



## Creating a 3-Tier Deployment, Arrays, Monitoring and Tags

John Fitzpatrick

## Table of Contents

Lab Overview and Requirements.....	4
Overview .....	4
Lab Environment.....	4
Requirements.....	4
Using Bash Scripts.....	5
Section 1. Create 3-Tier Deployment.....	6
1.1 Create a Session .....	6
1.2 Configure DNS .....	8
1.3 Set Cloud ID and Name Identifier Used in Lab Scripts.....	10
1.4 Create 3-Tier Deployment.....	12
1.5 Update DNS.....	13
1.6 Configure Master Database.....	15
1.7 Test The Application.....	16
1.8 Run Operational Script to Update Application.....	17
1.9 Testing The New Application.....	21
Section 2. Server Arrays .....	22
2.1 Create Array .....	22
2.2 Enable the Array.....	25
Section 3. Monitoring Servers.....	26
3.1 Select a Server To Monitor.....	26
3.2 View Metrics for Server .....	27
Section 4. Using Tags .....	30
4.1 Find all App Servers with a Certain Tag.....	30
4.2 Set Custom Tags on Load Balancer Servers.....	31
Section 5. Terminate & Delete All Servers.....	32
5.1 Disable the Array.....	32
5.2 Terminate All Servers .....	33
5.3 Delete the Array.....	34
5.4 Delete all Servers.....	35
5.5 Delete Deployment.....	36



## Lab Overview and Requirements

### Overview

This lab walks you through creating a 3-Tier Deployment using the RightScale API. This deployment is similar to the PHP 3Tier Deployment created in the IT Vending Machine demonstration.



All tasks required to create the 3-Tier Deployment are fully automated, and require running a number of scripts in a certain order.

These tasks include:-

- Create Security Group
- Create SSH Key
- Import ServerTemplates
- Create Deployment
- Create Servers
- Set Inputs
- Launch Servers
- Run Recipes and RightScripts on Servers

Once the 3Tier Application is up and running you will Create and Enable a Server Array, and perform monitoring tasks on it.

### 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](http://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 `vim` editor
- This lab assumes you already have an **myname API Sender** server running, based on the ServerTemplate **"API Sender [RSED]"**

### Using Bash Scripts

In this lab you will mainly be invoking API requests using bash scripts. This will allow you to rerun the commands again without having to type them in each time.

The bash scripts you'll use are in the following directory on your **myname API Sender** server

```
/opt/api/3tier
```

The parameters passed in the API request (e.g. Email , Password, Account Number, Deployment ID, etc) will be extracted out as variables in the script, for example

```
#!/bin/bash -e
DEPLOYMENT="123456789"      # Set the Deployment to "12346789"

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X \
GET https://us-3.rightscale.com/api/deployments/$DEPLOYMENT/servers.xml \
| tee output/${0##*/}.out
```

**Note:** The session cookie in this case is stored in your home directory, as indicated by the `~/`. Also, the line `tee output/${0##*/}.out` pipes the script output to a file, as well as screen. This file will be in the `output/` directory, and the name will be the same as the script name, but with `.out` appended. So in the example the file would be `output/My-API-Script.sh.out`

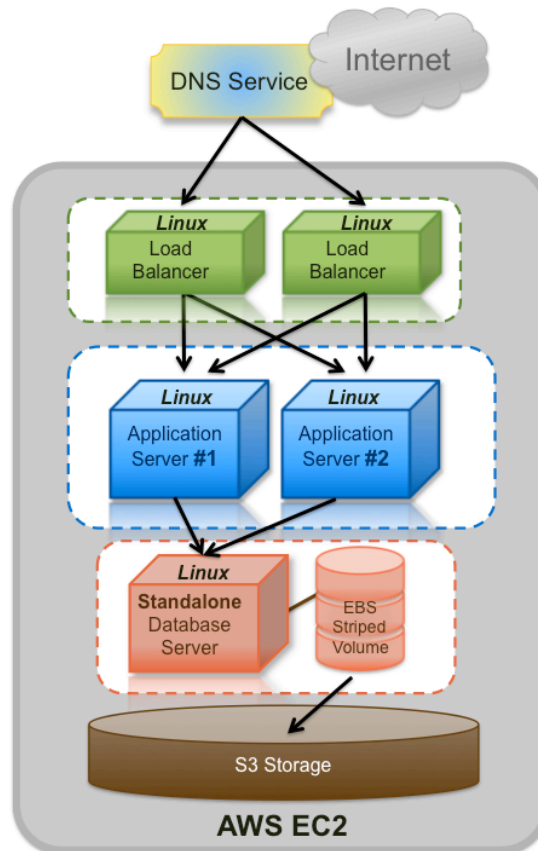
As the lab progresses the parameters will get read into scripts automatically from the output of previous scripts.

Many of the scripts used in this lab require information that is particular to your environment - ID number of the cloud you intend to build your 3Tier deployment in, FQDN of load balancers and database servers, as well as a name that can be used to distinguish the objects you create from those created by other in the class, e.g. Deployment, Security Groups, etc.

Instead of enter this information manually into each script, you only have to enter it once in the file `LabInfo`. Each script then read the information from this file.

## Section 1. Create 3-Tier Deployment

The previous lab walked you through Server and Deployment creation in some detail, so now we will introduce some automation. This section will see how you can quickly stand up a 3-Tier deployment using the API. You will then use this 3-Tier deployment in the next section to walk through how to create & configure a Server Array, Tags and how to Monitor Servers.



### 1.1 Create a Session

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

First you need to create a session before you can issue an API Request, but this time you'll do it using a script.

1. SSH Into the API Sender server and **sudo** to root **user**

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

2. Run the script `/opt/api/3tier/auth-bash.sh`

```
[api]# cd /opt/api/3tier
[api]# ./auth-bash.sh
...
Status: 204 No Content
...
```

You should have received a **Status: 204 No Content** in the reply, indicating you now have a valid session.

