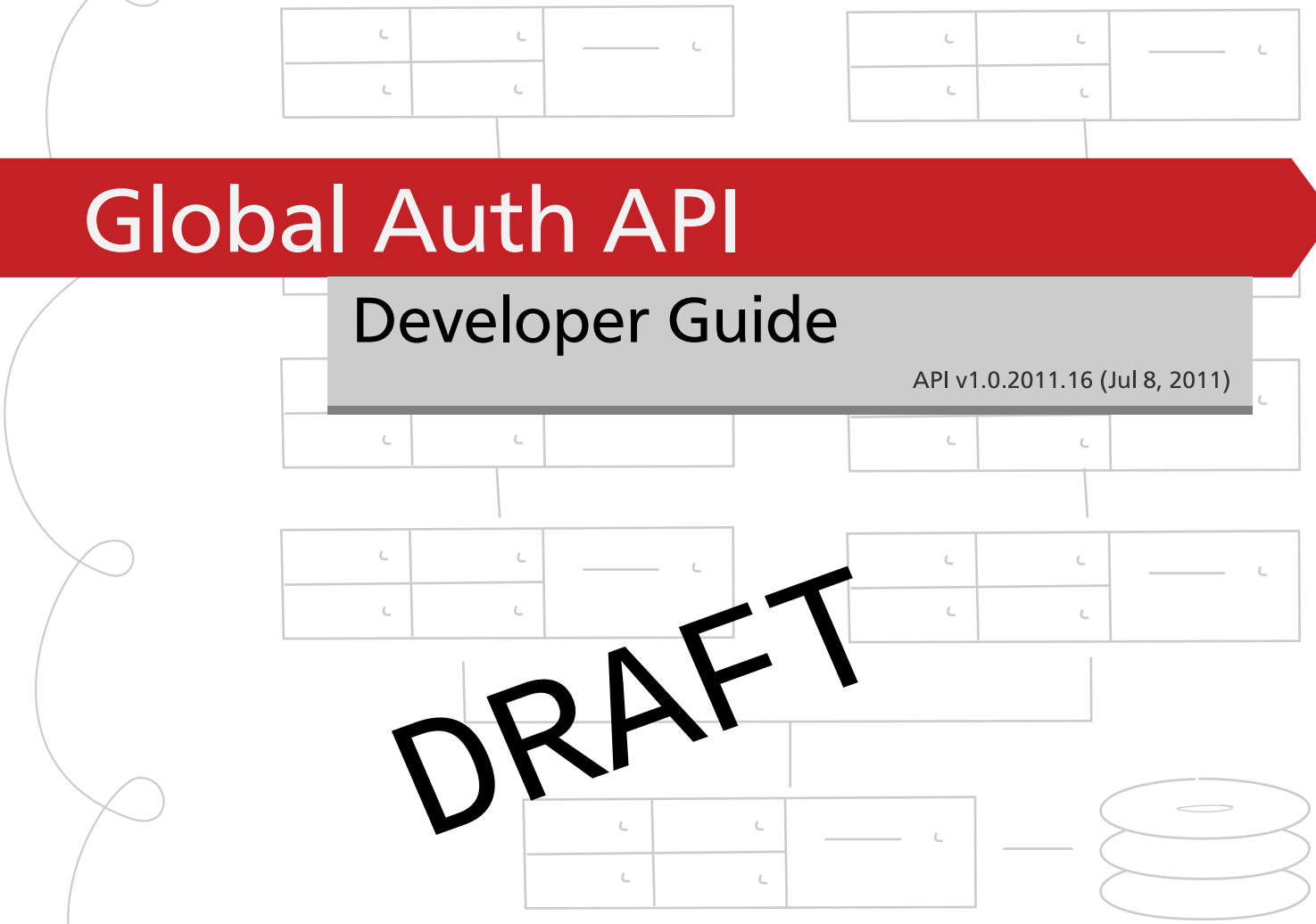
Global Auth API

Developer Guide

API v1.0.2011.16 (Jul 8, 2011)



DRAFT

rackspace® HOSTING

docs.rackspace.com/api

Global Auth API Developer Guide

API v1.0.2011.16 (2011-07-08)

This document is intended for software developers interested in developing applications which utilizes Global Auth API System as the authentication engine. It includes details on how to integrate with Global Auth API.

Table of Contents

List of Tables

List of Examples

1. Overview

The Global Auth API Service allows Rackspace Applications to obtain tokens that can be used to access resources in Rackspace. This document is intended for:

Service Developers	Service developers are interested in writing client for Global Auth API service.

This Guide assumes the reader is familiar with RESTful web services, HTTP/1.1, and JSON and/or XML serialization formats.

2. Concepts

The IdM system has several key concepts that are important to understand:

2.1. Token

Small bits of text that contain the security information for a login session and identifies the user, the user's groups, and the user's privileges. Tokens may be revoked at anytime and are valid for a finite duration.

2.2. Customer

Organizations that are customers of Rackspace

2.3. User

Individuals at a customer that have a login

2.4. Client

Rackspace applications (e.g.: Control Panel, Billing, etc.) or Customer third party applications. IdM will intially setup all Rackspace applications

3. General API Information

The Global Auth API is implemented using a RESTful web service interface. All requests to authenticate and operate against the Global Auth API are performed using SSL over HTTP (HTTPS) on TCP port 443.

3.1. Request/Response Types

The Global Auth API supports both the JSON and XML data serialization formats. The request format is specified using the `Content-Type` header and is required for operations that have a request body. The response format can be specified in requests using either the `Accept` header or adding an `.xml` or `.json` extension to the request URI. Note that it is possible for a response to be serialized using a format different from the request (see example below). If no response format is specified, JSON is the default. If conflicting formats are specified using both an `Accept` header and a query extension, the query extension takes precedence.

Table 3.1. Response Types

Format	Accept Header	Query Extension	Default
JSON	application/json	.json	Yes
XML	application/xml	.xml	No

Example 3.1. JSON Request with Headers

```
POST /v1.0/token HTTP/1.1
Host: {host name TBD}
Content-Type: application/json
Accept: application/xml
```

```
<?xml version="1.0" encoding="UTF-8"?>

<authCredentials
    xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    password="P@ssword1"
    username="testuser"
    client_secret="0923899flewriudsb"
    client_id="8972348923400fdshasdf"
    grant_type="PASSWORD" />
```

Example 3.2. XML Response with Headers

```
HTTP/1.1 200 OKAY
Date: Mon, 12 Nov 2010 15:55:01 GMT
Server: Apache
Content-Length:
Content-Type: application/xml; charset=UTF-8
```

```xml
<?xml version="1.0" encoding="UTF-8"?>

<auth xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <access_token
    expires_in="3600"
    id="ab48a9efdfedb23ty3494" />
 <refresh_token
    id="8792gdfskjbadf98y234r" />
 <user
     xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
     customerId="RCN-000-000-000"
     username="jqsmith" />
</auth>
```


3.2. Content Compression

Request and response body data my be encoded with gzip compression in order to accelerate interactive performance of API calls and responses. This is controlled using the `Accept-Encoding` header on the request from the client and indicated by the `Content-Encoding` header in the server response. Unless the header is explicitly set, encoding defaults to disabled.

Table 3.2. Compression Headers

Header Type	Name	Value
HTTP/1.1 Request	Accept-Encoding	gzip
HTTP/1.1 Response	Content-Encoding	gzip

3.3. API Version

The IdM API uses a URI versioning scheme. The first element of the path contains the target version identifier (e.g. https://idm.api.rackspace.com/v1.0/...) All requests (except to query for version - see below) must contain a target version. Any features or functionality changes that would necessitate a break in API-compatibility will require a new version, which will result in the URI version being updated accordingly. When new API versions are released, older versions will be marked as `Deprecated`. Rackspace will work with developers and partners to ensure there is adequate time to migrate to the new version before deprecated versions are discontinued.

3.3.1. List Versions

Verb	URI	Description
GET	https://{Hostname TBD}/	Retrieve a list of IdM API versions.

Normal Response Code(s):200

Error Response Code(s): badRequest (400), idmFault (500), serviceUnavailable(503)

Your application can programmatically determine available API versions by performing a **GET** on the root URL (https://idm.api.rackspace.com/).

This operation does not require a request body.

Example 3.3. XML Versions Response


```
<?xml version="1.0" encoding="UTF-8"?>

<versions xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <version
    id="v0.9"
    status="DEPRECATED"
    href="/v0.9" />
 <version
    id="v1.0"
    status="CURRENT"
    href="/v1.0" />
 <version
    id="v1.1"
    status="BETA"
    href="/v1.1" />
</versions>
```


3.3.2. Get Version Details

Verb	URI	Description
GET	https://{Hostname TBD}/v1.0/	Retrieve API Version details

Normal Response Code(s):200

Error Response Code(s): badRequest (400), idmFault (500), serviceUnavailable(503)

You can also obtain additional information about a specific version by performing a **GET** on the base version URL (e.g. https://idm.api.rackspace.com/v1.0). Version request URLs should always end with a trailing slash (/). If the slash is omitted, the server may respond with a 302 redirection request. Format extensions may be placed after the slash (e.g. https://idm.api.rackspace.com/v1.0/.xml).

This operation does not require a request body.

