Using right-api-client

John Fitzpatrick

Table of Contents

Lab Overview and Requirements

Overview

This lab introduces the RightScale right_api_client, and using Ruby scripts to invoke API requests.

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

- 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. Getting Started With right_api_client

In this lab you will authenticate with RightScale API to create a new session using the `right-api-client` script.

1.1 Simple Authentication Ruby Script

Authentication works differently for `right-api-client` as is does for to the `curl` examples. When using standard authentication with `curl`, a session cookie is stored in a local file (`mycookie` in our examples). The session cookie lasts for two hours, you're not required to re-authenticate during that time for subsequent API calls.

With the `right-api-client`, however, the authentication information is stored as an object in memory. Hence, each Ruby script you develop and execute will require authenticating with the API.

1.1.1 Authenticate using right-api-client

Since authentication information is stored in memory, and there is no cookie, there is no obvious sign that you authenticated correctly (although you will get an error if you don't!). So in this example you will authenticate and at the same time invoke an index request on the session resource to return a list of further resources you have access,

1. Run the following commands to install `right-api-client`

```
[api]# gem install right_api_client
```

2. View the Ruby script you will use to authenticate, `/opt/api/right-api-client/auth/auth-ruby.rb`

```
[api]# cd /opt/api/right-api-client/auth
[api]# cat auth-ruby.rb
```

   This script should be prepopulated with your email address you use to log into RightScale and the RightScale account you will authenticate in. When you run the script it will prompt you for your RightScale account password.

```
require 'rubygems'
require 'pp'
require 'right_api_client'

puts "Please enter your account password:-"
`stty -echo`
mypassword = STDIN.gets.chomp()
`stty echo`

@client = RightApi::Client.new(:api_url => 'https://us-3.rightscale.com', :email
=> 'john.fitzpatrick@rightscale.com', :password => mypassword, :account_id =>
'33172')

puts "Available methods: #{pp @client.api_methods}"
```

Note: pp (or 'pretty print') reformats the output to a more readable format.

3. Then as execute the script as follows

```
[api]# ruby auth-ruby.rb
Please enter your account password:- *******
[:session,
 :clouds,
 :deployments,
 :servers,
 :server_arrays,
 :server_templates,
 :server_template_multi_cloud_images,
 :multi_cloud_images,
 :tags,
 :backups,
 :accounts,
 :cloud_accounts,
 :child_accounts,
 :users,
 :permissions,
 :audit_entries,
 :account_groups,
 :publications,
 :publication_lineages,
 :identity_providers,
 :alert_specs,
 :security_group_rules]
Available methods: [:session, :clouds, :deployments, :servers,
:server_arrays, :server_templates, :server_template_multi_cloud_images,
:multi_cloud_images, :tags, :backups, :accounts, :cloud_accounts,
:child_accounts, :users, :permissions, :audit_entries, :account_groups,
:publications, :publication_lineages, :identity_providers, :alert_specs,
:security_group_rules]
```

You can see a list of methods returned indicating you authenticated correctly.

Section 2. Issuing Commands Using restclient

In this example you will use the `restclient` IRB environment to issue a number of requests.

2.1 Session Creation Script

In the previous example the Username and Password were explicitly defined in the script for demonstration purposes. This is not ideal, as you do not want anyone to view your account password.

In this example configure your username and account number in a separate file, and read it in, and prompt for the password to be entered at command line.

4. SSH into the API Sender server, `sudo` to user root, and navigate to the directory `/opt/api/right-api-client/auth`

```
[api]# sudo –i
```

5. Edit the file `/opt/api/right-api-client/auth/apirc` and populate it with your RightScale account password

```
[api]# vim apirc
defaults:
  user: myemail@example.com
  account: 33172
  accountname: accountname
  apiurl: 'https://us-3.rightscale.com'
```

   This file is used by the file `session.rb` to authenticate and create a session. The file `session.rb` can then be referenced by other files.

6. View the file `session.rb`

   This file reads `user` and `account` from the file `apirc` then prompts you for your `password` and authenticates you in the account

```
[api]# cat session.rb
require 'rubygems'
require 'pp'
require 'right_api_client'
require 'yaml'
require 'uri'

def init_auth
  parsed_file=YAML.load(File.read(File.join(File.dirname(__FILE__),"apirc")))
  @user=parsed_file["defaults"]["user"]
  @account=parsed_file["defaults"]["account"]
  @apiurl=parsed_file["defaults"]["apiurl"]
end

def password
 puts "Please enter your account password:-"
 `stty -echo`
 @mypassword = STDIN.gets.chomp()
 `stty echo`
end

@client=''
def init_client(apiurl,email,mypassword,id)
  @client = RightApi::Client.new(:api_url => apiurl, :email => email, :password =>
mypassword, :account_id => id)
  puts "Authenticated"
end

def init
  init_auth
  password
  init_client(@apiurl,@user,@mypassword,@account)
end

init
```

7. Now invoke this script

```
[api]# ruby session.rb
Please enter your account password:-
Authenticated
[api]#
```

2.2 Using restclient

8. Now run invoke the `restclient` IRB environment, and pull in the file `session.rb` to create a session

```
[api]# restclient
irb(main):001:0> require '/opt/api/right-api-client/auth/session.rb'
Please enter your account password:-
Authenticated
=> true
irb(main):002:0>
```

Once you're Authenticated in the `restclient` session you can invoke API requests directly in the IRB session without having to re-authenticate each time.

