Getting Started with API - Authentication and Session Creation

John Fitzpatrick

Table of Contents

Lab Overview and Requirements

Overview

This lab introduces the RightScale API by way of some simple examples. The tasks you will perform include

- Authenticating
- Listing Deployments
- Listing Servers within a Deployment

Lab Environment

You will launch an 'API Sender' server in AWS EC2 cloud, and SSH into to from your desktop. All API commands will then be issued from this server.

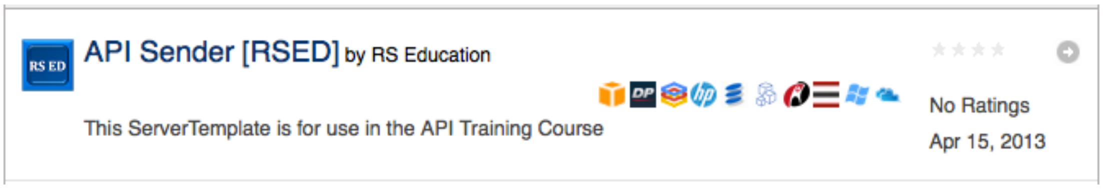
Requirements

- You need a currently supported Browser. See http://tinyurl.com/cl8p4mh
- Java must be enabled in the browser.
- Unrestricted Internet Access. In particular, you must be able to browse to rightscale.com on TCP port 80 (HTTP).
- You must have TCP port 22 (SSH) open so you can SSH into remote servers
- RightScale account with valid cloud credentials, and the following user role privileges - **'designer'**, **'actor'**, **'library'**, '**server_login**', '**security_manager**'
- You must also be familiar with `vi` editor

Section 1. API Environment Setup

1.1 Create an API Sender Cloud Server

1. Navigate to the MultiCloud Marketplace (**Design** > **MultiCloud Marketplace** > **ServerTemplates**)
2. Import the ServerTemplate "**API Sender [RSED]**" published by "RightScale Education"



API Sender [RSED] by RS Education

This ServerTemplate is for use in the API Training Course

No Ratings

Apr 15, 2013

3. Click 'Add Server' to add a server into the deployment using the latest revision of the imported '**API Sender [RSED]**' ServerTemplate.

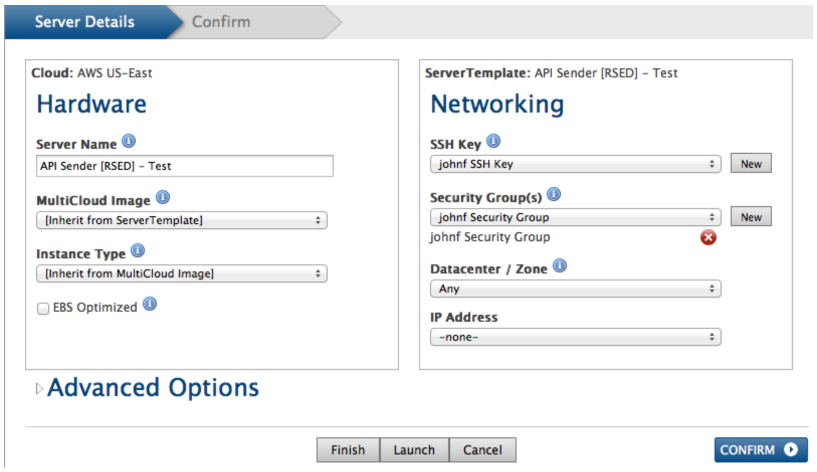
   Cloud – Any AWS EC2 Region

   Deployment – "**API Senders Deployment**"

   Server Name - "**myname API Sender**"

   SSH Key - Click 'New' and give your SSH Key a name.

   Security Group- Click 'New' and give your Security Group a name. You can accept default ports '22' and '80'.



Cloud: AWS US-East

Hardware

Server Name: API Sender [RSED] - Test

MultiCloud Image: [Inherit from ServerTemplate]

Instance Type: [Inherit from MultiCloud Image]

EBS Optimized

ServerTemplate: API Sender [RSED] - Test

Networking

SSH Key: johnf SSH Key

Security Group(s): johnf Security Group

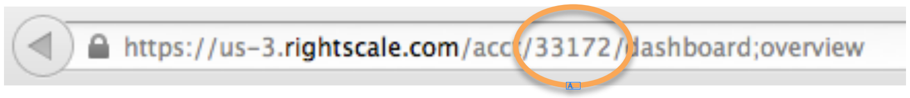
johnf Security Group

Datacenter / Zone: Any

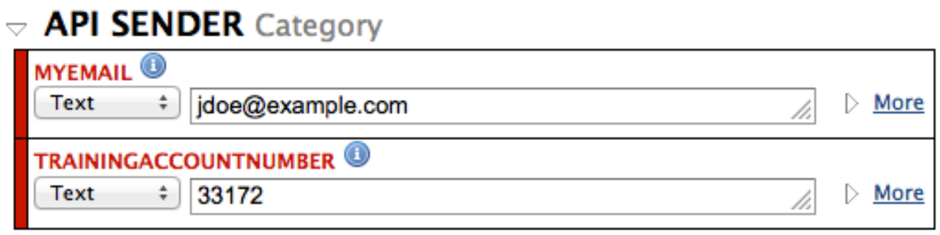
IP Address: -none-

4. Launch the server
5. Populate the input `MYEMAIL` with your email you use to access the RightScale Dashboard, and `TRAININGACCOUNTNUMBER` with the Account Number of the RightScale account used for the API Training

   You can get the `TRAININGACCOUNTNUMBER` from the Dashboard URL, i.e.



https://us-3.rightscale.com/acct/33172/dashboard;overview



API SENDER Category

MYEMAIL	Text	jdoe@example.com
TRAININGACCOUNTNUMBER	Text	33172

Alternatively the instructor can tell you what number to use.