Example 3.4. XML Version Response


```
<?xml version="1.0" encoding="UTF-8"?>

<version xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    wadl="http://docs.rackspacecloud.com/idm/api/v1.0/application.wadl"
    docURL="http://docs.rackspacecloud.com/idm/api/v1.0/idm-devguide.pdf"
    status="CURRENT"
    id="v1.0" />
```


The detailed version response contains a pointer to both the human readable and a machine processable description of the API service. The machine processable description is written in the Web Application Description Language (WADL).

3.4. Faults

When an error occurs the system will return an HTTP error response code denoting the type of error. The system will also return additional information about the fault in the body of the response.

Example 3.5. XML Fault Response

```
<?xml version="1.0" encoding="UTF-8"?>

<idmFault xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    code="500">
    <message>Fault</message>
    <details>Error Details...</details>
</idmFault>
```

Example 3.6. JSON Fault Response

```
{
  "message": "Fault",
  "details": "Error Details...",
  "code": 500
}
```

The error code is returned in the body of the response for convenience. The message section returns a human readable message. The details section is optional and may contain useful information for tracking down an error (e.g a stack trace).

The root element of the fault (e.g. idmFault) may change depending on the type of error. The following is an example of an itemNotFound error.

Example 3.7. XML Not Found Fault

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<itemNotFound xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    code="404">
    <message>Item not found.</message>
    <details>Error Details...</details>
</itemNotFound>
```

Example 3.8. JSON Not Found Fault

```
{
  "message": "Item not found.",
  "details": "Error Details...",
  "code": 404
}
```

The following is a list of possible fault types along with their associated error codes.

Table 3.3. Fault Types

Fault Element	Associated Error Code	Expected in All Requests
idmFault	500, 400	✓
serviceUnavailable	503	✓
unauthorized	401	✓
badRequest	400	✓
passwordValidation	400	
userDisabled	403	
forbidden	403	
itemNotFound	404	
clientConflict	409	
customerConflict	409	
emailConflict	409	
usernameConflict	409	

From an XML schema perspective, all API faults are extensions of the base fault type idmFault. When working with a system that binds XML to actual classes (such as JAXB), one should be capable of using idmFault as a “catch-all” if there's no interest in distinguishing between individual fault types.

3.5. Getting Started

The first step to using the Global Auth API is registering your application with Global Auth. The Global Auth Team will need the following information in order to register your application.

Table 3.4. Application Information needed to register with Global Auth API

Item	Description	Example
Name	The name of the application	Cloud Servers API
Simple Name	A continuous lower case name	cloud_servers_api
Description	A human readable description of what the application does, which should be presentable to end users.	Cloud Servers API is ...

Once the Global Auth Team has this information we will add your application to the Global Auth System and send you the clientId and clientSecret that you will need in order to authenticate with the Global Auth API (see section below for Authentication).

Example 3.9. JSON Request with Headers

```
POST /v1.0/token HTTP/1.1
Host: {host name TBD}
Content-Type: application/json
Accept: application/xml
```

```
<?xml version="1.0" encoding="UTF-8"?>

<authCredentials
    xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    password="P@ssword1"
    username="testuser"
    client_secret="0923899flewriudsb"
    client_id="8972348923400fdshasdf"
    grant_type="PASSWORD" />
```

Example 3.10. XML Response with Headers

```
HTTP/1.1 200 OKAY
Date: Mon, 12 Nov 2010 15:55:01 GMT
Server: Apache
Content-Length:
Content-Type: application/xml; charset=UTF-8
```

```
<?xml version="1.0" encoding="UTF-8"?>

<auth xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <access_token
    expires_in="3600"
    id="ab48a9efdfedb23ty3494" />
 <refresh_token
    id="8792gdfskjbadf98y234r" />
 <user
     xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
     customerId="RCN-000-000-000"
     username="jqsmith" />
</auth>
```

3.6. Authorization Header

Most calls made against the Global Auth API require the addition of an authorization header in the request.

The format of the authoriztion header is the word "OAuth" followed by a space followed by the access token id that belongs to the entity making the call. For example, here is an sample request for getting a user's details.

Example 3.11. Authorization Header Request Format

```
GET /v1.0/users/joeuser HTTP/1.1
Host: {host name TBD}
Accept: application/xml
Authorization: OAuth ab48a9efdfedb23ty3494
```

4. Service Developer Operations

4.1. Overview

The operations described in this chapter allow service developers to get and validate access tokens, manage users, manage clients, and password related operations (get password rules, valid password).

4.2. Token Operations

4.2.1. Authenticate

Verb	URI	Description
POST	/token	Authenticate a user or application and generate an access token.

Normal Response Code(s):200, 203

Error Response Code(s): unauthorized (401), userDisabled (403), badRequest (400), idmFault (500), serviceUnavailable(503)

Example 4.1. XML User Auth Request

```
 <?xml version="1.0" encoding="UTF-8"?>

<authCredentials
    xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    password="P@ssword1"
    username="testuser"
    client_secret="0923899flewriudsb"
    client_id="8972348923400fdshasdf"
    grant_type="PASSWORD" />
```

Example 4.2. JSON User Auth Request

```
{
  "grant_type": "PASSWORD",
  "client_id": "8972348923400fdshasdf",
  "client_secret": "0923899flewriudsb",
  "username": "testuser",
  "password": "P@ssword1"
}
```

Example 4.3. XML User Auth Response

```
<?xml version="1.0" encoding="UTF-8"?>

<auth xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <access_token
```

```
    expires_in="3600"
    id="ab48a9efdfedb23ty3494" />
 <refresh_token
    id="8792gdfskjbadf98y234r" />
 <user
     xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
     customerId="RCN-000-000-000"
     username="jqsmith" />
</auth>
```


Example 4.4. JSON User Auth Response


```
{
  "accessToken": {
    "id": "ab48a9efdfedb23ty3494",
    "expiresIn": 3600
  },
  "refreshToken": {
    "id": "8792gdfskjbadf98y234r"
  },
  "user" : {
    "username": "jqsmith",
    "customerId": "RCN-000-000-000",
  }
}
```


Example 4.5. XML Application Auth Request


```
 <?xml version="1.0" encoding="UTF-8"?>

<authCredentials
    xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    client_secret="0923899flewriudsb"
    client_id="8972348923400fdshasdf"
    grant_type="CLIENT_CREDENTIALS" />
```


Example 4.6. JSON Application Auth Request


```
{
  "grant_type": "CLIENT_CREDENTIALS",
  "client_id": "8972348923400fdshasdf",
  "client_secret": "0923899flewriudsb"
}
```


Example 4.7. XML Application Auth Response


```
<?xml version="1.0" encoding="UTF-8"?>

<auth xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <access_token
    expires_in="3600"
```

```
    id="ab48a9efdfedb23ty3494" />
 <client
        customerId="RACKSPACE"
        clientId="ab4820dhcb39347" />
</auth>
```

Example 4.8. JSON Application Auth Response

```
{
  "accessToken": {
    "id": "ab48a9efdfedb23ty3494",
    "expiresIn": 3600
  },
  "client": {
    "clientId": "ab4820dhcb39347",
    "customerId": "RACKSPACE",
   }
}
```

4.2.2. Validate Token

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
GET	/token/*tokenId*	Check that a token is valid and return the token details.

Normal Response Code(s):200, 203

Error Response Code(s): unauthorized (401), forbidden (403), userDisabled(403), badRequest (400), itemNotFound (404), idmFault(500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.9. XML Validate Token Response

```
<?xml version="1.0" encoding="UTF-8"?>

<auth xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <access_token
    expires_in="3600"
    id="ab48a9efdfedb23ty3494" />
 <user
     customerId="RCN-000-000-000"
     username="jqsmith" />
</auth>
```

Example 4.10. JSON Validate Token Response

```
{
    "access_token": {
```

```
        "id": "ab48a9efdfedb23ty3494",
        "expires_in": "3600"
    },
    "user" : {
          "username": "jqsmith",
          "customerId": "RCN-000-000-000",
     }
}
```


4.2.3. Check if Token has been Provisioned for Service

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/tokens/*tokenId*/services/*serviceId*	Check if token has been provisioned for service

Normal Response Code(s):200 404

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.2.4. Check if Token has Permission for Service

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/tokens/*tokenId*/services/*serviceId*/permissions/*permissionId*	Checks if token has permission for a service

Normal Response Code(s):200 404

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.2.5. Revoke Token

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
DELETE	/token/*tokenId*	Revoke an existing token.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), forbidden (403), userDisabled(403), badRequest (400), itemNotFound (404), idmFault(500), serviceUnavailable(503)

This operation does not require a request body.

4.3. Customer Operations

4.3.1. Get a Customer

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*	Get a customer.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), forbidden(403), itemNotFound(404), badRequest (400), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.11. XML Customer Response


```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<customer xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="false"
    locked="false"
    iname="@Rackspace*Customer"
    inum="@FFFF.FFFF.FFFF.FFFF!0ABE.34EC"
    customerId="RCN-123-549-034" />
```


Example 4.12. JSON Customer Response


```
{
  "customerId": "RCN-123-549-034",
  "inum": "@FFFF.FFFF.FFFF.FFFF!0ABE.34EC",
  "iname": "@Rackspace*Customer",
  "locked": false,
  "softDeleted": false
}
```

4.3.2. Set Customer Lock Status

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
PUT	/customers/*customerId*/actions/lock	Lock or unlock a customer.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), forbidden(403), itemNotFound(404), badRequest (400), idmFault (500), serviceUnavailable(503)

Example 4.13. XML Customer Lock Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<customer xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    locked="true"  />
```