3. Create an alias to make it easier to invoke API commands from the command line.

```
[api]# alias mycurl='curl -s -H X-API-Version:1.5 -b ~/mycookie '
```

For the remainder of this lab you will be working in the directory `/opt/api/3tier`, so navigate to that directory now

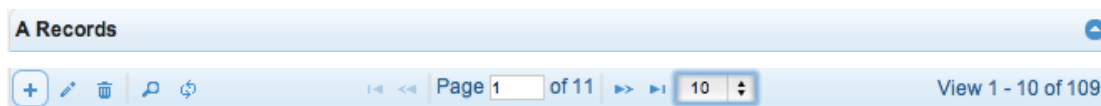
```
[api]# cd /opt/api/3tier
```

## 1.2 Configure DNS

Before you can commence setting up a 3Tier Application, you need to configure a number of DNS Records. This cannot be done via the RightScale API, but is done by logging into the DNS Provider's site directly.

### 1.2.1 Establish Database DNS Record

1. Browse to the website <http://www.dnsmadeeasy.com> and Log in  
The Instructor will give you the Username and Password
2. Click on the **rightscaletraining.com** domain link in the middle of the page under 'Recently Updated Domains'
3. Click the little '+' icon to create a dynamic Class A record for the Master database



Name: **myname-masterdb.rightscaletraining.com**

IP: **1.2.3.4** (this is a placeholder IP address which will be updated with the instance's private IP address by a script during server boot)

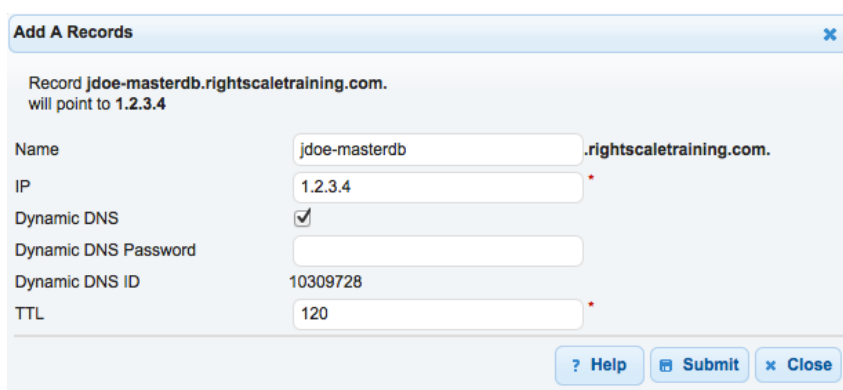
Dynamic DNS: **Checked**

**Do not** set DNS password

TTL: **30**

4. Click **Submit**

The **Dynamic DNS ID** will be auto-generated



5. Record the **Dynamic DNS ID** for the Master DB Record ID (You will need this value later)  
**Master DB Dynamic DNS ID:** \_\_\_\_\_
6. Update the file [LabInfo](#) with the Master DB Record ID (DDNS ID), and the FQDN of the Master Database Server.



```
[api]# cd /opt/api/3tier  
[api]# vi LabInfo
```

This file contains a number of variables that are read into script in this lab, and saves you from having to enter the values a number of times. The file should look like the following

```
#Fully Qualified Domain Name of DB Server  
DBFQDN="myname-masterdb.rightscaletraining.com"  
  
#DDNS ID of DB Server  
DDNSID=10079884
```

### 1.2.2 Establish Load Balancer DNS Record

#### 7. Create non-dynamic Class A records for Load Balancing tier

- A Record 1:

Name: **myname-www.rightscaletraining.com**

IP: **1.2.3.4** (this is a placeholder IP address which will be updated later once you've created your Elastic IPs)

Dynamic DNS: **LEAVE UNCHECKED**

TTL: **1800**

- A Record 2:

Name: **myname-www.rightscaletraining.com**

IP: **2.3.4.5** (this is a placeholder IP address which will be updated later once you've created your Elastic IPs)

Dynamic DNS: **LEAVE UNCHECKED**

TTL: **1800**

### 1.3 Set Cloud ID and Name Identifier Used in Lab Scripts

See: <http://reference.rightscale.com/api1.5/resources/ResourceClouds.html>

8. Invoke the following command to see a list of clouds

```
[api]# mycurl https://us-3.rightscale.com/api/clouds.xml
<?xml version="1.0" encoding="UTF-8"?>
<clouds>
  <cloud>
    <links>
      <link href="/api/clouds/1" rel="self"/>
      <link href="/api/clouds/1/datacenters" rel="datacenters"/>
      <link href="/api/clouds/1/instance_types" rel="instance_types"/>
      <link href="/api/clouds/1/security_groups" rel="security_groups"/>
      <link href="/api/clouds/1/instances" rel="instances"/>
      ...
      <link href="/api/clouds/1/volume_snapshots" rel="volume_snapshots"/>
      <link href="/api/clouds/1/volume_types" rel="volume_types"/>
      <link href="/api/clouds/1/volumes" rel="volumes"/>
    <description>Amazon's US Cloud on the East Coast</description>
    <name>EC2 us-east-1</name>
    <cloud_type>amazon</cloud_type>
  </cloud>
  <cloud>
    ...
  </cloud>
</clouds>
```

9. Decide on a cloud or AWS Region you're going to use to build your 3Tier deployment in. The Cloud ID can be identified from the href URL, e.g. 1 in the above example for **EC2 us-east-1**.

**Note:** Although you can use any Cloud, AWS impose a limit of 5 Elastic IPs per cloud. So you may run into difficulties with other in the class. However, cloud "EC2 us-east-1" (Cloud '1') has this restriction lifted so you may wish to use that one.

10. Now update the file **LabInfo** and populate it with

- The number for the cloud you have chosen (**CLOUD**)
- Your own name (**MYNAME**)

- The Database Schema name (**DBSCHEMA**)
- The Database Lineage name (**DBLINAGE**)

**Note:** The scripts used in this lab will read these values from this file, saving you to having to manually enter them into each script. The **DBLINAGE** and **DBSCHEMA** values are arbitrary.

```
[api]# cd /opt/api/3tier
[api]# vi LabInfo
#Replace with your cloud number
CLOUD=1
#Replace with your own name
MYNAME=jdoe
#Database Lineage, e.g. jdoelinage
DBLINAGE=jdoelinage
#Database schema name, e.g. jdoeschema
DBSCHEMA=jdoeschema
```