6. Click **Save and Launch**.

Note: During this class you will be authenticating with RightScale both at the command line, as well as from bash scripts, so you might wish to change your RightScale password for the purposes of this course. You can change your password by navigating to '**Settings > User Settings > Authentication**'.

Section 2. Session Management

See: http://reference.rightscale.com/api1.5/resources/ResourceSessions.html

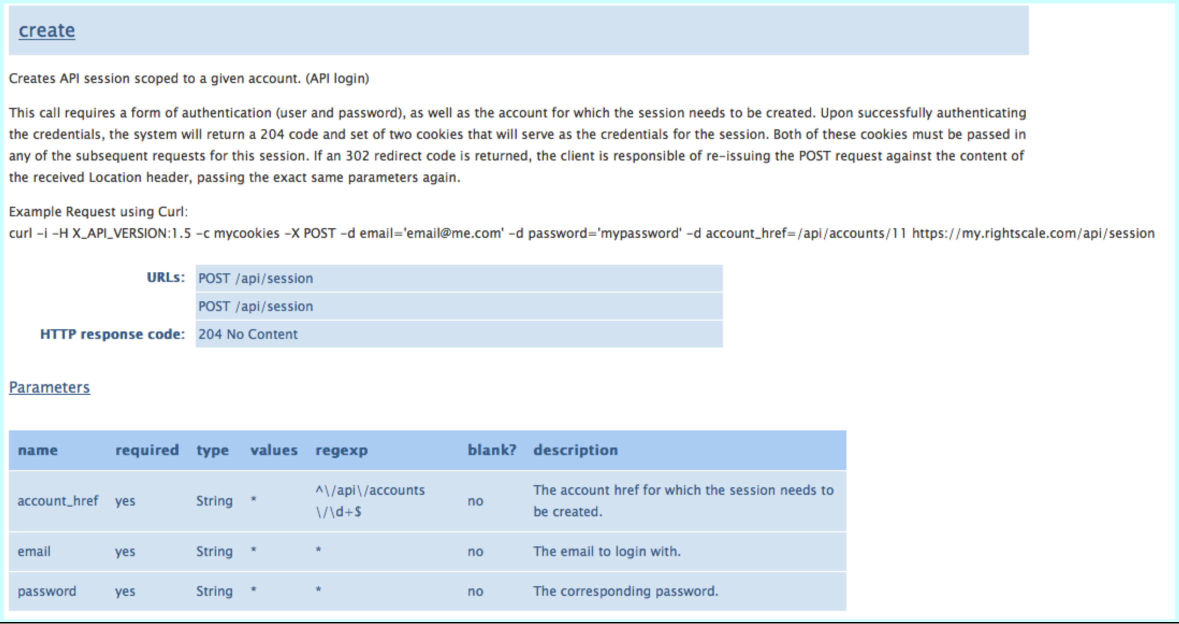
In this lab you will authenticate with RightScale API to create a new session using two methods

- Passing **Username** and **Password**
- Using an **OAuth**

Note: Please note that the commands in this lab can be found in the following file on your API Sender server:- `/opt/api/session_mgt/GettingStartedLab-Commands`. You may find it easier to copy/paste the commands from that file rather than typing them.

2.1 Authenticating using CURL - Creating a Session

See http://reference.rightscale.com/api1.5/resources/ResourceSessions.html#create



create

Creates API session scoped to a given account. (API login)

This call requires a form of authentication (user and password), as well as the account for which the session needs to be created. Upon successfully authenticating the credentials, the system will return a 204 code and set of two cookies that will serve as the credentials for the session. Both of these cookies must be passed in any of the subsequent requests for this session. If an 302 redirect code is returned, the client is responsible of re-issuing the POST request against the content of the received Location header, passing the exact same parameters again.

Example Request using Curl:

```
curl -i -H X_API_VERSION:1.5 -c mycookies -X POST -d email='email@me.com' -d password='mypassword' -d account_href=/api/accounts/11 https://my.rightscale.com/api/session
```

URLs:	POST /api/session
	POST /api/session
HTTP response code:	204 No Content

Parameters

name	required	type	values	regexp	blank?	description
account_href	yes	String	*	^\/api\/accounts\/\d+$	no	The account href for which the session needs to be created.
email	yes	String	*	*	no	The email to login with.
password	yes	String	*	*	no	The corresponding password.

In this section you will issue an API `HTTP POST` command to the resource `/api/session` to create a session. You should get a `204 No Content` response.