Example 4.14. JSON Customer Lock Request

```
{
    "locked": true
}
```

Example 4.15. XML Customer Lock Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<customer xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    locked="true"  />
```

Example 4.16. JSON Customer Lock Response

```
{
    "locked": true
}
```

4.3.3. Delete a Customer

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
DELETE	/customers/*customerId*	Delete a Customer.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.3.4. Set Customer Password Rotation Policy

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
PUT	/customers/*customerId*/passwordrotationpolicy	Sets the password rotation policy for a customer.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), forbidden (403), userDisabled(403), badRequest (400), itemNotFound (404), idmFault(500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.17. XML Password Rotation Policy Request


```
<?xml version="1.0" encoding="UTF-8"?>

<passwordRotationPolicy xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    enabled="true"
    duration="90" />
```


Example 4.18. JSON Password Rotation Policy Request


```
{
  "enabled": true,
  "duration": "60"
}
```


Example 4.19. XML Password Rotation Policy Response


```
<?xml version="1.0" encoding="UTF-8"?>

<passwordRotationPolicy xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    enabled="true"
    duration="90" />
```

Example 4.20. JSON Password Rotation Policy Response


```
{
  "enabled": true,
  "duration": "60"
}
```


4.4. User Operations

4.4.1. Create the First User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
POST	/users	Create a new customer entry and add the user as an Admin user.

The Username attribute, CustomerId attribute and Password element are required in this request. If a blank password is passed in the Password element, the API will generate a random password for the user. The prefLanguage attribute defaults to "US_en" and the timeZone attribute defaults to "America/Chicago".

Normal Response Code(s):201

Error Response Code(s): unauthorized (401), badRequest (400), passwordValidation(400), forbidden (403), itemNotFound (404), customerConflict(409), usernameConflict(409), emailConflict(409), idmFault (500), serviceUnavailable(503)

Example 4.21. XML Add First User Request


```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    region="SAT" prefLanguage="US_en" timeZone="America/Chicago"
    displayName="John Smith" lastName="Smith"
    middleName="Quincy" firstName="John"
    personId="RPN-111-111-111"
    email="john.smith@example.org"
    customerId="RCN-000-000-000" username="jqsmith">
    <secret secretAnswer="Francis"
        secretQuestion="What is your dogs name?" />
    <password password="C@n+f001me!" />
</user>
```

Example 4.22. JSON Add First User Request


```
{
  "secret": {
    "secretAnswer": "Francis",
    "secretQuestion": "What is your dogs name?"
  },
  "password": {
    "password": "P@ssword1"
  },
  "username": "jqsmith",
  "customerId": "RCN-000-000-000",
  "email": "john.smith@example.org",
  "personId": "RPN-111-111-111",
  "firstName": "John",
  "middleName": "Quincy",
  "lastName": "Smith",
  "displayName": "John Smith",
  "prefLanguage": "US_en",
  "region": "America/Chicago"
}
```


Example 4.23. XML Add First User Response


```
<?xml version="1.0" encoding="UTF-8"?>

<user
    xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="false" locked="false"
    status="ACTIVE" timeZone="America/Chicago"
    region="SAT" iname="@Example.Smith*John"
    inum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111"
    prefLanguage="US_en" displayName="John Smith"
    lastName="Smith" middleName="Quincy"
    firstName="John" personId="RPN-111-111-111"
    email="john.smith@example.org"
    customerInum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE"
    customerId="RCN-000-000-000"
    username="jqsmith">
    <secret secretAnswer="Francis"
        secretQuestion="What is your dogs name?" />
    <password password="C@n+f001me!" />
    <groups>
        <clientGroup type="RackspaceDefined"
            customerInum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE"
            customerId="RCN-000-000-000" name="Admin" />
    </groups>
</user>
```


Example 4.24. JSON Add First User Response


```
{
  "secret": {
    "secretQuestion": "What is your dogs name?",
    "secretAnswer": "sicnarF"
```

```
  },
  "password": {
    "password": "C@n+f001me!"
  },
  "groups": {
    "clientGroup": [
      {
        "permissions": {},
        "name": "Admin",
        "customerId": "RCN-234-592-018",
        "customerInum": "@FFFF.FFFF.FFFF.FFFF!2398.1FCB",
        "type": "RackspaceDefined"
      }
    ]},
  "username": "jqsmith",
  "customerId": "RCN-000-000-000",
  "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE",
  "email": "john.smith@example.org",
  "personId": "RPN-111-111-111",
  "firstName": "John",
  "middleName": "Quincy",
  "lastName": "Smith",
  "displayName": "John Smith",
  "prefLanguage": "US_en",
  "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111",
  "timeZone": "America/Chicago",
  "region": "SAT",
  "status": "ACTIVE",
  "locked": false,
  "softDeleted": false
}
```


4.4.2. Add a User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
POST	/customers/*customerId*/users	Create a new user. If a blank password is passed in, the API generates a random password for the user.

The Username attribute and Password element are required in this request. If a blank password is passed in the Password element, the API will generate a random password for the user. The prefLanguage attribute defaults to "US_en" and the timeZone attribute defaults to "America/Chicago".

Normal Response Code(s):201

Error Response Code(s): unauthorized (401), badRequest (400), passwordValidation(400), forbidden (403), itemNotFound (404), usernameConflict(409), emailConflict(409), idmFault (500), serviceUnavailable(503)

Example 4.25. XML Add User Request

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    region="SAT" prefLanguage="US_en" timeZone="America/Chicago"
    displayName="John Smith" lastName="Smith"
    middleName="Quincy" firstName="John"
    personId="RPN-111-111-111"
    email="john.smith@example.org"
    customerId="RCN-000-000-000" username="jqsmith">
    <secret secretAnswer="Francis"
        secretQuestion="What is your dogs name?" />
    <password password="C@n+f001me!" />
</user>
```

Example 4.26. JSON Add User Request

```
{
  "secret": {
    "secretAnswer": "Francis",
    "secretQuestion": "What is your dogs name?"
  },
  "password": {
    "password": "P@ssword1"
  },
  "username": "jqsmith",
  "customerId": "RCN-000-000-000",
  "email": "john.smith@example.org",
  "personId": "RPN-111-111-111",
  "firstName": "John",
  "middleName": "Quincy",
  "lastName": "Smith",
  "displayName": "John Smith",
  "prefLanguage": "US_en",
  "region": "America/Chicago"
}
```

Example 4.27. XML Add User Response

```
<?xml version="1.0" encoding="UTF-8"?>

<user
    xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
 softDeleted="false" locked="false"
    status="ACTIVE" timeZone="America/Chicago"
 region="SAT" iname="@Example.Smith*John"
 inum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111"
 prefLanguage="US_en" displayName="John Smith"
 lastName="Smith" middleName="Quincy"
 firstName="John" personId="RPN-111-111-111"
 email="john.smith@example.org"
 customerInum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE"
 customerId="RCN-000-000-000"
 username="jqsmith">
 <secret secretAnswer="Francis"
       secretQuestion="What is your dogs name?" />
    <password password="C@n+f001me!" />
```

```
</user>
```


Example 4.28. JSON Add User Response


```
{
  "secret": {
    "secretQuestion": "What is your dogs name?",
    "secretAnswer": "Francis"
  },
  "password": {
    "password": "C@n+f001me!"
  },
  "username": "jqsmith",
  "customerId": "RCN-000-000-000",
  "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE",
  "email": "john.smith@example.org",
  "personId": "RPN-111-111-111",
  "firstName": "John",
  "middleName": "Quincy",
  "lastName": "Smith",
  "displayName": "John Smith",
  "prefLanguage": "US_en",
  "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111",
  "timeZone": "America/Chicago",
  "region": "SAT",
  "status": "ACTIVE",
  "locked": false,
  "softDeleted": false
}
```


4.4.3. Get a User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
GET	/customers/*customerId*/users/*username*	Get a user.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.29. XML Get User Response


```
<?xml version="1.0" encoding="UTF-8"?>

<user
    xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="false" locked="false"
```

```
    status="ACTIVE" region="America/Chicago"
    inum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111"
    prefLanguage="US_en" displayName="John Smith"
    lastName="Smith" middleName="Quincy"
    firstName="John" personId="RPN-111-111-111"
    email="john.smith@example.org"
    customerInum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE"
    customerId="RCN-000-000-000"
    username="jqsmith">
    <groups>
        <clientGroup
            customerId="RCN-000-000-000",
            clientId="ab4820dhcb39347",
            type="Role"
            name="Admin" />
    </groups>
</user>
```

Example 4.30. JSON Get User Response

```
{
  "groups": {
    "clientGroup": [
      {
        "customerId" : "RCN-000-000-000",
        "clientId" : "ab4820dhcb39347",
        "name": "Admin",
        "type": "Role"
      }
    ]},
  "username": "jqsmith",
  "customerId": "RCN-000-000-000",
  "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE",
  "email": "john.smith@example.org",
  "personId": "RPN-111-111-111",
  "firstName": "John",
  "middleName": "Quincy",
  "lastName": "Smith",
  "displayName": "John Smith",
  "prefLanguage": "US_en",
  "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111",
  "region": "America/Chicago",
  "status": "ACTIVE",
  "locked": false,
  "softDeleted": false
}
```