2.2.1 View Available Methods

9. Issue the following command to view all available methods (`pp` is used for formatting only)

```
[irb(main):001:0> pp @client.api_methods
[:session,
 :clouds,
 :deployments,
 :servers,
 :server_arrays,
 :server_templates,
 :server_template_multi_cloud_images,
 :multi_cloud_images,
 :tags,
 :backups,
 :accounts,
 :cloud_accounts,
 :child_accounts,
 :users,
 :permissions,
 :audit_entries,
 :account_groups,
 :publications,
 :publication_lineages,
 :identity_providers,
 :alert_specs,
 :security_group_rules]
=> [:session, :clouds, :deployments, :servers, :server_arrays,
:server_templates, :server_template_multi_cloud_images,
:multi_cloud_images, :tags, :backups, :accounts, :cloud_accounts,
:child_accounts, :users, :permissions, :audit_entries, :account_groups,
:publications, :publication_lineages, :identity_providers, :alert_specs,
:security_group_rules]
```

You will now run a selection of these methods

2.2.2 Show Deployments

10. Issue the following command to view deployments

```
[irb(main):002:0> pp @client.deployments.index
[#<RightApi::ResourceDetail resource_type="deployment", name="Default">,
#<RightApi::ResourceDetail resource_type="deployment", name="API Sender
Deployment">]
=> [#<RightApi::ResourceDetail resource_type="deployment",
name="Default">, #<RightApi::ResourceDetail resource_type="deployment",
name="API Sender Deployment">]
```

2.2.3 Show Servers

11. Issue the following command to show all Servers

```
[irb(main):003:0> pp @client.servers.index
[#<RightApi::ResourceDetail resource_type="server", name="API Sender
[RSED]">]
=>[#<RightApi::ResourceDetail resource_type="server", name="API Sender
[RSED]">]
```

2.2.4 Show ServerTemplates

12. Issue the following command to view ServerTemplates

```
 [irb(main):002:0>  pp @client.server_templates.index
[#<RightApi::ResourceDetail resource_type="server_template", name="LAMP
All-In-One Wordpress Trial">,
#<RightApi::ResourceDetail resource_type="server_template", name="API
Sender [RSED]">,
 #<RightApi::ResourceDetail resource_type="server_template", name="Base
ServerTemplate for Linux (v13.3) v1">,
 #<RightApi::ResourceDetail resource_type="server_template", name="Load
Balancer with HAProxy (v13.2.1) [RSED]">,
 #<RightApi::ResourceDetail resource_type="server_template", name="PHP App
Server (v13.2.1) [RSED]">,
 #<RightApi::ResourceDetail resource_type="server_template",
name="Database Manager for MySQL 5.5 (v13.2.1) [RSED]">,]
```

2.2.5 Show Servers in Deployment

13. Issue the following command to view the Servers in a deployment, where `id` is the Deployment ID.

```
[irb(main):002:0> pp @client.deployments(:id => 377669003).show.servers.index
[#<RightApi::ResourceDetail resource_type="server", name="jaf App Server 1">,
 #<RightApi::ResourceDetail resource_type="server", name="jaf App Server 2">,
 #<RightApi::ResourceDetail resource_type="server", name="jaf Database">,
 #<RightApi::ResourceDetail resource_type="server", name="jaf Load Balancer 1">,
 #<RightApi::ResourceDetail resource_type="server", name="jaf Load Balancer 2">]
=> [#<RightApi::ResourceDetail resource_type="server", name="jaf App Server 1">,
#<RightApi::ResourceDetail resource_type="server", name="jaf App Server 2">,
#<RightApi::ResourceDetail resource_type="server", name="jaf Database">,
#<RightApi::ResourceDetail resource_type="server", name="jaf Load Balancer 1">,
#<RightApi::ResourceDetail resource_type="server", name="jaf Load Balancer 2">]
```

2.3 Exploring Available Methods

In this section you will explore further available methods by quering `.api_methods` for each resource returned