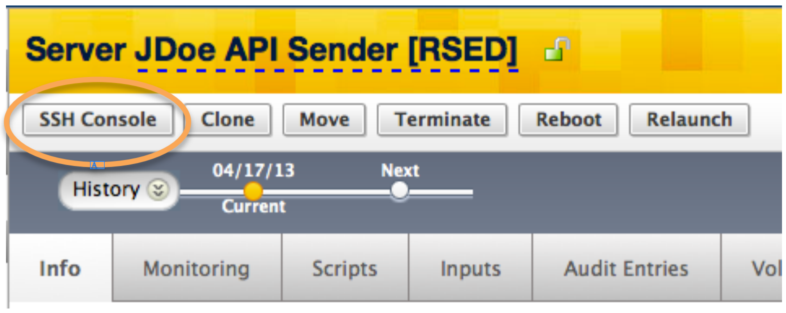
As you can see in the API Reference documentation, the format of the API request is

```
curl -i -H X-API-Version:1.5 -c mycookie -X POST -d email='email@me.com' -
d password='mypassword' -d account_href=/api/accounts/33172
https://my.rightscale.com/api/session
```

The `account_href`, `email` & `password` are mandatory. The `email` & `password` are your own username and password used to log into the RightScale Dashboard. The ID number in the `account_href` is the number of the RightScale account you're using for the training (i.e. the value used for `TRAININGACCOUNTNUMBER` when launching your server)

More details on using `curl` can be found here http://curl.haxx.se/docs/manpage.html.

1. Navigate to your server '**myname API Sender**' and click on the **SSH Console** button to SSH into the Server



2. Execute the following command, substituting
   - Your own RightScale username email and password
   - The ID of the account being used during the training (i.e. used in the input `TRAININGACCOUNTNUMBER` when launching your server)

```
[api]# curl -i -H X-API-Version:1.5 -c mycookie -X POST -d
email=jdoe@example.com -d password=mypassword -d
account_href=/api/accounts/33172 https://us-3.rightscale.com/api/session
```

The name of the cookie (`mycookie` in the above example) is arbitrary, but since that is the name of your authentication cookie, you must use that filename for every request.

Note: In this section of the course you will be entering your username and password at the command line. This may be perceived as a security risk, so you can clear your command history with the command `history —c`, if you are not comfortable entering your account password at the command line.

Later in the course you will be using bash scripts, but the password will be read in at command line, so nowhere is your password written to disk. However you may wish to change your password in any case.

You should receive the output similar to the following:-

```
HTTP/1.1 204 No Content
Server: nginx/1.0.14
Date: Wed, 27 Feb 2013 10:50:51 GMT
Connection: keep-alive
Status: 204 No Content
HTTP/1.1 204 No Content
Server: nginx/1.0.15
Date: Wed, 27 Feb 2013 11:37:27 GMT
Connection: keep-alive
Status: 204 No Content
X-Runtime: 2022
X-Request-Uuid: e3e9cfac3c3a4bc3a012f4fe5b6ea528
Set-Cookie:
rs_gbl=eNotkMtugkAUQP_lrp1kHkCBpJtKVIQaH7URNg1yR1EE6zCIo_Hfi0n355zFeUAGPlQ
GBoAN-A9oG6nA55ZliecAdA4-Ew7zHJta9gAO2NNUSkZdQYlLk-Tp2MvubXMI7j8ZUfZ-
PLSt2iTaU_EOmMpfa8_jZFt2ZkPyK7IhApxvQwTYvyjhtqjrF3vwbtNjDKSraibNVkSsJQn5bK
G6qLjLokqyfdcL9tKm896bApLZ0o-v5acnstyfL83NYafNvtNzyffwgjX0s%3D;
domain=.rightscale.com; path=/; HttpOnly
Set-Cookie: _session_id=deadbeef0badf00dfeedfacec0ffee24; path=/; HttpOnly
Cache-Control: no-cache
```

Note the following line in the HTTP Response Code

```
Status: 204 No Content
```

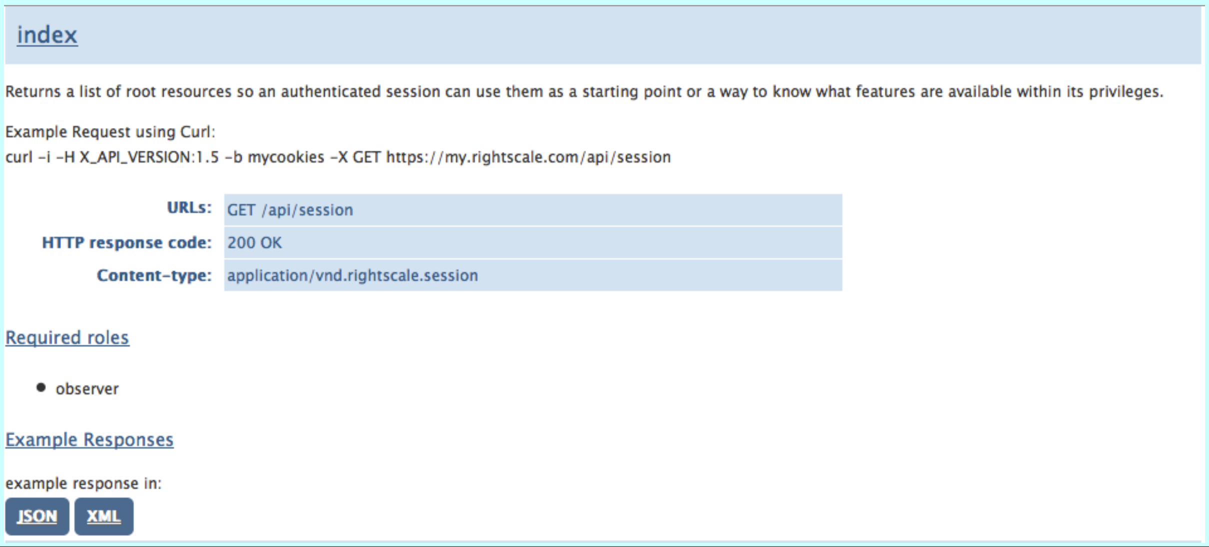
3. Now check your cookie is in place, and view the cookie contents

```
[api]$ ls -l
total 4
-rw-rw-r-- 1 myuser myuser 635 Feb 27 11:16 mycookie
[api]$ cat mycookie
# Netscape HTTP Cookie File
# http://curl.haxx.se/rfc/cookie_spec.html
# This file was generated by libcurl! Edit at your own risk.

#HttpOnly_.rightscale.com     TRUE   /      FALSE 0      rs_gbl
      eNotkMtugkAUQP_lrp1kHkCBpJtKVIQaH7URNg1yR1EE6zCIo_Hfi0n355zFeUAGPlQG
BoAN-A9oG6nA55ZliecAdA4-Ew7
zHJta9gAO2NNUSkZdQYlLkRPGJCceCiTosDzHzJFC0L6n5b_7xvnL7fOQUlPZ0sQ6KXngnotSn
DPB2kXYrG5LFu3Ssi4u96RNzO_ikxQ4nI1-Tp2MvubXMI7j8ZUfZ-
PLSt2iTaU_EOmMpfa8_jZFt2ZkPyK7IhApxvQwTY
vyjhtqjrF3vwbtNjDKSraibNVkSsJQn5bKG6qLjLokqyfdcL9tKm896bApLZ0o-
v5acnstyfL83NYafNvtNzyffwgjX0s%3D
#HttpOnly_us-3.rightscale.com FALSE /      FALSE 0      _session_id
      deadbeef0badf00dfeedfacec0ffee24
```