## 1.4 Create 3-Tier Deployment

11. Now navigate to directory `/opt/api/3tier`

12. Now run the following scripts in the order shown, waiting for the previous to complete before continuing:-

```
[api]# ./01_SecurityGroup-Create.sh
[api]# ./02_SecurityGroupRules-Create.sh
[api]# ./03_SSHKey-Create.sh
[api]# ./04_ElasticIPs-Create.sh
[api]# ./05_ServerTemplates_Import.sh
[api]# ./06_Deployment-Create.sh
[api]# ./07_Servers-Create.sh
[api]# ./08_Servers-Clone.sh
[api]# ./09_Servers-Clone-Rename.sh
[api]# ./10_Server-ElasticIP-Attach-NextInstance.sh
[api]# ./11_DeployLevelInputs-Set.sh
[api]# ./12_ServerDB-Launch.sh
[api]# ./16_ServerLB-Launch.sh
[api]# ./18_ServerAPP-Launch.sh
```

**Note:** Note the numbering sequence change in the last 3 scripts.

## 1.5 Update DNS

**Note:** You can complete this step while you're waiting for the servers to become operational.

You now need to update DNS with the Elastic IP Addresses created when the script `04_ElasticIPs-Create.sh` was run. First you need to find the actual value of the Elastic IPs that you have been assigned

13. Run the following command to retrieve the Elastic IP Addresses IDs, returned when the script `04_ElasticIPs-Create.sh` was invoked

```
[api]# grep Location output/04_ElasticIPs-Create.sh.out
Location: /api/clouds/1/ip_addresses/1SFPJ9HH48PF9
Location: /api/clouds/1/ip_addresses/B50R3SRMABS0D
```

14. Issue an API request to each of these HREFs

```
[api]# mycurl -X GET https://us-
3.rightscale.com/api/clouds/1/ip_addresses/1SFPJ9HH48PF9.xml
<?xml version="1.0" encoding="UTF-8"?>
</ip_addresses>
  <ip_address>
    <address>54.228.197.213</address>
    <created_at>2013/03/25 11:42:05 +0000</created_at>
    <updated_at>2013/03/25 11:42:05 +0000</updated_at>
    <links>
      <link href="/api/clouds/2/ip_addresses/1SFPJ9HH48PF9" rel="self"/>
    </links>
    <name>jaf250313 EIP1</name>
  </ip_address>
</ip_addresses>
```

```
[api]# mycurl -X GET https://us-
3.rightscale.com/api/clouds/1/ip_addresses/B50R3SRMABS0D.xml
</ip_addresses>
  <ip_address>
    <address>54.228.197.215</address>
    <created_at>2013/03/25 11:42:06 +0000</created_at>
    <updated_at>2013/03/25 11:42:06 +0000</updated_at>
    <links>
      <link href="/api/clouds/2/ip_addresses/B50R3SRMABS0D" rel="self"/>
    </links>
    <name>jaf250313 EIP2</name>
  </ip_address>
</ip_addresses>
```

Here you can see the actual IP Address values for **myname EIP1** and **myname EIP2**

15. Alternatively you can browse to '**Cloud > Your Cloud > IP Addresses**' in the Dashboard and you should see the Elastic IPs have been created
16. Browse again to the website <http://www.dnsmadeeasy.com> and click on the **rightscaletraining.com** domain link in the middle of the page under 'Recently Updated Domains'  
Username:**rightscaletraining**  
Password:\*\*\*\*\*
17. Click on each of the A Record 1 for **myname-www.rightscaletraining.com** and give it the IP address assigned to **myname EIP1**
18. Click on each of the A Record 2 for **myname-www.rightscaletraining.com** and give it the IP address assigned to **myname EIP2**

So you should have two seemingly identical DNS entries, but with different IP Addresses, like this:-

Record myname-www.rightscaletraining.com. will point to 54.225.101.126

Name	myname-www	.rightscaletraining.com.
IP	54.225.101.126	*
Dynamic DNS	<input type="checkbox"/>	
TTL	1800	*

Buttons: ? Help, Submit, Close

Record myname-www.rightscaletraining.com. will point to 54.225.101.34

Name	myname-www	.rightscaletraining.com.
IP	54.225.101.34	*
Dynamic DNS	<input type="checkbox"/>	
TTL	1800	*

Buttons: ? Help, Submit, Close

## 1.6 Configure Master Database

19. Once the Database Server is Operational, run the following script

```
[api]# ./13_ServerDB-RunRecipe-LoadDB.sh
```

This bash script makes an API Request that invokes execution of the Chef Recipe `db::do_dump_import`. Monitor Audit Entries for this Recipe to complete before continuing

20. Once the previous script has completed successfully run the following script

```
[api]# ./14_ServerDB-RunRecipe-MakeMaster.sh
```

This bash script makes an API Request that invokes execution of the Chef Recipe `db::do_init_and_become_master`. Monitor Audit Entries for this Recipe to complete before continuing

21. Once all servers are Operational, and these two recipes have completed, you should have a fully functional 3-Tier Deployment

22. Browse to your Load Balancer FQDN, e.g. <http://myname-www.rightscaletraining.com>

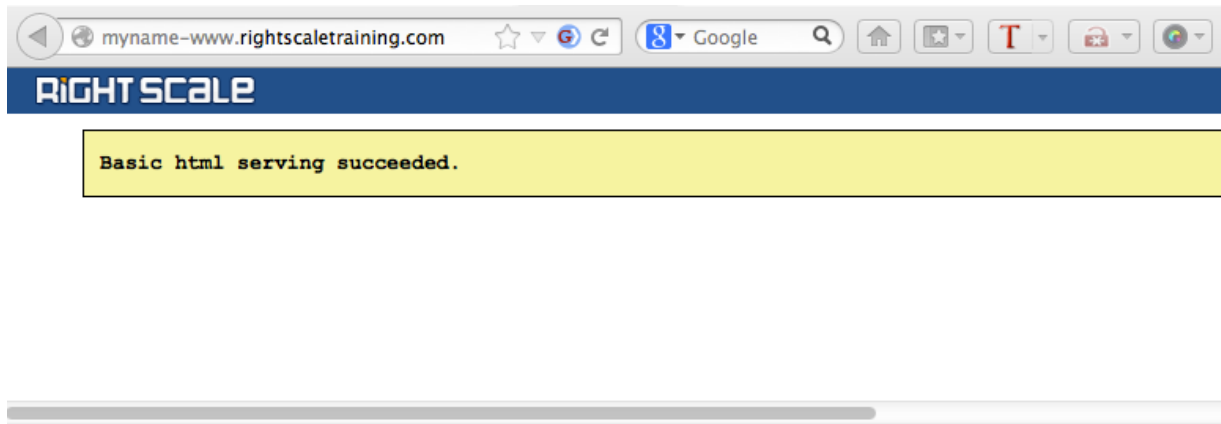
## 1.7 Test The Application

In this section you will test your 3-Tier application to ensure its functioning end-to-end. You will also use the API to run an Operational Script across all Application Servers to update the actual application.