4.4.4. Get a User by Mosso ID

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/mosso/*mossoId*	Get a user with mossoId.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.31. XML Get User By MossoId Response


```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    username="qauser"
    customerId="RCN-QATestingCustomer"
    customerInum="@!FFFF.FFFF.FFFF.FFFF!EEEE.999"
    email="quauser@example.com"
    personId="RPN-qauser-000-10001"
    firstName="qauser"
    middleName="none"
    lastName="Tester"
    displayName="qauser"
    prefLanguage="en-us"
    inum="@!FFFF.FFFF.FFFF.FFFF!EEEE.999!10001"
    status="ACTIVE"
    locked="false"
    softDeleted="false"
    country="USA"
    timeZone="America/Chicago"
    mossoId="999112358"
    nastId="nastid-qauser"/>
```


Example 4.32. JSON Get User By MossoId Response


```
{
   "username": "qauser",
   "customerId": "RCN-QATestingCustomer",
   "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.999",
   "email": "quauser@example.com",
   "personId": "RPN-qauser-000-10001",
   "firstName": "qauser",
   "middleName": "none",
   "lastName": "Tester",
   "displayName": "qauser",
   "prefLanguage": "en-us",
   "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.999!10001",
   "status": "ACTIVE",
   "locked": false,
   "softDeleted": false,
   "country": "USA",
   "timeZone": "America/Chicago",
   "mossoId": 999112358,
   "nastId": "nastid-qauser"
}
```

4.4.5. Get a User by NAST ID

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/nast/*nastId*	Get a user with a nastId.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.33. XML Get User By NastId Response

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    username="qauser"
    customerId="RCN-QATestingCustomer"
    customerInum="@!FFFF.FFFF.FFFF.FFFF!EEEE.999"
    email="quauser@example.com"
    personId="RPN-qauser-000-10001"
    firstName="qauser"
    middleName="none"
    lastName="Tester"
    displayName="qauser"
    prefLanguage="en-us"
    inum="@!FFFF.FFFF.FFFF.FFFF!EEEE.999!10001"
    status="ACTIVE"
    locked="false"
    softDeleted="false"
    country="USA"
    timeZone="America/Chicago"
    mossoId="999112358"
    nastId="nastid-qauser"/>
```

Example 4.34. JSON Get User By NastId Response

```
{
   "username": "qauser",
   "customerId": "RCN-QATestingCustomer",
   "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.999",
   "email": "quauser@example.com",
   "personId": "RPN-qauser-000-10001",
```

```
   "firstName": "qauser",
   "middleName": "none",
   "lastName": "Tester",
   "displayName": "qauser",
   "prefLanguage": "en-us",
   "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.999!10001",
   "status": "ACTIVE",
   "locked": false,
   "softDeleted": false,
   "country": "USA",
   "timeZone": "America/Chicago",
   "mossoId": 999112358,
   "nastId": "nastid-qauser"
}
```

4.4.6. Get a User by Username

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/users/*username*	Get a user with a username.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.35. XML Get User By Username Response

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    username="qauser"
    customerId="RCN-QATestingCustomer"
    customerInum="@!FFFF.FFFF.FFFF.FFFF!EEEE.999"
    email="quauser@example.com"
    personId="RPN-qauser-000-10001"
    firstName="qauser"
    middleName="none"
    lastName="Tester"
    displayName="qauser"
    prefLanguage="en-us"
    inum="@!FFFF.FFFF.FFFF.FFFF!EEEE.999!10001"
    status="ACTIVE"
    locked="false"
    softDeleted="false"
    country="USA"
    timeZone="America/Chicago"
```

```
    mossoId="999112358"
    nastId="nastid-qauser"/>
```

Example 4.36. JSON Get User By Username Response

```
{
   "username": "qauser",
   "customerId": "RCN-QATestingCustomer",
   "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.999",
   "email": "quauser@example.com",
   "personId": "RPN-qauser-000-10001",
   "firstName": "qauser",
   "middleName": "none",
   "lastName": "Tester",
   "displayName": "qauser",
   "prefLanguage": "en-us",
   "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.999!10001",
   "status": "ACTIVE",
   "locked": false,
   "softDeleted": false,
   "country": "USA",
   "timeZone": "America/Chicago",
   "mossoId": 999112358,
   "nastId": "nastid-qauser"
}
```

4.4.7. Update a User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
PUT	/customers/*customerId*/users/*userId*	Update a user.

Normal Response Code(s):200, 203

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), emailConflict (409), idmFault (500), serviceUnavailable(503)

Example 4.37. XML Update User Request

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    email="john.smith@somenewemail.org"
    username="jqsmith" />
```

Example 4.38. JSON Update User Request

```
{
```

```
  "email": "john.smith@example.org",
  "username": "jqsmith"
}
```

Example 4.39. XML Update User Response

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    timeZone="America/Chicago" prefLanguage="US_en"
    displayName="John Smith" lastName="Smith"
    middleName="Quincy" firstName="John"
    region="SAT" personId="RPN-111-111-111"
    email="john.smith@somenewemail.org"
    customerId="RCN-000-000-000" username="jqsmith" />
```

Example 4.40. JSON Update User Response

```
{
  "username": "jqsmith",
  "customerId": "RCN-000-000-000",
  "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE",
  "email": "john.smith@somenewemail.org",
  "personId": "RPN-111-111-111",
  "firstName": "John",
  "middleName": "Quincy",
  "lastName": "Smith",
  "displayName": "John Smith",
  "prefLanguage": "US_en",
  "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111",
  "iname": "@Example.Smith*John",
  "region": "America/Chicago",
  "status": "ACTIVE",
  "locked": false,
  "softDeleted": false
}
```

4.4.8. Soft Delete a User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
PUT	/customers/*customerId*/users/*userId*/softDelete	Set a User's softDeleted Flag

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.41. XML User Soft Delete Request

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="true" />
```

Example 4.42. JSON User Soft Delete Request

```
{
  "softDeleted": "true"
}
```

Example 4.43. XML User Soft Delete Response

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="true" />
```

Example 4.44. JSON User Soft Delete Response

```
{
  "softDeleted": "true"
}
```

4.4.9. Delete User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/customers/*customerId*/users/*userId*	Delete a user.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.4.10. Get User List

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
GET	/customers/*customerId*/users/?offset=*x*&limit=*y*	Gets a page of users where *x* is the offset and *y* is the limit of records to return.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.45. XML Get User List Response

```
<?xml version="1.0" encoding="UTF-8"?>

<users xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    limit="10"
    offset="0"
    totalRecords="2">
    <user
        softDeleted="false" locked="false"
        status="ACTIVE" region="America/Chicago"
        iname="@Example.Smith*John"
        inum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111"
        prefLanguage="US_en" displayName="John Smith"
        lastName="Smith" middleName="Quincy"
        firstName="John" personId="RPN-111-111-111"
        email="john.smith@example.org"
        customerInum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE"
        customerId="RCN-000-000-000"
        username="jqsmith" />
    <user softDeleted="false" locked="false"
        status="ACTIVE" region="America/Chicago"
        iname="@Example.Anderson*Bob"
        inum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!2222"
        prefLanguage="US_en" displayName="Bob Anderson"
        lastName="Anderson" middleName="Mark"
        firstName="Bob" personId="RPN-111-111-222"
        email="bob.anderson@example.org"
        customerInum="@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE"
        customerId="RCN-000-000-000"
        username="bmanderson" />
</users>
```

Example 4.46. JSON Get User List Response

```
{"user": [
    {
      "username": "jqsmith",
      "customerId": "RCN-000-000-000",
      "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE",
      "email": "john.smith@example.org",
      "personId": "RPN-111-111-111",
      "firstName": "John",
      "middleName": "Quincy",
      "lastName": "Smith",
```

```
      "displayName": "John Smith",
      "prefLanguage": "US_en",
      "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!1111",
      "iname": "@Example.Smith*John",
      "region": "America/Chicago",
      "status": "ACTIVE",
      "locked": false,
      "softDeleted": false
    },
    {
      "username": "bmanderson",
      "customerId": "RCN-000-000-000",
      "customerInum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE",
      "email": "bob.anderson@example.org",
      "personId": "RPN-111-111-222",
      "firstName": "Bob",
      "middleName": "Mark",
      "lastName": "Anderson",
      "displayName": "Bob Anderson",
      "prefLanguage": "US_en",
      "inum": "@!FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!2222",
      "iname": "@Example.Anderson*Bob",
      "timeZone": "America/Chicago",
      "region": "SAT",
      "status": "ACTIVE",
      "locked": false,
      "softDeleted": false
    }
  ],
  "totalRecords": 2,
  "offset": 0,
  "limit": 10
}
```


4.4.11. Add User to Customer IdM Group

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
PUT	/customers/*customerId*/users/*userId*/groups/*groupName*	Add a user to a Customer IdM group

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

4.4.12. Remove User from Customer IdM Group

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
DELETE	/customers/*customerId*/users/*userId*/groups/*groupName*	Remove a user from a customer IdM group

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

4.4.13. Get the Groups That a User is a Member Of

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
GET	/customers/*customerId*/users/*userId*/groups?clientId=*clientId*&type=*type*	Get a user's groups optionally filtered by *clientId* and/or *type*

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.47. XML User Groups Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<clientGroups xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <clientGroup customerId="RCN-000-000-000"
        clientId="IDM"
        name="Idm Admin"
        type="Role"  />
</clientGroups>
```

Example 4.48. JSON User Groups Response

```
{"clientGroup": [
    {
        "customerId" : "RCN-000-000-000",
        "clientId" : "IDM",
        "name" : "Idm Admin",
        "type" : "Role"
    }]
}
```

4.4.14. Generate/Reset Auth Service Key

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
POST	/customers/*customerId*/users/*username*/key	Generate or reset a user's Auth key.

Normal Response Code(s):200, 203

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.49. XML User Auth Key Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<userApiKey xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    apiKey="403958809d0809834e0809808a" />
```

Example 4.50. JSON User Auth Key Response

```
{
  "apiKey": "403958809d0809834e0809808a"
}
```

4.4.15. Set Auth Service Key

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
PUT	/customers/*customerId*/users/*username*/key	Set a user's Auth key.