2.2 Verify Session

See: http://reference.rightscale.com/api1.5/resources/ResourceSessions.html#index



index

Returns a list of root resources so an authenticated session can use them as a starting point or a way to know what features are available within its privileges.

Example Request using Curl:
curl -i -H X_API_VERSION:1.5 -b mycookies -X GET https://my.rightscale.com/api/session

URLs:	GET /api/session
HTTP response code:	200 OK
Content-type:	application/vnd.rightscale.session

Required roles

- observer

Example Responses

example response in:

If you've authenticated successfully, you'll be able to perform further API Request for the duration of the Session Cookie (i.e. 2 Hours).

So to verify, you will now invoke an index request on the session resource. This will return a list of further resources you have access to once authenticated.

As indicated in the API Reference Documentation, to invoke an `index` request on the `session` resource, you invoke an `HTTP Get` request to the same API href as when creating the session, but you do not need to supply the `account_href`, `email` & `password`. You also need Observer permissions

4. Execute the following command

```
[api]# curl -H X-API-Version:1.5 -b mycookie -X GET https://us-3.rightscale.com/api/session
```

You should receive the following JSON response

```
{"links":[{"href":"/api/session","rel":"self"},{"href":"/api/clouds","rel":"clouds"},{"href":"/api/deployments","rel":"deployments"},{"href":"/api/servers","rel":"servers"},{"href":"/api/server_arrays","rel":"server_arrays"},{"href":"/api/server_templates","rel":"server_templates"},{"href":"/api/server_template_multi_cloud_images","rel":"server_template_multi_cloud_images"},{"href":"/api/multi_cloud_images","rel":"multi_cloud_images"},{"href":"/api/tags","rel":"tags"},{"href":"/api/backups","rel":"backups"},{"href":"/api/accounts","rel":"accounts"},{"href":"/api/cloud_accounts","rel":"cloud_accounts"},{"href":"/api/child_accounts","rel":"child_accounts"},{"href":"/api/users","rel":"users"},{"href":"/api/permissions","rel":"permissions"},{"href":"/api/audit_entries","rel":"audit_entries"},{"href":"/api/account_groups","rel":"account_groups"},{"href":"/api/publications","rel":"publications"},{"href":"/api/publication_lineages","rel":"publication_lineages"},{"href":"/api/identity_providers","rel":"identity_providers"},{"href":"/api/alert_specs","rel":"alert_specs"},{"href":"/api/security_group_rules","rel":"security_group_rules"}],"actions":[],"message":"You have successfully logged into the RightScale API."}
```

JSON is the default output, and is intended to be parsed by a script, e.g. Ruby, PHP, and as such it difficult to read. However there are a number of online tools you can use to reformat it to make it more legible, for example http://json.parser.online.fr/.

You can also produce an `.XML` response by appending `'.xml'` to the end of the URL request

5. Now invoke the command again with an `.xml` extension to get the same content in XML format

```
[api]# curl -H X-API-Version:1.5 -b mycookie -X GET https://us-
3.rightscale.com/api/session.xml
<?xml version="1.0" encoding="UTF-8"?>
<session>
  <links>
    <link href="/api/session" rel="self"/>
    <link href="/api/clouds" rel="clouds"/>
    <link href="/api/deployments" rel="deployments"/>
    <link href="/api/servers" rel="servers"/>
    <link href="/api/server_arrays" rel="server_arrays"/>
    <link href="/api/server_templates" rel="server_templates"/>
    <link href="/api/server_template_multi_cloud_images"
rel="server_template_multi_cloud_images"/>
    <link href="/api/multi_cloud_images" rel="multi_cloud_images"/>
    <link href="/api/tags" rel="tags"/>
    <link href="/api/backups" rel="backups"/>
    <link href="/api/accounts" rel="accounts"/>
    <link href="/api/cloud_accounts" rel="cloud_accounts"/>
    <link href="/api/child_accounts" rel="child_accounts"/>
    <link href="/api/users" rel="users"/>
    <link href="/api/permissions" rel="permissions"/>
    <link href="/api/audit_entries" rel="audit_entries"/>
    <link href="/api/account_groups" rel="account_groups"/>
    <link href="/api/publications" rel="publications"/>
    <link href="/api/publication_lineages" rel="publication_lineages"/>
    <link href="/api/identity_providers" rel="identity_providers"/>
    <link href="/api/alert_specs" rel="alert_specs"/>
    <link href="/api/security_group_rules" rel="security_group_rules"/>
  </links>
  <actions></actions>
  <message>You have successfully logged into the RightScale API.</message>
</session>
```

Note: Each of the hrefs returned in this request may be used in URLs in further API requests. The API Reference Documentation gives further details of each.