23. Open browser and browse to your Load Balancers URL you registered in DNS, e.g.

<http://myname-www.rightscaletraining.com/>

You should see a very basic web page



This web application is not very meaningful, so you will run an operational script on the Application Servers to install a new application.



## 1.8 Run Operational Script to Update Application

See: <http://reference.rightscale.com/api1.5/resources/ResourcePublications.html>

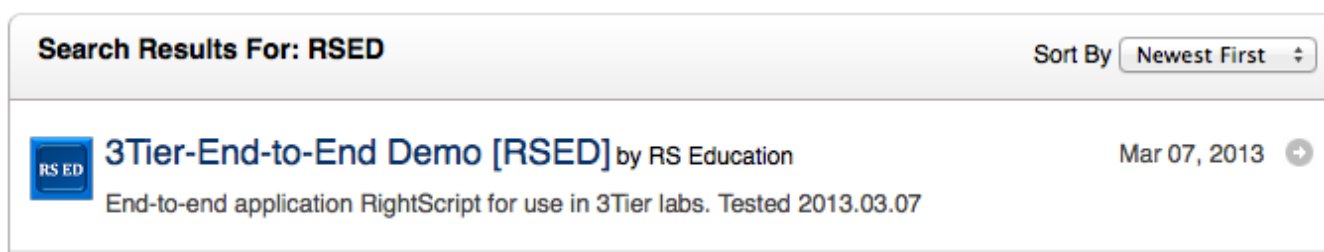
Now you will update the application. To do this you will import the and run the Operational RightScript **3Tier-End-to-End Demo [RSED]** on both Application Servers simultaneously using an API Request.

### 1.8.1 (Optional) Determine RightScript Object ID

**PLEASE NOTE** that the script you will to import the RightScript already has the correct ID values for this lab hard coded, so you do not need to complete the following two steps.

However, you can find the RightScript IDs as follows:-

24. Navigate to '**Design > MultiCloud Marketplace > RightScripts**' in the RightScale Dashboard, and search for these the RightScript **3Tier-End-to-End Demo [RSED]**



25. Hover over the 'Import' button and make a note of the 'Object ID' in the URL displayed at the bottom of the screen, e.g.

[https://us-3.rightscale.com/library/right\\_scripts/3Tier-End-to-End-Demo-RSED-/48616](https://us-3.rightscale.com/library/right_scripts/3Tier-End-to-End-Demo-RSED-/48616)

### 1.8.2 Import RightScript

For simplicity the bash script used to import the RightScripts ([19\\_RightScript-APP-Import.sh](#)) already has the correct ID in place, so there is no need to edit it.

26. View the bash script [19\\_RightScript-APP-Import.sh](#)

```
[api]# cat 19_RightScript-APP-Import.sh
#!/bin/bash -e

RIGHTSCRIPTID="48616"

echo "***Importing RightScript '3Tier-End-to-End Demo [RSED]'***"
curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X POST \
https://us-3.rightscale.com/api/publications/$RIGHTSCRIPTID/import
```

27. Take a few minutes to read through this script

28. Run the script

```
[api]# ./19_RightScript-APP-Import.sh
HTTP/1.1 201 Created
Server: nginx/1.0.14
Date: Wed, 13 Mar 2013 17:27:49 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Status: 201 Created
Location: /api/right_scripts/426335003
X-Runtime: 433
X-Request-Uuid: 2e5a9bea01544a5b86a9597b18b973e4
Set-Cookie:
Cache-Control: no-cache
```

### 1.8.3 Run Script on Application Servers

You will now run the RightScript **3Tier-End-to-End Demo [RSED]** to install the new application.

The **multi\_run\_executable** method in the **Instance** Resource script is used to run a script across multiple servers in a Deployment. You will use this method, but in this case you will filter the Servers on Name, i.e. only run in on those with “**APP**” in the name.

29. (Optional) You can check your filter is correct searching your deployment for Servers, and filtering on those with **APP** in the name, as follows (replacing your own Deployment ID for **350944003** shown in the example).

```
[api]# mycurl -d filter[]="name==APP" -X GET \
https://us-3.rightscale.com/api/deployments/350944003/servers.xml
```

This should only return your Application servers **myname App Server 1** and **myname App Server 2**.

30. View the script **20\_ServerAPPs-RunScript-E2EDemo.sh**.

```
[api]# cat 20_ServerAPPs-RunScript-E2EDemo.sh
```

The script pulls in some parameters you set earlier in the **LabInfo** file, as well as the output from the **19\_RightScript-APP-Import.sh** file.

```
#!/bin/bash -e

MYNAME=`grep MYNAME LabInfo | cut -c 8-|tr -d '\r'`
CLOUD=`grep CLOUD LabInfo | cut -c 7-|tr -d '\r'`
DBSCHEMA=`grep DBSCHEMA LabInfo | cut -c 10-|tr -d '\r'`
DBFQDN=`grep DBFQDN LabInfo | cut -c 8-|tr -d '\r'`
E2ESCRIPID=`grep Location output/19_RightScript-APP-Import.sh.out|cut -c 30-|tr -d '\r'`

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -d filter[]="name==APP" -X POST \
-d right_script_href="/api/right_scripts/$E2ESCRIPID" \
-d inputs[][name]="DBADMIN_PASSWORD" \
-d inputs[][value]="text:adminpassword" \
-d inputs[][name]="DBADMIN_USER" \
-d inputs[][value]="text:adminuser" \
-d inputs[][name]="APPLICATION" \
-d inputs[][value]="text:myapp" \
-d inputs[][name]="MASTER_DB_DNSNAME" \
-d inputs[][value]="text:$DBFQDN" \
-d inputs[][name]="INSTANCE" \
-d inputs[][value]="env:INSTANCE_ID" \
-d inputs[][name]="DB_SCHEMA_NAME" \
-d inputs[][value]="text:$DBSCHEMA" \
https://us-3.rightscale.com/api/clouds/$CLOUD/instances/multi_run_executable \
| tee output/${0##*/}.out
```