Normal Response Code(s):200, 203

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.51. XML Set Auth Key Reqeust

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<userApiKey xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    apiKey="403958809d0809834e0809808a" />
```

Example 4.52. JSON Set Auth Key Request

```
{
  "apiKey": "403958809d0809834e0809808a"
}
```

Example 4.53. XML Set Auth Key Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<userApiKey xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    apiKey="403958809d0809834e0809808a" />
```

Example 4.54. JSON Set Auth Key Response

```
{
  "apiKey": "403958809d0809834e0809808a"
}
```

4.4.16. Set User Lock

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
PUT	/customers/*customerId*/users/*username*/lock	Set a user's lock

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.55. XML Set User Lock Request

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    locked="true" />
```

Example 4.56. JSON Set User Lock Request

```
{
  "locked": "true"
}
```

Example 4.57. XML Set User Lock Response

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    locked="true" />
```

DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - DRAFT - D

Example 4.58. JSON Set User Lock Response

```
{
  "locked": "true"
}
```

4.4.17. Set User Password

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
PUT	/customers/*customerId*/users/*username*/password?recovery=*boolean*	Sets a user's password.

This method takes an optional boolean query parameter (recovery). True indicates that the password change is for Forget Password and will not require the old password. False is the normal reset route when an old password is used to reset the password.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), passwordValidation(400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.59. XML Set User Password Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<userCredentials xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <newPassword password="newpassword" />
    <currentPassword password="oldpassword" />
</userCredentials>
```

Example 4.60. JSON Set User Password Request

```
{
  "newPassword": {
    "password": "newpassword"
  },
  "currentPassword": {
    "password": "oldpassword"
  }
}
```

Example 4.61. XML User Password Response

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<userPassword xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    password="newpassword" />
```

Example 4.62. JSON User Password Response

```
{
  "password": "newpassword"
}
```

4.4.18. Reset User Password

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
POST	/customers/*customerId*/users/*username*/password	Reset a user's password.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.63. XML Reset User Password Response

```
<?xml version="1.0" encoding="UTF-8"?>

<userPassword xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    password="7ud$dnF" />
```

Example 4.64. JSON Reset User Password Response

```
{
  "password": "7ud$dnF"
}
```

4.4.19. Get a User Password

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/users/*userId*/password	Get a user's password.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.65. XML Get User Password Response

```
<?xml version="1.0" encoding="UTF-8"?>

<userPassword xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    password="newpassword" />
```

Example 4.66. JSON Get User Passwrord Response

```
{
  "password": "newpassword"
}
```

4.4.20. Get Password Recovery Token

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
POST	/customers/*customerId*/users/*username*/password/recoveryToken	Get a token that can be used to reset a user's password.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.67. XML Password Recovery Token Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<token xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    expires_in="3600"
    id="309487987f0892397a9439875900b" />
```

Example 4.68. JSON Password Recovery Token Response

```
{
  "id": "309487987f0892397a9439875900b",
  "expiresIn": 3600
}
```

4.4.21. Set User Status

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
PUT	/customers/*customerId*/users/*username*/status	Set a user's status to ACTIVE or INACTIVE

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.69. XML Set User Status Request

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    status="ACTIVE" />
```

Example 4.70. JSON Set User Status Request

```
{
  "status": "INACTIVE"
}
```

Example 4.71. XML Set User Status Response

```
<?xml version="1.0" encoding="UTF-8"?>

<user xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    status="ACTIVE" />
```

Example 4.72. JSON Set User Status Response

```
{
  "status": "INACTIVE"
}
```

4.4.22. Set User Secret

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
PUT	/customers/*customerId*/users/*username*/secret	Sets a user's secret question and answer.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.73. XML Set User Secret Request

```
<?xml version="1.0" encoding="UTF-8"?>

<userSecret xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    secretAnswer="Not a very good one."
    secretQuestion="Is this a secret question?" />
```

Example 4.74. JSON Set User Secret Request

```
{
  "secretQuestion": "Is this a secret question?",
  "secretAnswer": "Not a very good one."
}
```

Example 4.75. XML Set User Secret Response

```
<?xml version="1.0" encoding="UTF-8"?>

<userSecret xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    secretAnswer="Not a very good one."
    secretQuestion="Is this a secret question?" />
```

Example 4.76. JSON Set User Secret Response

```
{
  "secretQuestion": "Is this a secret question?",
  "secretAnswer": "Not a very good one."
}
```

4.4.23. Get the Services that a User has Provisioned

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/users/*userId*/services	Get the services that a user has provisioned

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.77. XML Get User Services Response

```
<?xml version="1.0" encoding="UTF-8"?>

<clients xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <client
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="ab4820dhcb39347">
 </client>
 <client
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0002"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="632h389cv902bde">
 </client>
</clients>
```

Example 4.78. JSON Get User Services Response

```
{
  "client": [
    {
      "clientId": "ab4820dhcb39347",
      "customerId": "RCN-000-000-000",
      "name": "Test Application2",
      "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001",
      "status": "ACTIVE",
      "locked": false,
      "softDeleted": false
    },
    {
      "clientId": "632h389cv902bde",
      "customerId": "RCN-000-000-000",
      "name": "Test Application2",
      "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0002",
      "status": "ACTIVE",
      "locked": false,
      "softDeleted": false
    }
  ]
}
```

4.4.24. Provision a Service for a User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
POST	/customers/*customerId*/users/*userId*/services	Get the services that a user has provisioned

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.79. XML Provision User Service Request

```
<?xml version="1.0" encoding="UTF-8"?>

<client xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    clientId="ab4820dhcb39347" />
```

Example 4.80. JSON Get Provision User Service Request

```
{
  "clientId": "ab4820dhcb39347",
}
```

4.4.25. Remove Provision a Service for a User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/customers/*customerId*/users/*userId*/services/*serviceId*	Removes a provisioned service from a user

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.4.26. Grant a User Permission for Service

This operation requires an authorization header. See Section 3.6, "Authorization Header" for details.

Note

This is a management API function.

Verb	URI	Description
POST	/customers/*customerId*/users/*userId*/services/*serviceId*/permissions	Grant a user permission for a service

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.81. XML Grant User Permission Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<permission xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    clientId="IDM"
    customerId="RCN-000-000-000"
    permissionId="addCustomer" />
```

Example 4.82. JSON Grant User Permission Request

```
{
  "permissionId": "addCustomer",
  "customerId": "RCN-000-000-000",
  "clientId": "IDM"
}
```

4.4.27. Revoke a User Permission for Service

This operation requires an authorization header. See Section 3.6, "Authorization Header" for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/customers/*customerId*/users/*userId*/services/*serviceId*/permissions/*permissionId*	Revoke a user permission for a service

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.4.28. Check if User has Permission for Service

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/users/*userId*/services/*serviceId*/permissions/*permissionId*	Check to see if user has permision for a service

Normal Response Code(s):200 404

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.4.29. Get a List of Delegated Refresh Tokens for a User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/users/*userId*/delegatedrefreshtokens	Get a user's delegated refresh tokens

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.83. XML Delegated Tokens Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<delegatedTokens  xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <delegatedToken
        expires="2010-11-02T03:32:15-05:00"
        id="309487987f0892397a9439875900b"
        clientId="ab4820dhcb39347" />
    <delegatedToken
        expires="2010-11-02T03:32:15-05:00"
```

```
        id="27aaf6789a9dcb789879972729abf"
        clientId="af62785da123567" />
</delegatedTokens>
```

Example 4.84. JSON Delegated Tokens Response

```
{ "delegatedToken": [
   {
     "id": "309487987f0892397a9439875900b",
     "expires": "2010-11-02T03:32:15-05:00",
     "clientId": "ab4820dhcb39347"
   },
   {
     "id": "27aaf6789a9dcb789879972729abf",
     "expires": "2010-11-02T03:32:15-05:00",
     "clientId": "af62785da123567"
   }]
}
```

4.4.30. Get a Delegated Refresh Token for a User

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/users/*userId*/delegatedrefreshtokens/*tokenId*	Get a user's delegated refresh token

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.85. XML Delegated Token Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<token xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    expires="2010-11-02T03:32:15-05:00"
    id="309487987f0892397a9439875900b"
    clientId="ab4820dhcb39347" />
```

Example 4.86. JSON Delegated Token Response

```
{
  "id": "309487987f0892397a9439875900b",
  "expires": "2010-11-02T03:32:15-05:00",
  "clientId": "ab4820dhcb39347"
```

```
}
```

4.4.31. Delete a Delegated Refresh Token for a User

This operation requires an authorization header. See Section 3.6, "Authorization Header" for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/customers/*customerId*/users/*userId*/ delegatedrefreshtokens/*tokenId*	Deletes a user's delegated refresh tokens

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.5. Authentication Operations

4.5.1. Authenticate a user by Mosso ID

This operation requires an authorization header. See Section 3.6, "Authorization Header" for details.

Verb	URI	Description
POST	/auth/mosso	Authenticate a user by mosso Id. Pass in the mosso credentials shown below.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.87. XML Auth By Mosso Credentials Request