6. Now rerun the original command that returned JSON (without the `–i`) and pipe it through `python -mjson.tool` to reformat the JSON to a more readable format

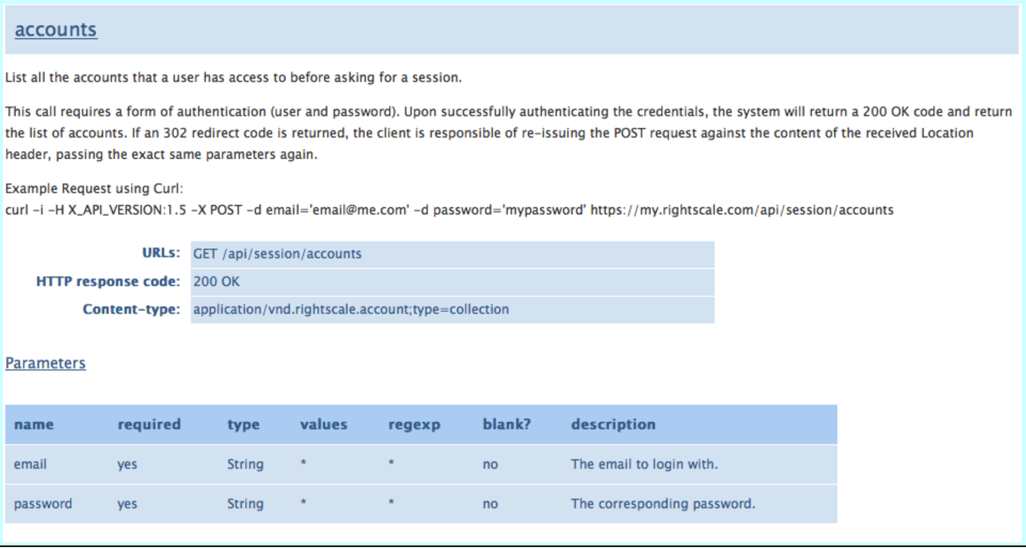
```
[api]# curl -H X-API-Version:1.5 -b mycookie -X GET https://us-
3.rightscale.com/api/session | python -mjson.tool
```

You should get the following response (truncated for brevity)

```
{
    "actions": [],
    "links": [
        {
            "href": "/api/session",
            "rel": "self"
        },
        {
            "href": "/api/clouds",
            "rel": "clouds"
        },
        …
        …
        {
            "href": "/api/identity_providers",
            "rel": "identity_providers"
        },
        {
            "href": "/api/alert_specs",
            "rel": "alert_specs"
        },
        {
            "href": "/api/security_group_rules",
            "rel": "security_group_rules"
        }
    ],
    "message": "You have successfully logged into the RightScale API."
}
```

2.3 List Accounts Access

See: http://reference.rightscale.com/api1.5/resources/ResourceSessions.html#accounts



accounts

List all the accounts that a user has access to before asking for a session.

This call requires a form of authentication (user and password). Upon successfully authenticating the credentials, the system will return a 200 OK code and return the list of accounts. If an 302 redirect code is returned, the client is responsible of re-issuing the POST request against the content of the received Location header, passing the exact same parameters again.

Example Request using Curl:
```
curl -i -H X_API_VERSION:1.5 -X POST -d email='email@me.com' -d password='mypassword' https://my.rightscale.com/api/session/accounts
```

URLs:	GET /api/session/accounts
HTTP response code:	200 OK
Content-type:	application/vnd.rightscale.account;type=collection

Parameters

name	required	type	values	regexp	blank?	description
email	yes	String	*	*	no	The email to login with.
password	yes	String	*	*	no	The corresponding password.

You will now use the `accounts` method on the `session` resource to view a list of all the RightScale accounts that you have access to.

To view the list you must invoke a `HTTP GET` to the href `/api/session/accounts`. As the documentation states, you must pass your email and password in the request to reauthenticate.

7. So execute the following command, substituting your own RightScale username and password

```
[api]# curl -H X-API-Version:1.5 -X GET -d email='email@example.com' -d
password='mypassword' https://us-3.rightscale.com/api/session/accounts.xml
<?xml version="1.0" encoding="UTF-8"?>
<accounts>
  <account>
    <created_at>2012/06/15 10:18:15 +0000</created_at>
    <updated_at>2012/07/11 20:14:49 +0000</updated_at>
    <links>
      <link href="/api/accounts/58650" rel="self"/>
      <link href="/api/users/57241" rel="owner"/>
      <link href="/api/clusters/1" rel="cluster"/>
    </links>
    <name>Development Work</name>
  </account>
  <account>
    <created_at>2010/12/04 00:06:13 +0000</created_at>
    <updated_at>2012/08/29 21:20:04 +0000</updated_at>
    <links>
      <link href="/api/accounts/33085" rel="self"/>
      <link href="/api/users/34141" rel="owner"/>
      <link href="/api/clusters/1" rel="cluster"/>
    </links>
    <name>QA</name>
  </account>
</accounts>
```

Note: If you wish you could include '`-i`' in the request (i.e. `curl –i –H` …) which will also return the HTTP headers in the output.