31. Take a few minutes to read through this script

32. Run the script

```
[api]# ./20_ServerAPPs-RunScript-E2EDemo.sh
HTTP/1.1 202 Accepted
Server: nginx/1.0.14
Date: Thu, 14 Mar 2013 11:35:02 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Status: 202 Accepted
Location: /api/clouds/1/instances/62MIVJOJFVQDM/live/tasks/ae-159705715003
X-Runtime: 2097
X-Request-Uuid: 56de1784397f4e019d4c46bea4daac8b
Set-Cookie:
Cache-Control: no-cache
```

33. You can monitor progress by doing a **GET** on the Audit Entries href returned in the **Location** line of the API Response, i.e.

```
Location: /api/clouds/1/instances/62MIVJOJFVQDM/live/tasks/ae-159705715003
```

for example

```
[api]# mycurl -X GET https://us-
3.rightscale.com/api/clouds/1/instances/62MIVJOJFVQDM/live/tasks/ae-159705715003.xml
```

34. Wait for the RightScript to complete before continuing

## 1.9 Testing The New Application

35. Open browser and browse to your Load Balancers URL you registered in DNS, e.g.

You should see the '**RightScale Education: End-to-end Application**' test page.

myname-www.rightscaletraining.com

RightScale Education: End-to-end Application

Server Information:

Instance Resource ID:	i-479d6035
Availability Zone:	us-east-1d

Verification of PHP-MySQL Connection

(1) Verified that PHP is working.
(2) Connected to the database server myhandle-masterdb.rightscaletraining.com. Connect ID= Resource id #2
(3) Selected the database mynameSchema. Select ID= 1
(4) Value returned from mysql_select. Resource id #3

Database Contents and Controls:

ID	Country	Capital
1	France	Paris
2	Japan	Tokyo
3	Spain	Madrid

Add a Record

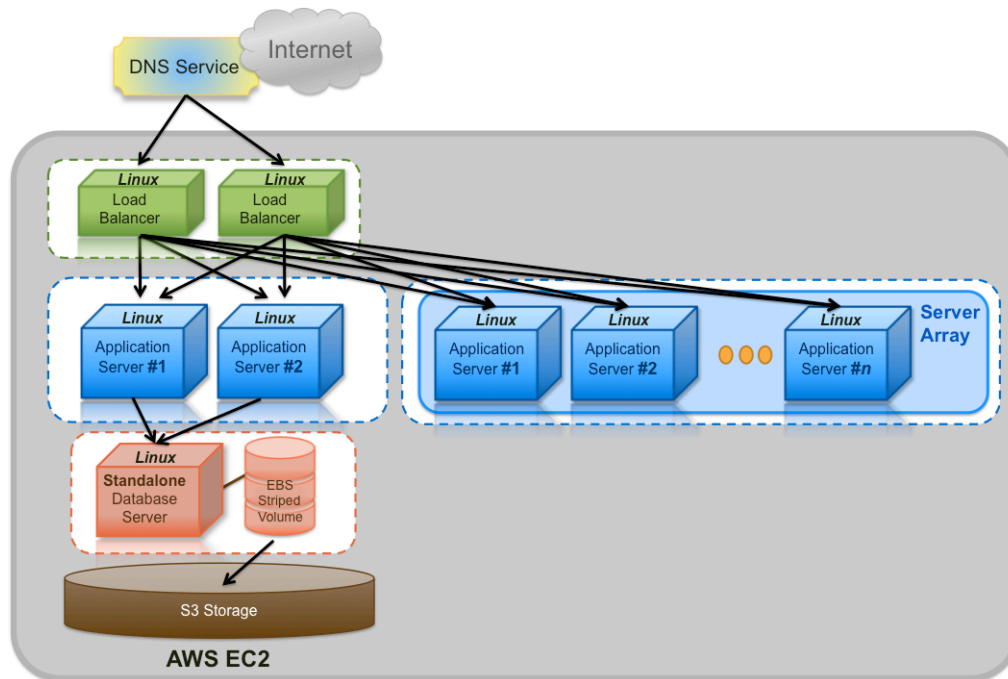
Country:

Capital:

Remove a Record

ID:

## Section 2. Server Arrays



### 2.1 Create Array

See: <http://reference.rightscale.com/api1.5/resources/ResourceServerArrays.html#create>

Now you will add an array of Application Servers to your deployment. The servers in this array will be based of the same ServerTemplate as the existing two Application Servers

36. View the `script 22_Array-Create.sh`

**Note:** For simplicity, a number of the required values when creating a Server Array (e.g. `min_count`, `max_count`, `resize_calm_time`, etc) have been prepopulated in this script. Feel free to edit these values if you wish.

```
[api]# cat 22_Array-Create.sh
#!/bin/bash -e

CLOUD=`grep CLOUD LabInfo | cut -c 7-|tr -d '\r'`
DEPLOYMENT=`grep Location output/06_Deployment-Create.sh.out |cut -c 28-|tr -d '\r'`
APP_ST=`grep Location output/05_ServerTemplates_Import.sh.out|cut -c 33-|tr -d '\r'|sed -n 2p`
SG=`grep Location output/01_SecurityGroup-Create.sh.out |cut -c 41-|tr -d '\r'`
SSH=`grep Location output/03_SSHKey-Create.sh.out |cut -c 34-|tr -d '\r'`

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X POST \
-d server_array[name]=my_array_server \
-d server_array[description]=my_app_server_description \
-d server_array[array_type]=alert \
-d server_array[state]=disabled \
-d server_array[instance][server_template_href]=/api/server_templates/$APP_ST \
-d server_array[instance][cloud_href]=/api/clouds/$CLOUD \
-d server_array[instance][security_group_hrefs]=/api/clouds/$CLOUD/security_groups/$SG \
-d server_array[instance][ssh_key_href]=/api/clouds/$CLOUD/ssh_keys/$SSH \
-d server_array[elasticity_params][alert_specific_params][decision_threshold]=51 \
-d server_array[elasticity_params][bounds][min_count]=2 \
-d server_array[elasticity_params][bounds][max_count]=3 \
-d server_array[elasticity_params][pacing][resize_calm_time]=5 \
-d server_array[elasticity_params][pacing][resize_down_by]=1 \
-d server_array[elasticity_params][pacing][resize_up_by]=1 \
https://us-3.rightscale.com/api/deployments/$DEPLOYMENT/server_arrays \
| tee output/${0##*/}.out
```