14. Execute the following commands with `restclient` IRB session

```
irb> @client.deployments
=> #<RightApi::Resources resource_type="deployments">
irb> @client.deployments.api_methods
=> [:create, :index]
irb> @client.deployments.index
=> [#<RightApi::ResourceDetail resource_type="deployment",
name="Default">, #<RightApi::ResourceDetail resource_type="deployment",
name="API Senders Deployment">]
Irb>@client.deployments(:id => 391056003)
=> #<RightApi::Resource resource_type="deployment">
irb> @client.deployments(:id => 391056003).api_methods
=> [:destroy, :update, :show]
irb> @client.deployments(:id => 391056003).show
=> #<RightApi::ResourceDetail resource_type="deployment", name="jf test">
irb(main):031:0> @client.deployments(:id => 391056003).show.api_methods
=>[:links, :clone, :servers, :server_arrays, :inputs, :server_tag_scope,
:description, :name, :href, :destroy, :update, :show]
irb> @client.deployments(:id => 391056003).show.href
=> "/api/deployments/391056003"
```

Section 3. Launch Servers using right_api_client

3.1 Simple Script to Launch a Single Server

In this example you will use a simple Ruby script to create a Deployment and a Base Linux Server

15. SSH into API Sender and navigate to `/opt/development/right-api-client`

```
[api]# cd /opt/api/right-api-client/
```

16. Run the Ruby Script `Deployment-Server-Create.rb`, and follow the instructions

```
[api]# ruby Deployment-Server-Create.rb
Please enter your account password:-
Authenticated
What would you like to call your deployment?
Joe Doe's Deployment
What would you like to call your server?
Joe Doe's Server
Creating Server....
Starting Server....
The Server 'Joe Doe's Server' has been launched.  Now go verify this
Deployment and Server in the Dashboard
```

17. Now view the Ruby Script `Deployment-Server-Create.rb`, and in particular, the API commands contained within it.

```
[api]# cat Deployment-Server-Create.rb
require 'rubygems'
require 'pp'
require 'right_api_client'
require 'yaml'
require 'uri'
require './auth/session.rb'

puts "What would you like to call your deployment?"
depname = STDIN.gets.chomp()

puts "What would you like to call your server?"
servname = STDIN.gets.chomp()

puts "Creating Server...."
deploy_href = @client.deployments.create({:deployment => {:name =>
depname}}).href

server_template_href = @client.server_templates.index(:filter =>
['name==Base ServerTemplate']).first.href
cloud = @client.clouds(:id => '1').show
 params = { :server => {
 :name => servname,
 :deployment_href => deploy_href,
 :instance => {
   :server_template_href => server_template_href,
   :cloud_href  => cloud.href,
   :security_group_hrefs => [cloud.security_groups.index(:filter =>
['name==default']).first.href],
   :ssh_key_href  => cloud.ssh_keys.index.first.href,
   :datacenter_href  => cloud.datacenters.index.first.href
  }}}
new_server = @client.servers.create(params)
new_server.api_methods

#Launch the Server
puts "Starting Server...."
```

```
new_server.show.launch
"\n"
puts "The Server '#{servname}' has been launched.  Now go verify this
Deployment and Server in the Dashboard"
```

3.2 Simple Script to Launch 3 Tier Application (Beta)

18. Edit the file `/opt/api/right-api-client/ruby3tier/labinfo.yaml` and populate it with the same information as `/opt/api/3tier/LabInfo`, i.e.

```
[api]# vi /opt/api/right-api-client/ruby3tier/labinfo.yaml
inputs:
  dbfqdn: jdoe-masterdb.rightscaletraining.com
  ddnsid: 10079884
  cloud: 1
  myname: jdoe
  dblineage: jdoelinage
  dbschema: jdoeschema
```

19. Run the Ruby Script `3Tier-Create.rb`, and follow the instructions

```
[api]# ./3Tier-Create.rb
Please enter your account password:-
Authenticated
What would you like to call your deployment?
John test deployment
Creating Deployment....
Deployment '/api/deployments/391056003' created
Creating LB Server....
Creating App Server....
Creating DB Server....
Setting Inputs...
text 'text:johnschema'
array 'array:php53u-mysql,php53u-pecl-memcache'
text 'text:false'
text 'text:prefork'
text 'text:adminuser'
text 'text:adminpassword'
text 'text:appuser123'
text 'text:apppasword123'
text 'text:$DBLINAGE'
text 'text:john12345-masterdb.rightscaletraining.com'
text 'text:10079884'
cred 'cred:DUMP_CONTAINER'
text 'text:john5schema'
text 'text:sampledb'
cred 'cred:AWS_ACCESS_KEY_ID'
cred 'cred:AWS_SECRET_ACCESS_KEY'
text 'text:repluser'
text 'text:replpassword'
text 'text:db_mysql_5.5'
text 'text:git://github.com/rightscale/examples.git'
text 'text:unified_php'
text 'text:DNSMadeEasy'
cred 'cred:DNS_PASSWORD'
cred 'cred:DNS_USER'
Starting Servers....
The Server 'john Load Balancer Server' has been launched
```

```
The Server 'john Application Server' has been launched
The Server 'john Database Server' has been launched
```

20. Now view the Ruby Script `3Tier-Create.rb`, and in particular, the API commands contained within it.

```
[api]# cat 3Tier-Create.rb
```