```
<?xml version="1.0" encoding="UTF-8"?>

<mossoCredentials xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    mossoId="999112358"
    key="qauser-000-001" />
```

Example 4.88. JSON Auth By Mosso Credentials Request

```
{
    "mossoId": "999112358"
```

```
    "key": "qauser-000-001"
}
```


Example 4.89. XML Auth By Mosso Credentials Response


```
<?xml version="1.0" encoding="UTF-8"?>

<cloudAuth xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <token id="DFW-c6dfc4456f6545f8b8265ff5b6bda775"
        expires_in="16640"/>
 <serviceCatalog xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
     <service name="cloudFiles">
         <endpoint region="DFW"
             v1Default="true"
             publicURL="https://storage.clouddrive.com/v1/KovacsNastId"
             internalURL="https://storage-snet.clouddrive.com/v1/
KovacsNastId"/>
    </service>
    <service name="cloudServers">
         <endpoint v1Default="false"
             publicURL="https://servers.api.rackspacecloud.com/v1.0/90000"/>
    </service>
 </serviceCatalog>
</cloudAuth>
```


Example 4.90. JSON Auth By Mosso Credentials Response


```
{"token": {
   "id": "DFW-c6dfc4456f6545f8b8265ff5b6bda775",
   "expires_in": 16110
}}
```


4.5.2. Authenticate a user by NAST ID

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
POST	/auth/nast	Authenticate a user by nast Id. Pass in the nast credentials shown below.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.91. XML Auth By Nast Credentials Request


```
<?xml version="1.0" encoding="UTF-8"?>

<nastCredentials xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    nastId="nastid-qauser"
```

```
    key="qauser-000-001" />
```


Example 4.92. JSON Auth By Nast Credentials Request


```
{
    "nastId": "nastid-qauser"
    "key": "qauser-000-001"
}
```


Example 4.93. XML Auth By Nast Credentials Response


```
<?xml version="1.0" encoding="UTF-8"?>

<cloudAuth xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <token id="DFW-c6dfc4456f6545f8b8265ff5b6bda775"
        expires_in="16640"/>
 <serviceCatalog xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
     <service name="cloudFiles">
         <endpoint region="DFW"
             v1Default="true"
             publicURL="https://storage.clouddrive.com/v1/KovacsNastId"
             internalURL="https://storage-snet.clouddrive.com/v1/
KovacsNastId"/>
    </service>
    <service name="cloudServers">
         <endpoint v1Default="false"
             publicURL="https://servers.api.rackspacecloud.com/v1.0/90000"/>
    </service>
 </serviceCatalog>
</cloudAuth>
```


Example 4.94. JSON Auth By Nast Credentials Response


```
{"token": {
   "id": "DFW-c6dfc4456f6545f8b8265ff5b6bda775",
   "expires_in": 16110
}}
```


4.5.3. Authenticate a user by Username

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
POST	/auth	Authenticate a user by username. Pass in the username credentials shown below.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.95. XML Auth By Username Request

```
<?xml version="1.0" encoding="UTF-8"?>

<usernameCredentials xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    username="qauser"
    key="qauser-000-001" />
```

Example 4.96. XML Auth By Username Response

```
<?xml version="1.0" encoding="UTF-8"?>

<cloudAuth xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <token id="DFW-c6dfc4456f6545f8b8265ff5b6bda775"
        expires_in="16640"/>
 <serviceCatalog xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
     <service name="cloudFiles">
         <endpoint region="DFW"
             v1Default="true"
             publicURL="https://storage.clouddrive.com/v1/KovacsNastId"
             internalURL="https://storage-snet.clouddrive.com/v1/
KovacsNastId"/>
    </service>
    <service name="cloudServers">
         <endpoint v1Default="false"
             publicURL="https://servers.api.rackspacecloud.com/v1.0/90000"/>
    </service>
 </serviceCatalog>
</cloudAuth>
```

Example 4.97. JSON Auth By Username Response

```
{"token": {
   "id": "DFW-c6dfc4456f6545f8b8265ff5b6bda775",
   "expires_in": 16110
}}
```

4.6. Application Operations

4.6.1. Add an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
POST	/customers/*customerId*/clients	Add a client.

Normal Response Code(s):201

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.98. XML Add Client Request

```
<?xml version="1.0" encoding="UTF-8"?>

<client xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    name="Test Application2"
    customerId="RCN-000-000-000" />
```

Example 4.99. JSON Add Client Request

```
{
  "customerId": "RCN-000-000-000",
  "name": "Test Application2"
}
```

Example 4.100. XML Add Client Response

```
<?xml version="1.0" encoding="UTF-8"?>

<client xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="ab4820dhcb39347">
    <credentials clientSecret="3af738fbeiwu23" />
</client>
```

Example 4.101. JSON Add Client Response

```
{
  "credentials": {
    "clientSecret": "3af738fbeiwu23"
  },
  "clientId": "ab4820dhcb39347",
  "customerId": "RCN-000-000-000",
  "name": "Test Application2",
  "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001",
  "status": "ACTIVE",
  "locked": false,
  "softDeleted": false
}
```

4.6.2. Get an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
GET	/customers/*customerId*/clients/*clientId*	Get a client.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.102. XML Get Client Response

```
<?xml version="1.0" encoding="UTF-8"?>

<client xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    iname="@Rackspace*Rackspace*ControlPanel"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="ab4820dhcb39347" />
```

Example 4.103. JSON Get Client Response

```
{
  "clientId": "ab4820dhcb39347",
  "customerId": "RCN-000-000-000",
  "name": "Test Application2",
  "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001",
  "status": "ACTIVE",
  "locked": false,
  "softDeleted": false
}
```

4.6.3. Get All Applications for a Customer

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
GET	/customers/*customerId*/clients/	Get all clients for *customerId*

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.104. XML Get Clients Response

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<clients xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <client
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="ab4820dhcb39347">
 </client>
 <client
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0002"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="632h389cv902bde">
 </client>
</clients>
```

Example 4.105. JSON Get Clients Response

```
{
  "client": [
    {
      "clientId": "ab4820dhcb39347",
      "customerId": "RCN-000-000-000",
      "name": "Test Application2",
      "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001",
      "status": "ACTIVE",
      "locked": false,
      "softDeleted": false
    },
    {
      "clientId": "632h389cv902bde",
      "customerId": "RCN-000-000-000",
      "name": "Test Application2",
      "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0002",
      "status": "ACTIVE",
      "locked": false,
      "softDeleted": false
    }
  ]
}
```

4.6.4. Update an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
PUT	/customers/*customerId*/clients/*clientId*	Update a client.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden(403), itemNotFound (404), emailConflict (409), idmFault (500), serviceUnavailable(503)

Example 4.106. XML Update Client Request

```
<?xml version="1.0" encoding="UTF-8"?>

<client xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    scope="cloud_servers"
    clientId="ab4820dhcb39347" />
```

Example 4.107. JSON Update Client Request

```
{
  "clientId": "ab4820dhcb39347",
  "scope": "cloud_servers"
}
```

Example 4.108. XML Update Client Response

```
<?xml version="1.0" encoding="UTF-8"?>

<client xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    iname="@Rackspace*Rackspace*ControlPanel"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="ab4820dhcb39347"
    scope="cloud_servers" />
```

Example 4.109. JSON Update Client Response

```
{
  "clientId": "ab4820dhcb39347",
  "customerId": "RCN-000-000-000",
  "name": "Test Application2",
  "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001",
  "scope": "cloud_servers",
  "status": "ACTIVE",
  "locked": false,
  "softDeleted": false
}
```

4.6.5. Soft Delete an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
DELETE	/customers/*customerId*/clients/*clientId*/softdeleted	Delete a customer application.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.110. XML Soft Delete Application Request

```
<?xml version="1.0" encoding="UTF-8"?>

<client xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="true" />
```

Example 4.111. JSON Soft Delete Application Request

```
{
  "softDeleted": "true"
}
```

Example 4.112. XML Soft Delete Application Response

```
<?xml version="1.0" encoding="UTF-8"?>

<client xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    softDeleted="true" />
```

Example 4.113. JSON Soft Delete Application Response

```
{
  "softDeleted": "true"
}
```

4.6.6. Delete an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
DELETE	/customers/*customerId*/clients/*clientId*	Delete a client.