**Note:** The documentation states that the data type for the mandatory parameters `server_array[instance][security_group_hrefs]` is of type array.

server_array[instance][security_group_hrefs]	no	Array	*	*	no	The hrefs of the Security Groups.
--	----	-------	---	---	----	-----------------------------------

This is represented in the code by the following `[ ]`, i.e.

```
-d server_array[instance][security_group_hrefs][]=
```

37. Run the script

```
[api]# ./22_Array-Create.sh
HTTP/1.1 201 Created
Server: nginx/1.0.14
Date: Mon, 25 Mar 2013 18:02:25 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Status: 201 Created
Location: /api/server_arrays/219528003
X-Runtime: 884
X-Request-Uuid: 0391576d84ab40a2bd97623360b9fad0
Set-Cookie: rs_gbl=eNotkFtvvgjAARv9Ln21CL0A12QNi8JbpAIvAi-
lqcU4dhlZkGP77MNnjl3znPJwnEMADdw0JGIGDBt5zGKoGHqaUkn4EjAQeIg7FhNrUHoHTYfg7pXVAwn
Xg2HYEREhhyISkkH20FaMI2dhSg8-
of5Yi98UOehCzNHR39ngqV3kw5ZkUJDmGGV_HU1aV28mZfOX1jEN_HzSZ5rfE0d2yy25qQ9PiUURFek5
xpfjyyhv4UAu4xFsd-
XrmFpY5b3C048G8i9tVfSG0newvmYkT_ZH6iyMLbzKvTIkj3PU6jtnJyaoTui2DOaw0W04MbW_kL_d7
Nqs3_XbK0n7SiKkr04_BniEIBf3_R_rnFzs; domain=.rightscale.com; path=/; HttpOnly
Cache-Control: no-cache
```



## 2.2 Enable the Array

In this example you will run a script, which will enable the Array.

See: <http://reference.rightscale.com/api1.5/resources/ResourceServerArrays.html#update>

38. View the script `23_Array-Enable.sh`

```
[api]# cat 23_Array-Enable.sh
#!/bin/bash -e

ARRAY=`grep Location output/22_Array-Create.sh.out |cut -c 30-|tr -d '\r'`

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X PUT \
-d server_array[state]=enabled \
https://us-3.rightscale.com/api/server_arrays/$ARRAY \
| tee output/${0##*/}.out
```

39. Take a few minutes to read through this script

40. Run the script

```
[api]# ./23_Array-Enable.sh
...
Status: 204 No Content
...
```

## Section 3. Monitoring Servers

### 3.1 Select a Server To Monitor

41. Run the following command to view all operational instances in the account (replace your **Cloud ID** and **myname** as appropriate)

```
[api]# mycurl -d filter[]="state==operational" -d filter[]="name==myname" -X GET  
https://us-3.rightscale.com/api/clouds/1/instances.xml
```

For each operational server you will see a line similar to the following

```
<link href="/api/clouds/1/instances/BRM7PNUAVBJFC/monitoring_metrics"  
rel="monitoring_metrics"/>
```

You'll use the href to access the monitoring data for the server. Select a particular server you wish to monitor.

### 3.2 View Metrics for Server

42. Run the following command (replacing the href from above) to view all metrics for your server

```
[api]# mycurl -X GET https://us-
3.rightscale.com/api/clouds/1/instances/BRM7PNUAVBJFC/monitoring_metrics.xml
```

You may see ~100 metric returned. So now you'll drill down to see a specific metric.

**Note:** In this example you will use the `cpu-0/cpu_idle` metric, though you may choose another one if you like

43. Run the same command, but this time introduce a filter to select the desired output

```
[api]# mycurl \
-d filter[]="plugin==cpu-0" \
-d filter[]="view==cpu-idle" \
-X GET https://us-
3.rightscale.com/api/clouds/1/instances/CP2LAIHFRV0MJ/monitoring_metrics.xml
<?xml version="1.0" encoding="UTF-8"?>
<monitoring_metrics>
  <monitoring_metric>
    <actions></actions>
    <plugin>cpu-0</plugin>
    <view>cpu-idle</view>
    <graph_href>/sketchy1-175/hosts/03-4NFRPVSTN6S74/plugins/cpu-0/views/cpu-
idle.png?period=day&tok=pILW5Aa-2dO6-
GozbncOyNg&tz=Europe%2FBelfast&size=small</graph_href>
    <links>
      <link href="/api/clouds/1/instances/CP2LAIHFRV0MJ/monitoring_metrics/cpu-
0:cpu-idle" rel="self"/>
      <link href="/api/clouds/1/instances/CP2LAIHFRV0MJ/monitoring_metrics/cpu-
0:cpu-idle/data" rel="data"/>
    </links>
  </monitoring_metric>
</monitoring_metrics>
```

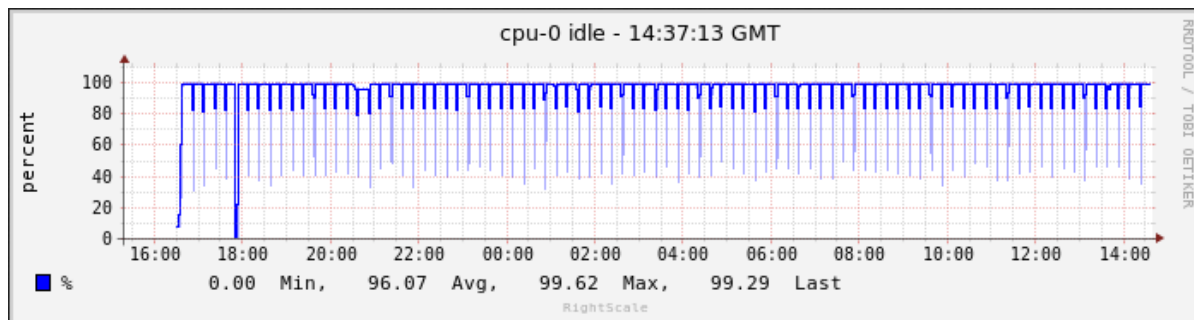
44. Note specifically the data href and the graph\_href