2.3.1 Invoke an API Request Without a Valid Session

Now you will remove your session cookie, hence revoking access via API Client. The response returned will be the same response you'd get if the cookie expired.

8. Delete your session cookie

```
[api]# rm mycookie
```

9. Execute the following command again to retrieve a list of available API resources

```
[api]# curl -H X-API-Version:1.5 -b mycookie -X GET https://us-
3.rightscale.com/api/session
```

10. Since there is no valid session cookie you will receive the following error

```
Permission denied
```

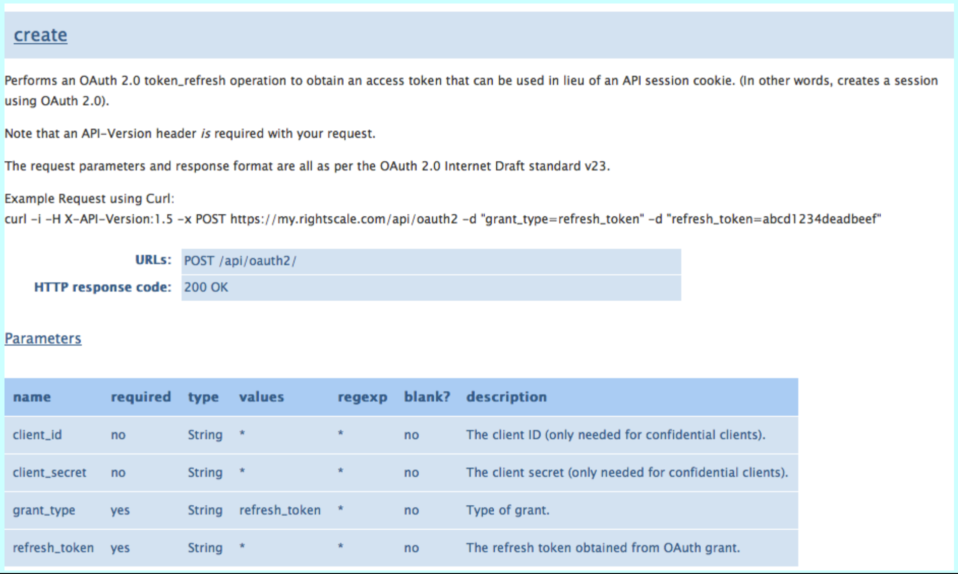
As expected

11. Now create a new session and try the command again

```
[api]# curl –i -H X-API-Version:1.5 -c mycookie -X POST -d \
email=jdoe@example.com -d password=mypassword -d \
account_href=/api/accounts/12345 https://us-3.rightscale.com/api/session
[api]# curl -H X-API-Version:1.5 -b mycookie -X GET https://us-
3.rightscale.com/api/session
{"links":[{"href":"/api/session","rel":"self"},{"href":"/api/clouds","rel"
:"clouds"},{"href":"/api/deployments","rel":"deployments"},{"href":"/api/s
ervers","rel":"servers"},{"href":"/api/server_arrays","rel":"server_arrays
"},{"href":"/api/server_templates","rel":"server_templates"},{"href":"/api
/server_template_multi_cloud_images","rel":"server_template_multi_cloud_im
ages"},{"href":"/api/multi_cloud_images","rel":"multi_cloud_images"},{"hre
f":"/api/tags","rel":"tags"},{"href":"/api/backups","rel":"backups"},{"hre
f":"/api/accounts","rel":"accounts"},{"href":"/api/cloud_accounts","rel":"
cloud_accounts"},{"href":"/api/child_accounts","rel":"child_accounts"},{"h
ref":"/api/users","rel":"users"},{"href":"/api/permissions","rel":"permiss
ions"},{"href":"/api/audit_entries","rel":"audit_entries"},{"href":"/api/a
ccount_groups","rel":"account_groups"},{"href":"/api/publications","rel":"
publications"},{"href":"/api/publication_lineages","rel":"publication_line
ages"},{"href":"/api/identity_providers","rel":"identity_providers"},{"hre
f":"/api/alert_specs","rel":"alert_specs"},{"href":"/api/security_group_ru
les","rel":"security_group_rules"}],"actions":[],"message":"You have
successfully logged into the RightScale API."}
```

Section 3. Session Creation with OAuth

See: http://reference.rightscale.com/api1.5/resources/ResourceOauth2.html



create

Performs an OAuth 2.0 token_refresh operation to obtain an access token that can be used in lieu of an API session cookie. (In other words, creates a session using OAuth 2.0).

Note that an API-Version header *is* required with your request.

The request parameters and response format are all as per the OAuth 2.0 Internet Draft standard v23.