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.6.7. Get Permissions defined by an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
GET	/customers/*customerId*/clients/*clientId*/permissions	Get all permissions defined by *clientId* of a *customerId*

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.114. XML Get Permissions Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<permissions xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
   <permission
       clientId="IDM"
       customerId="RCN-000-000-000"
       permissionId="addCustomer"/>
   <permission
       clientId="IDM"
       customerId="RCN-000-000-000"
       permissionId="getCustomer"/>
</permissions>
```

Example 4.115. JSON Get Permissions Response

```
{ "permission": [
   {
     "permissionId": "addCustomer",
     "customerId": "RCN-000-000-000",
     "clientId": "IDM"
   },
   {
     "permissionId": "getCustomer",
     "customerId": "RCN-000-000-000",
     "clientId": "IDM"
   }]
}
```

4.6.8. Add a Permission defined by an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
POST	/customers/*customerId*/clients/*clientId*/permissions	Define a permission for a *clientId* of a *customerId*

Normal Response Code(s):201

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.116. XML Add a Defined Permission Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<permission xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    type="application/text"
    clientId="IDM"
    customerId="RCN-000-000-000"
    permissionId="addCustomer">POST /customers</permission>
```

Example 4.117. JSON Add a Defined Permission Request

```
{
  "value": "POST /customers",
  "permissionId": "addCustomer",
  "customerId": "RCN-000-000-000",
  "clientId": "IDM",
  "type": "application/text"
}
```

Example 4.118. XML Add a Defined Permission Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<permission xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    type="application/text"
    clientId="IDM"
    customerId="RCN-000-000-000"
    permissionId="addCustomer">POST /customers</permission>
```

Example 4.119. JSON Add a Defined Permission Response

```
{
  "value": "POST /customers",
  "permissionId": "addCustomer",
  "customerId": "RCN-000-000-000",
  "clientId": "IDM",
  "type": "application/text"
}
```

4.6.9. Get a Permission defined by an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/clients/*clientId*/permissions/defined/*permissionId*	Get a *permissionId* for a *clientId* of a *customerId*

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.120. XML Get Defined Permission Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<permission xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    type="application/text"
    clientId="IDM"
    customerId="RCN-000-000-000"
    permissionId="addCustomer">POST /customers</permission>
```

Example 4.121. JSON Get Defined Permission Response

```
{
  "value": "POST /customers",
  "permissionId": "addCustomer",
  "customerId": "RCN-000-000-000",
  "clientId": "IDM",
  "type": "application/text"
}
```

4.6.10. Update a Permission defined by an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
PUT	/customers/*customerId*/clients/*clientId*/permissions/defined/*permissionId*	Update a permission defined by a *clientId* of a *customerId*

Normal Response Code(s):201

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.122. XML Update a Defined Permission Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<permission xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    type="application/text"
    clientId="IDM"
    customerId="RCN-000-000-000"
    permissionId="addCustomer">POST /customers</permission>
```

Example 4.123. JSON Update a Defined Permission Request

```
{
  "value": "POST /customers",
  "permissionId": "addCustomer",
  "customerId": "RCN-000-000-000",
  "clientId": "IDM",
  "type": "application/text"
}
```

Example 4.124. XML Update a Defined Permission Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<permission xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    type="application/text"
    clientId="IDM"
    customerId="RCN-000-000-000"
    permissionId="addCustomer">POST /customers</permission>
```

Example 4.125. JSON Update a Defined Permission Response

```
{
  "value": "POST /customers",
  "permissionId": "addCustomer",
  "customerId": "RCN-000-000-000",
  "clientId": "IDM",
  "type": "application/text"
```

```
}
```

4.6.11. Delete a Permission defined by an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/customers/*customerId*/clients/*clientId*/permissions/defined/*permissionId*	Delete a permission defined by a *clientId* of a *customerId*

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.6.12. Reset an Application Secret

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
POST	/customers/*customerId*/clients/*clientId*/secret	Reset client secret.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.126. XML Reset Client Secret Response

```
<?xml version="1.0" encoding="UTF-8"?>

<clientCredentials xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
 clientSecret="c582fe94dda4c0da19bf234c504db0b63df96daa" />
```

Example 4.127. JSON Reset Client Secret Response

```
{
  "clientSecret" : "eca2d413a1790dc6473842ab5a49387a29996757"
}
```

4.6.13. Add a Group for an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
POST	/customers/*customerId*/clients/*clientId*/groups	Add a group for a `clientId` of a `customerId`

Normal Response Code(s):201

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.128. XML Add a Group Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<clientGroup xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    customerId="RCN-000-000-000"
    clientId="IDM"
    name="Idm Admin"
    type="Role"  />
```

Example 4.129. JSON Add a Group Request

```
{
    "customerId" : "RCN-000-000-000",
    "clientId" : "IDM",
    "name" : "Idm Admin",
    "type" : "Role"
}
```

Example 4.130. XML Add a Group Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<clientGroup xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    customerId="RCN-000-000-000"
    clientId="IDM"
    name="Idm Admin"
    type="Role"  />
```

Example 4.131. JSON Add a Group Response

```
{
```

```
    "customerId" : "RCN-000-000-000",
    "clientId" : "IDM",
    "name" : "Idm Admin",
    "type" : "Role"
}
```

4.6.14. Get all Groups for an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/clients/*clientId*/groups	Get all groups for a `clientId` of a `customerId`

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.132. XML Get Groups Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<clientGroups xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <clientGroup customerId="RCN-000-000-000"
        clientId="IDM"
        name="Idm Admin"
        type="Role"  />
</clientGroups>
```

Example 4.133. JSON Get Groups Response

```
{"clientGroup": [
    {
        "customerId" : "RCN-000-000-000",
        "clientId" : "IDM",
        "name" : "Idm Admin",
        "type" : "Role"
    }]
}
```

4.6.15. Update a Group for an Application

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
PUT	/customers/*customerId*/clients/*clientId*/groups/*groupName*	Update a group for a `clientId` of a `customerId`

Normal Response Code(s):201

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.134. XML Update a Group Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<clientGroup xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    customerId="RCN-000-000-000"
    clientId="IDM"
    name="Idm Admin"
    type="Role"  />
```

Example 4.135. JSON Update a Group Request

```
{
    "customerId" : "RCN-000-000-000",
    "clientId" : "IDM",
    "name" : "Idm Admin",
    "type" : "Role"
}
```

Example 4.136. XML Update a Group Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<clientGroup xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    customerId="RCN-000-000-000"
    clientId="IDM"
    name="Idm Admin"
    type="Role"  />
```

Example 4.137. JSON Update a Group Response

```
{
    "customerId" : "RCN-000-000-000",
    "clientId" : "IDM",
    "name" : "Idm Admin",
    "type" : "Role"
}
```

4.6.16. Delete a Group for an Application

This operation requires an authorization header. See Section 3.6, "Authorization Header" for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/customers/*customerId*/clients/*clientId*/groups/*groupName*	Delete a group for a *clientId* of a *customerId*

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.6.17. Add a User to a Group for an Application

This operation requires an authorization header. See Section 3.6, "Authorization Header" for details.

Note

This is a management API function.

Verb	URI	Description
PUT	/customers/*customerId*/clients/*clientId*/groups/*groupName*/members/*username*	Add a user to a group for a *clientId* of a *customerId*

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.6.18. Remove a User from a Group for an Application

This operation requires an authorization header. See Section 3.6, "Authorization Header" for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/customers/*customerId*/clients/*clientId*/groups/*groupName*/mbmers/*username*	Remove a user from a group for a *clientId* of a *customerId*

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.6.19. Get the Services that a Client has Provisioned

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/clients/*clientId*/services	Get the services that a user has provisioned

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

Example 4.138. XML Get Cilent Services Response


```xml
<?xml version="1.0" encoding="UTF-8"?>

<clients xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
 <client
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="ab4820dhcb39347">
 </client>
 <client
    softDeleted="false"
    locked="false"
    status="ACTIVE"
    inum="@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0002"
    name="Test Application2"
    customerId="RCN-000-000-000"
    clientId="632h389cv902bde">
 </client>
</clients>
```


Example 4.139. JSON Get Client Services Response

```
{
  "client": [
    {
      "clientId": "ab4820dhcb39347",
      "customerId": "RCN-000-000-000",
      "name": "Test Application2",
      "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0001",
      "status": "ACTIVE",
      "locked": false,
      "softDeleted": false
    },
    {
      "clientId": "632h389cv902bde",
      "customerId": "RCN-000-000-000",
      "name": "Test Application2",
      "inum": "@FFFF.FFFF.FFFF.FFFF!EEEE.EEEE!0002",
      "status": "ACTIVE",
      "locked": false,
      "softDeleted": false
    }
  ]
}
```

4.6.20. Grant a Client Permission for Service

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
POST	/customers/*customerId*/clients/*clientId*/services/*serviceId*/permissions	Grant a client permission for a service

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.140. XML Grant Client Permission Request

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<permission xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    clientId="IDM"
    customerId="RCN-000-000-000"
    permissionId="addCustomer" />
```

Example 4.141. JSON Grant Client Permission Request

```
{
```

```
  "permissionId": "addCustomer",
  "customerId": "RCN-000-000-000",
  "clientId": "IDM"
}
```

4.6.21. Revoke a Client Permission for Service

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/customers/*customerId*/cilents/*clientId*/services/*serviceId*/permissions/*permissionId*	Check to see if client has permisison for a service

Normal Response Code(s):204

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.6.22. Check if Client has Permission for Service

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/customers/*customerId*/clients/*clientId*/services/*serviceId*/permissions/*permissionId*	Revoke a client permission for a service

Normal Response Code(s):200 404

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503)

This operation does not require a request body.