45. For the metric data, make an API call to the data href including `start` and `end` data points (represent an integer number of seconds from current time), as follows

```
[api]# mycurl -d start="-100" -d end="-5" -X GET https://us-3.rightscale.com
/api/clouds/1/instances/CP2LAIHFRV0MJ/monitoring_metrics/cpu-0:cpu-
idle/data.xml
<?xml version="1.0" encoding="UTF-8"?>
<monitoring_metric_data>
  <start>2013-03-21 16:19:40</start>
  <end>2013-03-21 16:21:40</end>
  <variables_data>
    <variable_data>
      <variable>value</variable>
      <points>
        <point>99.1675</point>
        <point>99.245</point>
        <point>99.345</point>
        <point>99.16</point>
        <point>36.545</point>
      </points>
    </variable_data>
  </variables_data>
  <links>
    <link href="/api/clouds/1/instances/CP2LAIHFRV0MJ/monitoring_metrics/cpu-
0:cpu-idle/data" rel="self"/>
  </links>
  <actions/>
</monitoring_metric_data>
```

46. To view the graph for this metric, open a browser and browse to the [graph\\_href](#), e.g.

<https://us-3.rightscale.com/sketchy1-175/hosts/03-4NFRPVSTN6S74/plugins/cpu-0/views/cpu-idle.png?period=day&tok=pILW5Aa-2dO6-GozbncOyNg&tz=Europe%2FBelfast&size=small>



## Section 4. Using Tags

### 4.1 Find all App Servers with a Certain Tag

See: [http://reference.rightscale.com/api1.5/resources/ResourceTags.html#by\\_tag](http://reference.rightscale.com/api1.5/resources/ResourceTags.html#by_tag)

In this section you will find all the Application Servers, both in the Deployment and in the array, using the tag `appserver:active=true`.

47. View the Script `24_Servers-ByTag-List.sh` which you will use to list servers all load balancers' i.e. those with the tag `loadbalancer:default=lb`.

```
#!/bin/bash -e

SEARCHTAG="loadbalancer:default=lb"

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X POST \
-d resource_type=instances \
-d tags[]=$SEARCHTAG \
https://us-3.rightscale.com/api/tags/by_tag.xml \
| tee output/${0##*/}.out
```

Notice the `tags` syntax includes “[ ]” (`-d tags [ ]`), while `resource_type` does not. This is because `tags` is an array, while `resource_type` is of type text.

48. Now run the script

```
[api]# ./24_Servers-ByTag-List.sh
<?xml version="1.0" encoding="UTF-8"?>
<resource_tags>
  <resource_tag>
    <actions></actions>
    <tags></tags>
    <links>
      <link href="/api/clouds/1/instances/FICM1BVDR9NR3" rel="resource"/>
      <link href="/api/clouds/1/instances/E8L90A1KM8QRJ" rel="resource"/>
    </links>
  </resource_tag>
</resource_tags>
```

## 4.2 Set Custom Tags on Load Balancer Servers

See: [http://reference.rightscale.com/api1.5/resources/ResourceTags.html#multi\\_add](http://reference.rightscale.com/api1.5/resources/ResourceTags.html#multi_add)

In this section you will set the custom tag `testtag:dbserver=yes` on the Database Server.

49. View the Script `24.5_DBServer-Tag-Add.sh` which you will use to add the Machine Tag.

```
#!/bin/bash -e

NEWTAG="testtag:dbserver=yes"
DBSERVERHREF=`grep Location output/07_Servers-Create.sh.out |cut -c 11-|tr -d '\r' |
sed -n 3p`
CURRDBINSTANCE=`curl -s -H X-API-Version:1.5 -b ~/mycookie -X GET https://us-
3.rightscale.com${DBSERVERHREF} |python -mjson.tool | grep instances |sed -n 1p | awk
'{split($2,a,""); print a[2]}' | tr -d '\r'`

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X POST \
-d resource_hrefs[]=$CURRDBINSTANCE \
-d tags[]=$NEWTAG \
https://us-3.rightscale.com/api/tags/multi_add \
| tee output/${0##*/}.out
```

50. Take a few minutes to read the format of this API Request

51. Now run the script

```
[api]# ./24_Servers-ByTag-List.sh
```

52. Now verify in the Dashboard that this Machine Tag has been set

## Section 5. Terminate & Delete All Servers

You will now terminate and delete all servers using the API.

### 5.1 Disable the Array

See: <http://reference.rightscale.com/api1.5/resources/ResourceServerArrays.html#update>

53. View the script `25_Array-Disable.sh` you will use to disable the array.

```
[api]# cat 25_Array-Disable.sh
#!/bin/bash -e

ARRAY=`grep Location output/22_Array-Create.sh.out | cut -c 30- | tr -d '\r'`

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X PUT \
-d server_array[state]=disabled \
https://us-3.rightscale.com/api/server_arrays/$ARRAY \
| tee output/${0##*/}.out
```

54. Now run the script

```
[api]# ./25_Array-Disable.sh
HTTP/1.1 204 No Content
Server: nginx/1.0.14
Date: Tue, 26 Mar 2013 11:35:50 GMT
Connection: keep-alive
Status: 204 No Content
X-Runtime: 82
X-Request-Uuid: a962ead5718047ebbc2524892ba6e77b
Set-Cookie:
Cache-Control: no-cache
```



## 5.2 Terminate All Servers

See:

[http://reference.rightscale.com/api1.5/resources/ResourceInstances.html#multi\\_terminate](http://reference.rightscale.com/api1.5/resources/ResourceInstances.html#multi_terminate)

55. View the script you will use to terminate the servers.

```
[api]# cat 26_TerminateServers-All.sh
#!/bin/bash -ex

CLOUD=`grep CLOUD LabInfo | cut -c 7-|tr -d '\r'`
DEPLOYMENT=`grep Location output/06_Deployment-Create.sh.out |cut -c 11-|tr -d '\r'`

cd /opt/api/3tier

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X POST \
-d filter[]="deployment_href==$DEPLOYMENT" \
https://us-3.rightscale.com/api/clouds/$CLOUD/instances/multi_terminate \
| tee output/${0##*/}.out
```