Example Request using Curl:

```
curl -i -H X-API-Version:1.5 -x POST https://my.rightscale.com/api/oauth2 -d "grant_type=refresh_token" -d "refresh_token=abcd1234deadbeef"
```

URLs:	POST /api/oauth2/
HTTP response code:	200 OK

Parameters

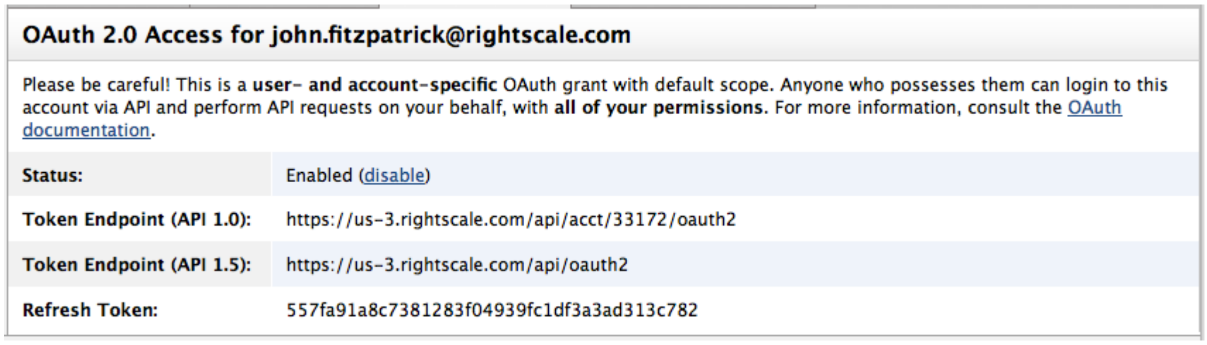
name	required	type	values	regexp	blank?	description
client_id	no	String	*	*	no	The client ID (only needed for confidential clients).
client_secret	no	String	*	*	no	The client secret (only needed for confidential clients).
grant_type	yes	String	refresh_token	*	no	Type of grant.
refresh_token	yes	String	*	*	no	The refresh token obtained from OAuth grant.

You will now create a session using OAuth instead using your username and password. You will use bash scripts in this example.

3.1 Enable OAuth & Obtain your Refresh Token and Endpoint

To create a session using OAuth you must make an `HTTP POST` to the href `/api/oauth2/`, and passing a **refresh_token** as a parameter.

12. In the Dashboard browse to '**Settings > Account Settings > API Credentials**' and ensure the **Status** is set to `Enabled`



OAuth 2.0 Access for john.fitzpatrick@rightscale.com

Please be careful! This is a **user- and account-specific** OAuth grant with default scope. Anyone who possesses them can login to this account via API and perform API requests on your behalf, with **all of your permissions**. For more information, consult the OAuth documentation.

Status:	Enabled (disable)
Token Endpoint (API 1.0):	https://us-3.rightscale.com/api/acct/33172/oauth2
Token Endpoint (API 1.5):	https://us-3.rightscale.com/api/oauth2
Refresh Token:	557fa91a8c7381283f04939fc1df3a3ad313c782

13. Note **Token Endpoint (API 1.5)** and **Refresh Token**

    You will be asked to copy/paste these shortly

3.2 Update Authentication Script & Obtain Access Token

14. SSH into your 'API Sender' server, switch to user `root` and

```
[api]# sudo -i
```

15. Edit the file `/opt/api/OAuth/auth-oauth.sh`

```
[api]# cd /opt/api/OAuth
[api]# vim auth-oauth.sh
```

16. Copy/paste the values for `my_token_endpoint` and `my_refresh_token` you retrieved from the Dashboard ('**Settings > Account Settings > API Credentials**')

```
#!/bin/bash

my_token_endpoint="https://us-3.rightscale.com/api/oauth2"
my_refresh_token="557fa91a8c7381283…"

curl --include \
  -H "X-API-Version:1.5" \
  --request POST "$my_token_endpoint" \
  -d "grant_type=refresh_token" \
  -d "refresh_token=$my_refresh_token"
```

17. Run the script to obtain the Access Token

```
[api]# ./auth-oauth.sh
HTTP/1.1 200 OK
Server: nginx/1.0.14
Date: Wed, 20 Mar 2013 12:52:06 GMT
Content-Type: application/json; charset=utf-8
Connection: keep-alive
Status: 200 OK
X-Runtime: 1851
Pragma: no-cache
Content-Length: 419
Set-Cookie:
Cache-Control: private, max-age=0, must-revalidate

{"access_token":"eNotkFtvgjAARv9Ln21CLwIl2QO3QYgVZWqcLwZLq5UxptAoM_73YbLHL
_nOeTgPUAIPmA4SMAFVB7zHOOQVeJhSSp4T0AvgIWITx6XURROgq_FPCaFCUgsyNJUQIYmhSxw
FlWQOQ6WDlBSjr5f_LEP2ix31YMPdZt7LDct4kc-tRW-ZmcptC12-
ppal2ipRea6StQ8vmuPVb7ibd7Osdd4PbPkBI3yuu85PzkjnwXEXut26vcW3n9h3Rbo0h12gVz
U3n1VUXFHRGJ1pHOG4P4ZheiKLLS8zyi9p2p5mq7oawiA-
b9mVMzYU9ULfg2YPE9bQoTb7Qby9ktxfSUohWvPdA48Q5ODn8w_VWFx4","expires_in":720
0,"token_type":"bearer"}
```

