



ROCKET.CHAT

Technical Assessment Challenge

Matheus Paiva_April2021

Summary

- Deployment Environment 2
- Rocket.chat Installation steps..... 2
- Configure Auto SSL (Caddy service).....2
- Rocket.chat APIs Test Results 3
- Create a new user via an API endpoint..... 3
- Get the room information via an API endpoint..... 4
- Get a list of all user roles in the system via an API endpoint 5
- Integrate Rocket.chat with ELK: 5
- Connect to my Rocket.chat server via Rocket.chat App (Android) 6

Deployment Environment

- Installed a snap of Rocket.chat on Ubuntu 16.04 LTS as per documentation available on <https://docs.rocket.chat>
- Created an Ubuntu VM running on Microsoft Azure platform following the recommended Hardware resources. (Public IP: 23.97.102.57)

Rocket.chat server credentials:

URL : <https://rocketchatmatheuspaiva.brazilsouth.cloudapp.azure.com>

User: mmcp108@gmail.com

Pass: RC\$_17042021

Rocket.chat Installation steps

1- Connected to the Ubuntu VM via terminal (on this case I used putty) and entered the following command line:

```
sudo snap install rocketchat-server
```

2- On my computer, browse to <http://23.97.102.57:3000> to setup Rocket.chat.

Note: On this step I noticed that there was no access to the page, then I added the inbound port role for port 3000 on Azure Networking settings, then I was able to browse to the URL and setup Rocket.chat.

Configure Auto SSL (Caddy service)

Note: As this Ubuntu VM was created on MS Azure platform it provided me a domain to use.

1- Entered the following command line via terminal:

```
sudo snap set rocketchat-server caddy-url=https://rocketchatmatheuspaiva.brazilsouth.cloudapp.azure.com
```

```
sudo snap set rocketchat-server caddy=enable
```

```
sudo snap set rocketchat-server https=enable
```

```
sudo snap run rocketchat-server.initcaddy
```

2- Restarted Rocket.chat and Caddy services:

```
sudo systemctl restart snap.rocketchat-server.rocketchat-server.service
```

```
sudo systemctl restart snap.rocketchat-server.rocketchat-caddy.service
```

In order to proof that configuration was applied, I browsed again to the URL <https://rocketchatmatheuspaiva.brazilsouth.cloudapp.azure.com> and we can see that the padlock icon is being shown in the browser and if we click on that we can see that there is a valid SSL certificate and the server can use HTTPS protocol.

Then, disabled the port 3000 in MS Azure Networking settings.

Rocket.chat APIs Test Results

NOTE: In order to perform the queries/ updates via rest API, firstly you must have to generate a personal access token in your profile page: *Profile -> My Account -> Security -> Personal Access Tokens*

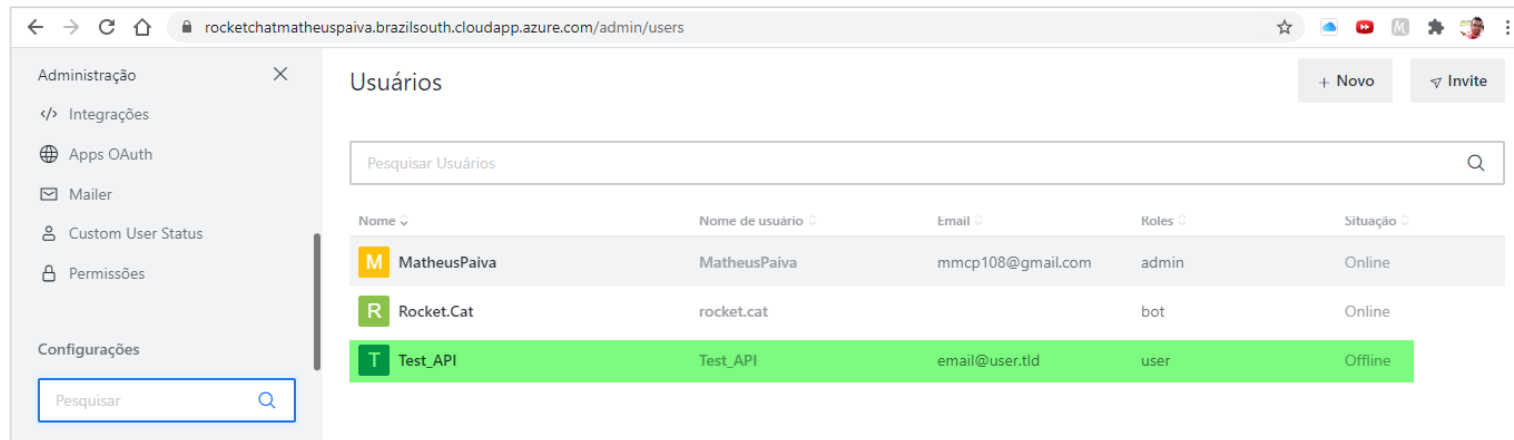
Token: `cOHVSJJODRcZAiElnOQLUBM2LTfHs3x1E3QKGtSf_jz`