56. Now run the script

```
[api]# ./26_TerminateServers-All.sh
HTTP/1.1 202 Accepted
Server: nginx/1.0.14
Date: Tue, 26 Mar 2013 11:19:05 GMT
Content-Type: text/html; charset=utf-8
Transfer-Encoding: chunked
Connection: keep-alive
Status: 202 Accepted
Location: /api/clouds/2/instances/DILJ7PFFQ0QHJ/live/tasks/ae-163889399003
X-Runtime: 4284
X-Request-Uuid: 024bdbcc1cfa43e4b104e81b979703d3
Set-Cookie:
Cache-Control: no-cache
```

57. Wait until all servers have terminated before continuing

### 5.3 Delete the Array

See: <http://reference.rightscale.com/api1.5/resources/ResourceServerArrays.html#destroy>

58. View the script `27_Array-Destroy.sh` you will use to delete the array.

```
[api]# cat 27_Array-Destroy.sh
#!/bin/bash -ex

ARRAY=`grep Location output/22_Array-Create.sh.out |cut -c 30-|tr -d '\r'`

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X DELETE \
https://us-3.rightscale.com/api/server_arrays/$ARRAY \
| tee output/${0##*/}.out
```

59. Now run the script

```
[api]# ./27_Array-Destroy.sh
HTTP/1.1 204 No Content
Server: nginx/1.0.14
Date: Tue, 26 Mar 2013 11:48:23 GMT
Connection: keep-alive
Status: 204 No Content
X-Runtime: 391
X-Request-Uuid: 916eedc1eafe48ddaddfd8c7d474d225
Set-Cookie: rs_gbl=eNolkNlugjAARt-l15KUlpawZBf-
RYvAJtYNUatABXSaCVWG8d2H2eWXfOdcnAdQwAOmsTAYgbwB3mMYxRV4yHEc_ByBNgOejamDOCMQj0CV
D3-
oYL5nClqcQtuy7QJZjBfcyhRWiGsCOXEhX1v8sxsG98UOelB39WEdyjA3yWlejOFqRmKWlHGyYlmyUsx
v2903Pfb3adc5JboiMvHjZfq-EXQqRfWRB_f9sjGuMq0WwS2GcGYuE8PKSmyP77N03f9-
ULbYRlWY9uye0a_oqEqXd05iLUmyOq3OgfQbOb5umk-thb8R2pV-JM2h1Pq0S3-
iPlgvdC3eXkm6VxKVZRdzboGHse2i5_MPExtcvw%3D%3D; domain=.rightscale.com; path=/;
HttpOnly
Cache-Control: no-cache
```

## 5.4 Delete all Servers

See: <http://reference.rightscale.com/api1.5/resources/ResourceServers.html#destroy>

Now you will delete all servers in your deployment

60. View the script `28_Servers-Destroy.sh` you will use to delete the Servers.

```
[api]# cat 28_Servers-Destroy.sh
#!/bin/bash -ex

LB1=`grep Location output/07_Servers-Create.sh.out|cut -c 24-|tr -d '\r' | sed -n 1p`
LB2=`grep Location output/08_Servers-Clone.sh.out|cut -c 24-|tr -d '\r' | sed -n 1p`
APP1=`grep Location output/07_Servers-Create.sh.out|cut -c 24-|tr -d '\r' | sed -n 2p`
APP2=`grep Location output/08_Servers-Clone.sh.out|cut -c 24-|tr -d '\r' | sed -n 2p`
DBSERVER=`grep Location output/07_Servers-Create.sh.out |cut -c 24-|tr -d '\r' | sed -n 3p`

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X DELETE \
https://us-3.rightscale.com/api/servers/$LB1 \
| tee output/${0##*/}.out

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X DELETE \
https://us-3.rightscale.com/api/servers/$LB2 \
| tee -a output/${0##*/}.out

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X DELETE \
https://us-3.rightscale.com/api/servers/$APP1 \
| tee -a output/${0##*/}.out

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X DELETE \
https://us-3.rightscale.com/api/servers/$APP2 \
| tee -a output/${0##*/}.out

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X DELETE \
https://us-3.rightscale.com/api/servers/$DBSERVER \
| tee -a output/${0##*/}.out
```

61. Now run the script

```
[api]# ./28_Servers-Destroy.sh
```

## 5.5 Delete Deployment

Finally you will delete the Deployment

62. View the script `29_Deployment-Destroy.sh` you will use to delete the array.

```
[api]# cat 29_Deployment-Destroy.sh
#!/bin/bash -ex

# Delete Deployment

DEPLOYMENT=`grep Location output/06_Deployment-Create.sh.out |cut -c 28-|tr -d
'\r'`

curl -s -i -H X-API-Version:1.5 -b ~/mycookie -X DELETE \
https://us-3.rightscale.com/api/deployments/$DEPLOYMENT \
| tee -a output/${0##*/}.out
```

63. Now run the script

```
[api]# ./28_Deployment-Destroy.sh
HTTP/1.1 204 No Content
Server: nginx/1.0.14
Date: Tue, 26 Mar 2013 12:08:36 GMT
Connection: keep-alive
Status: 204 No Content
X-Runtime: 233
X-Request-Uuid: 0da9d9f611eb44fba3fb001985769659
Set-Cookie:
rs_gbl=eNolkMlugzAABf_F51gyXjBG6iELJTRSRTZKuCBjloS0ISEmBaL8e0E9PunNHOYJJLBBC4cET
EB6B_ZzGfKNbEwpJa8J0ArYBjEpFoIb5gSc0uGPJEoTSyIoTGRAw8gwtEQmoJJEYpEzJBgffDr7Zwkyx
cgOepDzn9nhGNAPCqIkZofS2fkfZbGvuzrPVnTaR9ZNLnjvhNSq87PswjjgnRdjLzbXlYYh97qWiQe7V
ItTGV-Vd9ueHlRu0Lbw9dVJv635Eun3qEbq2G5_d0lTFASTaHPOShX0tPIqv1764Y3pFf2aN-
e2c5u189n4snJht5lHAVPusp15b2OSdkwilaqaiwY2IQbHr9cfrEdd3g%3D%3D;
domain=.rightscale.com; path=/; HttpOnly
Cache-Control: no-cache
```