3.3 Use Access Token to Issue Request

Now you will now use this Access Token to make a request

18. Edit the file `Accounts-List-OAuth.sh`

```
[api]# cd /opt/api/OAuth
[api]# vim Accounts-List-OAuth.sh
```

19. Enter the Access Token retrieved in the last request

```
#!/bin/bash
access_token="eNotkFtvgjAARv9Ln21CLwIl2QO3QYgVZWqcLwZLq5UxptAoM_73YbLHL_nO
eTgPUAIPmA4SMAFVB7zHOOQVeJhSSp4T0AvgIWITx6XURROgq_FPCaFCUgsyNJUQIYmhSxwFlW
QOQ6WDlBSjr5f_LEP2ix31YMPdZt7LDct4kc-tRW-ZmcptC12-
ppal2ipRea6StQ8vmuPVb7ibd7Osdd4PbPkBI3yuu85PzkjnwXEXut26vcW3n9h3Rbo0h12gVz
U3n1VUXFHRGJ1pHOG4P4ZheiKLLS8zyi9p2p5mq7oawiA-
b9mVMzYU9ULfg2YPE9bQoTb7Qby9ktxfSUohWvPdA48Q5ODn8w_VWFx4"

curl --include \
     -H "X-API-Version:1.5" \
     -H "Authorization: Bearer $access_token" \
     --request GET "https://us-3.rightscale.com/api/session.xml"
```

20. Run the script to make the sample API Request

```
[api]# ./Accounts-List-OAuth.sh
<?xml version="1.0" encoding="UTF-8"?>
<session>
  <actions></actions>
  <message>You have successfully logged into the RightScale API.</message>
  <links>
    <link href="/api/session" rel="self"/>
    <link href="/api/clouds" rel="clouds"/>
    <link href="/api/deployments" rel="deployments"/>
    <link href="/api/servers" rel="servers"/>
    <link href="/api/server_arrays" rel="server_arrays"/>
    <link href="/api/server_templates" rel="server_templates"/>
    <link href="/api/server_template_multi_cloud_images"
rel="server_template_multi_cloud_images"/>
    <link href="/api/multi_cloud_images" rel="multi_cloud_images"/>
    <link href="/api/tags" rel="tags"/>
    <link href="/api/backups" rel="backups"/>
    <link href="/api/accounts" rel="accounts"/>
    <link href="/api/cloud_accounts" rel="cloud_accounts"/>
    <link href="/api/child_accounts" rel="child_accounts"/>
    <link href="/api/users" rel="users"/>
    <link href="/api/permissions" rel="permissions"/>
    <link href="/api/audit_entries" rel="audit_entries"/>
    <link href="/api/account_groups" rel="account_groups"/>
    <link href="/api/publications" rel="publications"/>
    <link href="/api/publication_lineages" rel="publication_lineages"/>
    <link href="/api/identity_providers" rel="identity_providers"/>
    <link href="/api/alert_specs" rel="alert_specs"/>
    <link href="/api/security_group_rules" rel="security_group_rules"/>
  </links>
</session>
```

3.4 Automating Session Creation with Tokens

Using Token can appear cumbersome as the Access Token changes with every request. However, this process can be automated using scripts.

21. Navigate to the directory `/opt/api/OAuth/automated`

```
[api]# cd /opt/api/OAuth/automated
```

22. Edit the file `RefreshToken` and add the token you got from the Dashboard in step 12 above.

```
[api]# vim RefreshToken
my_refresh_token=e76042a6c9561234567894c95bcab0b875ce97b3
```

23. View the script `GetNewAccessToken.sh`

```
[api]# cat ./GetNewAccessToken.sh
#!/bin/bash
my_token_endpoint="https://us-3.rightscale.com/api/oauth2"
my_refresh_token=`grep my_refresh_token RefreshToken|cut -c 18-|tr -d '\r'`

#The following gives the Access Token based on the Refresh Token
curl -H "X-API-Version:1.5" \
  --request POST "$my_token_endpoint" \
  -d "grant_type=refresh_token" \
  -d "refresh_token=$my_refresh_token" \
|python -mjson.tool|grep access_token|cut -c22-381|tr -d '\r' > AccessToken
```

Notice that this script read the Refresh Token from the file **RefreshToken** as an input, and outputs the Access Token to the file **AccessToken**. Further scripts can then read the Access Token from this file.

24. Now run this script to generate a valid Access Token & store it in the file '**AccessToken**'.

```
[api]# ./GetNewAccessToken.sh
[api]# cat AccessToken
nzAU5hAcxP-XPkvSb1tLS7IHEOavBZ1ag7wsUIqi4qbCRA3_-zB7ueSS-
1xy90QJclB9tSjqoeyKnGdnzAU5hDFG2x6qNHKAcm5Lyfq8h4qsy2MtBLF1YiW6EwBDLJkyaQG
hQDMMXGDR9VXmnxWY8xfbtPCvtfLnalVt-8cdVGqauXmfykEfHcBCaFQniueTsVnvNRC-
H_EhxIiUv16_QS2pVQx34_WVI_Y4mGS8ecJ3r3hdHObWGyqZ8XlJ4sKUDtmXDYOznqvYiUPPl7
7Rb_O87mKduEp2TT-oAlcLzRDwMtgpNXRhvBx_4r3wpi7eHtd0rwu6WZ-16cKOZSCTdr2DxwC
```

25. Now you can invoke an API request using this Access Token, for example, in the script `Accounts-List-OAuth__auto.sh`

```
#!/bin/bash

access_token=`cat AccessToken`

curl -H "X-API-Version:1.5" \
     -H "Authorization: Bearer $access_token" \
     --request GET "https://us-3.rightscale.com/api/session.xml"
```

26. Now run the scripts `Accounts-List-OAuth__auto.sh`

```
[api]# ./Accounts-List-OAuth__auto.sh
```