4.7. Password Operations

4.7.1. Get Password Rules

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
GET	/passwordrules	Get Password Rules

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503)

Example 4.142. XML Get Password Rules Response

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>

<passwordRuleResults
    xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <passwordRuleResults
        ruleMessage="Password must be at least 7 characters."
        ruleName="Minimum Length"
        ruleId="1"
        passed="true" />
    <passwordRuleResults
        ruleMessage="Password must contain a lowercase."
        ruleName="Lowercase Character"
        ruleId="2"
        passed="true" />
</passwordRuleResults>
```

Example 4.143. JSON Get Password Rules Response

```
{
  "passwordRuleResult": [
    {
      "passed": true,
      "ruleId": 1,
      "ruleName": "Minimum Length",
      "ruleMessage": "Password must be at least 7 characters long."
    },
    {
      "passed": true,
      "ruleId": 2,
      "ruleName": "Lowercase Character",
      "ruleMessage": "Password must contain a lowercase character."
    }
  ]
}
```

4.7.2. Password validation check

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Verb	URI	Description
POST	/passwordrules/validation	Check Password Validation for `password`

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), itemNotFound (404), idmFault (500), serviceUnavailable(503)

Example 4.144. XML Password Validation Request


```
<?xml version="1.0" encoding="UTF-8"?>

<userPassword xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    password="newpassword" />
```


Example 4.145. JSON Password Validation Request


```
{
  "password": "newpassword"
}
```


Example 4.146. XML Password Validation Response


```
<?xml version="1.0" encoding="UTF-8"?>

<passwordValidation xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    validPassword="true">
    <passwordRuleResults>
        <passwordRuleResults
          ruleMessage="Password must be at least 7 characters long."
          ruleName="Minimum Length"
          ruleId="1"
          passed="true" />
        <passwordRuleResults
          ruleMessage="Password must contain a lowercase character."
          ruleName="Lowercase Character"
          ruleId="2"
          passed="true" />
    </passwordRuleResults>
</passwordValidation>
```


Example 4.147. JSON Password Validation Response


```
{"passwordRuleResults":{
    "passwordRuleResults":[
        {
            "passed":true,
            "ruleId":1,"ruleName":
            "Mininumn Length",
            "ruleMessage":"The password must be at least 7 characters long"
        },
        {
            "passed":false,
            "ruleId":2,"ruleName":
            "Uppercase Rule","ruleMessage":
            "The password must contain an uppercase charater"
        }
```

```
    ]},
    "validPassword":false
}
```


4.8. Admin Operations

4.8.1. Get Base URLs

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/baseurls	Get base urls

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

Example 4.148. XML Get BaseURLs Response


```
<?xml version="1.0" encoding="UTF-8"?>

<baseURLs xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <baseURL
        id="111111"
        userType="NAST"
        serviceName="cloudFiles"
        region="DFW"
        publicURL="https://storage.clouddrive.com/v1"
        internalURL="https://storage-snet.clouddrive.com/v1"
        default="true"/>
    <baseURL
        id="222222"
        userType="MOSSO"
        serviceName="cloudServers"
        publicURL="https://servers.api.rackspacecloud.com/v1.0"
        default="true"/>
</baseURLs>
```


Example 4.149. JSON Get BaseURLs Response


```
{"baseURL": [
      {
      "id": 111111,
      "userType": "NAST",
      "serviceName": "cloudFiles",
      "region": "DFW",
```

```
      "publicURL": "https://storage.clouddrive.com/v1",
      "internalURL": "https://storage-snet.clouddrive.com/v1",
      "adminURL": null,
      "default": true
   },
      {
      "id": 222222,
      "userType": "MOSSO",
      "serviceName": "cloudServers",
      "region": null,
      "publicURL": "https://servers.api.rackspacecloud.com/v1.0",
      "internalURL": null,
      "adminURL": null,
      "default": true
   }
]}
```

4.8.2. Add Base URL

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
POST	/baseurls	Add base urls. Pass the base url data (shown below).

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

Example 4.150. XML Add BaseUrl Request

```
<?xml version="1.0" encoding="UTF-8"?>

<baseURL xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    id="111114"
    userType="NAST"
    serviceName="cloudFiles"
    region="DFW"
    publicURL="https://storage.clouddrive.com/v1"
    internalURL="https://storage-snet.clouddrive.com/v1"
    adminURL="https://storage-snet.clouddrive.com/v1"
    default="true"/>
```

4.8.3. Get a Base URL

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/baseurls/*baseUrlId*	Get a base url

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

Example 4.151. XML Get BaseUrl Response

```
<?xml version="1.0" encoding="UTF-8"?>

<Response xmlns="http://10.127.7.164/v1.0/baseurls/111111"
    adminURL="https://storage-snet.clouddrive.com/v1"
    default="true"
    id="111111"
    internalURL="https://storage-snet.clouddrive.com/v1"
    publicURL="https://storage.clouddrive.com/v1"
    region="DFW"
    serviceName="cloudFiles"
    userType="NAST"/>
```

Example 4.152. JSON Get BaseUrl Response

```
{
   "id": 111111,
   "userType": "NAST",
   "serviceName": "cloudFiles",
   "region": "DFW",
   "publicURL": "https://storage.clouddrive.com/v1",
   "internalURL": "https://storage-snet.clouddrive.com/v1",
   "adminURL": null,
   "default": true
}
```

4.8.4. Delete a Base URL

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/baseurls/*baseUrlId*	Delete a base url

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

4.8.5. Get Base Url Refs

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/users/*username*/baseurlrefs	Gets a base url reference

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

Example 4.153. XML Get Base Url Ref Response

```
<?xml version="1.0" encoding="UTF-8"?>

<baseURLRefs xmlns="http://docs.rackspacecloud.com/auth/api/v1.1">
 <baseURLRef
        href="https://auth.api.rackspacecloud.com/v1.1/baseURLs/1"
        id="1"
        v1Default="true" />
 <baseURLRef
        href="https://auth.api.rackspacecloud.com/v1.1/baseURLs/2"
        id="2" />
 <baseURLRef
        href="https://auth.api.rackspacecloud.com/v1.1/baseURLs/3"
        id="3"
        v1Default="true" />
 <baseURLRef
        href="https://auth.api.rackspacecloud.com/v1.1/baseURLs/4"
        id="4" />
 <baseURLRef
        href="https://auth.api.rackspacecloud.com/v1.1/baseURLs/5"
        id="5"
        v1Default="true" />
</baseURLRefs>
```

Example 4.154. JSON Get Base Url Ref Response

```
{
 "baseURLRefs" : [
  {
  "id" : 1,
  "href" : "https://auth.api.rackspacecloud.com/v1.1/baseURLs/1",
  "v1Default" : true
```

```
  },
  {
  "id" : 2,
  "href" : "https://auth.api.rackspacecloud.com/v1.1/baseURLs/2"
  },
  {
  "id" : 3,
  "href" : "https://auth.api.rackspacecloud.com/v1.1/baseURLs/3",
  "v1Default" : true
  },
  {
  "id" : 4,
  "href" : "https://auth.api.rackspacecloud.com/v1.1/baseURLs/4"
  },
  {
  "id" : 5,
  "href" : "https://auth.api.rackspacecloud.com/v1.1/baseURLs/5",
  "v1Default" : true
  }
 ]
}
```

4.8.6. Add Base URL Ref

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
PUT	/users/*username*/baseurlrefs	Add base url Ref. Pass the base url ref data (shown below).

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

Example 4.155. XML Add Base Url Ref Request

```
<?xml version="1.0" encoding="UTF-8"?>

<baseURLRef xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    id="111114"
    href="https://storage-snet.clouddrive.com/v1"
    v1Default="true"/>
```

4.8.7. Get Base URL Ref

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/users/*username*/baseurlrefs/*baseUrlId*	Get a base url.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

Example 4.156. XML Get Base Url Ref Response


```xml
<?xml version="1.0" encoding="UTF-8"?>

<baseURLRef xmlns="http://docs.rackspacecloud.com/idm/api/v1.0"
    id="111111"
    href="https://idm.rackspace.com/v1.0/baseurls/111111"
    v1Default="true" />
```


Example 4.157. JSON Get Base Url Ref Response


```json
{
   "id": 111111,
   "href": "https://idm.rackspace.com/v1.0/baseurls/111111",
   "v1Default": true
}
```


4.8.8. Delete a Base URL Ref

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
DELETE	/users/*username*/baseurls/*baseUrlId*	Delete a base url ref

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

4.8.9. Get Service catalog for user

This operation requires an authorization header. See Section 3.6, “Authorization Header” for details.

Note

This is a management API function.

Verb	URI	Description
GET	/users/*username*/servicecatalog	Get Service Catalog for users.

Normal Response Code(s):200

Error Response Code(s): unauthorized (401), badRequest (400), forbidden (403), idmFault (500), serviceUnavailable(503), itemNotFound (404)

Example 4.158. XML Get Service Catalog Response

```
<?xml version="1.0" encoding="UTF-8"?>

<serviceCatalog xmlns="http://docs.rackspacecloud.com/idm/api/v1.0">
    <service name="cloudFiles">
        <endpoint region="DFW"
            v1Default="true"
            publicURL="https://storage.clouddrive.com/v1/KovacsNastId"
            internalURL="https://storage-snet.clouddrive.com/v1/KovacsNastId"/
>
   </service>
   <service name="cloudServers">
        <endpoint v1Default="false"
            publicURL="https://servers.api.rackspacecloud.com/v1.0/90000"/>
   </service>
</serviceCatalog>
```

Example 4.159. JSON Get Service Catalog Response

```
{"service": [
      {
      "endpoint": [      {
         "region": "DFW",
         "v1Default": true,
         "publicURL": "https://storage.clouddrive.com/v1/KovacsNastId",
         "internalURL": "https://storage-snet.clouddrive.com/v1/KovacsNastId",
         "adminURL": null
      }],
      "name": "cloudFiles"
   },
      {
      "endpoint": [      {
         "region": null,
         "v1Default": false,
         "publicURL": "https://servers.api.rackspacecloud.com/v1.0/90000",
         "internalURL": null,
         "adminURL": null
      }],
      "name": "cloudServers"
   }
]}
```