User ID: `7xX2r9ttmeLciR8FR`

Create a new user via an API endpoint

```
mmcp108@Linux:~$ curl -H "X-Auth-Token: cOHVSJJODRcZAiElnOQLUBM2LTfHs3x1E3QKGtSf_jz" \
> -H "X-User-Id: 7xX2r9ttmeLciR8FR" \
> -H "Content-type:application/json" \
> http://localhost:3000/api/v1/users.create \
> -d '{"name": "Test_API", "email": "email@user.tld", "password": "Changeme_123", "username": "Test_API"}'
{"user":{"_id":"27PqxshKmXenIdAtQ","createdAt":"2021-04-18T21:14:15.233Z","username":"Test_API","emails":[{"address":"email@user.tld","verified":false}],"type":"user","status":"offline","active":true,"_updatedAt":"2021-04-18T21:14:15.427Z","rooms":["GENERAL"],"roles":["user"],"name":"Test_API","settings":{}},"success":true}mmcp108@Linux:~$
```

Created User
via API



Nome	Nome de usuário	Email	Roles	Situação
M MatheusPaiva	MatheusPaiva	mmcp108@gmail.com	admin	Online
R Rocket.Cat	rocket.cat		bot	Online
T Test_API	Test_API	email@user.tld	user	Offline

Get the room information via an API endpoint

For testing propose only, I created manually a new Room named "Teste" via Rocket.chat web application:

The screenshot shows the Rocket.Chat admin interface. On the left is a sidebar with navigation options like 'Administração', 'Integrações', 'Apps OAuth', 'Mailer', 'Custom User Status', 'Permissões', 'Configurações', 'Analytics', 'Assistente de configuração', 'Atlassian Crowd', 'Blockstack', 'Bots', 'CAS', 'Contas', and 'Criptografia E2E'. The main area is titled 'Salas' (Rooms) and contains a search bar and a table of rooms. The table has columns for 'Nome', 'Tipo', 'Usuários', 'Msgs', 'Padrão', and 'Featured'. Two rooms are listed: '# Teste' (Group, 1 user, 3 msgs, Not Default, Not Featured) and '# general' (Canal, 2 users, 3 msgs, Default, Not Featured). The right sidebar, titled 'Informações da Sala', shows details for the selected room 'Teste', including its name, owner (MatheusPaiva), description ('This is for a test propose'), announcement ('None advt'), and topic ('anyone').

```
mmcp108@Linux: ~  
mmcp108@Linux:~$ curl -H "X-Auth-Token: cOHVSJJODRcZAiElnoQLUBM2LTfHs3x1E3QKGtSf_jz" \\  
> -H "X-User-Id: 7xX2r9ttmeLciR8FR" \\  
> http://localhost:3000/api/v1/rooms.info?roomName=Teste  
{  
  "room": {  
    "_id": "ayGeBZG5pLSzFaTwZ",  
    "name": "Teste",  
    "fname": "Teste",  
    "t": "p",  
    "msgs": 3,  
    "usersCount": 1,  
    "u": {  
      "_id": "7xX2r9ttmeLciR8FR",  
      "username": "MatheusPaiva",  
      "customFields": {},  
      "description": "This is for a test propose",  
      "broadcast": false,  
      "encrypted": false,  
      "ts": "2021-04-18T21:20:51.868Z",  
      "ro": true,  
      "default": false,  
      "_updatedAt": "2021-04-18T21:24:46.822Z",  
      "announcement": "None advt",  
      "topic": "anyone",  
      "success": true  
    }  
  }  
}
```

Get a list of all user roles in the system via an API endpoint

```
mmcp108@Linux: ~$ curl -H "X-Auth-Token: COHVSJJODRczAiElnoQLUBM2LTfHs3x1E3QKgtSf_jz" \
-H "X-User-Id: 7xX2r9ttmeLclR8FR" \
http://localhost:3000/api/v1/roles.list
{"roles":[{"_id":"admin","description":"Admin","mandatory2fa":false,"name":"admin","protected":true,"scope":"Users"}, {"_id":"moderator","description":"Moderator","mandatory2fa":false,"name":"moderator","protected":true,"scope":"Subscriptions"}, {"_id":"leader","description":"Leader","mandatory2fa":false,"name":"leader","protected":true,"scope":"Subscriptions"}, {"_id":"owner","description":"Owner","mandatory2fa":false,"name":"owner","protected":true,"scope":"Subscriptions"}, {"_id":"user","description":"","mandatory2fa":false,"name":"user","protected":true,"scope":"Users"}, {"_id":"bot","description":"","mandatory2fa":false,"name":"bot","protected":true,"scope":"Users"}, {"_id":"app","description":"","mandatory2fa":false,"name":"app","protected":true,"scope":"Users"}, {"_id":"guest","description":"","mandatory2fa":false,"name":"guest","protected":true,"scope":"Users"}, {"_id":"anonymous","description":"","mandatory2fa":false,"name":"anonymous","protected":true,"scope":"Users"}, {"_id":"livechat-agent","description":"Livechat Agent","mandatory2fa":false,"name":"livechat-agent","protected":true,"scope":"Users"}, {"_id":"livechat-manager","description":"Livechat Manager","mandatory2fa":false,"name":"livechat-manager","protected":true,"scope":"Users"}],"success":true}mmcp108@Linux: ~$
```

Integrate Rocket.chat with ELK:

In order to have an observability of the Rocket.chat server, I created a trial account on Elastic Cloud.

After that, I downloaded the heartbeat monitoring and installed the monitoring agent on my VM.

Configured the heartbeat settings file (heartbeat.yml) to point to the URL of my Rocket.chat server.

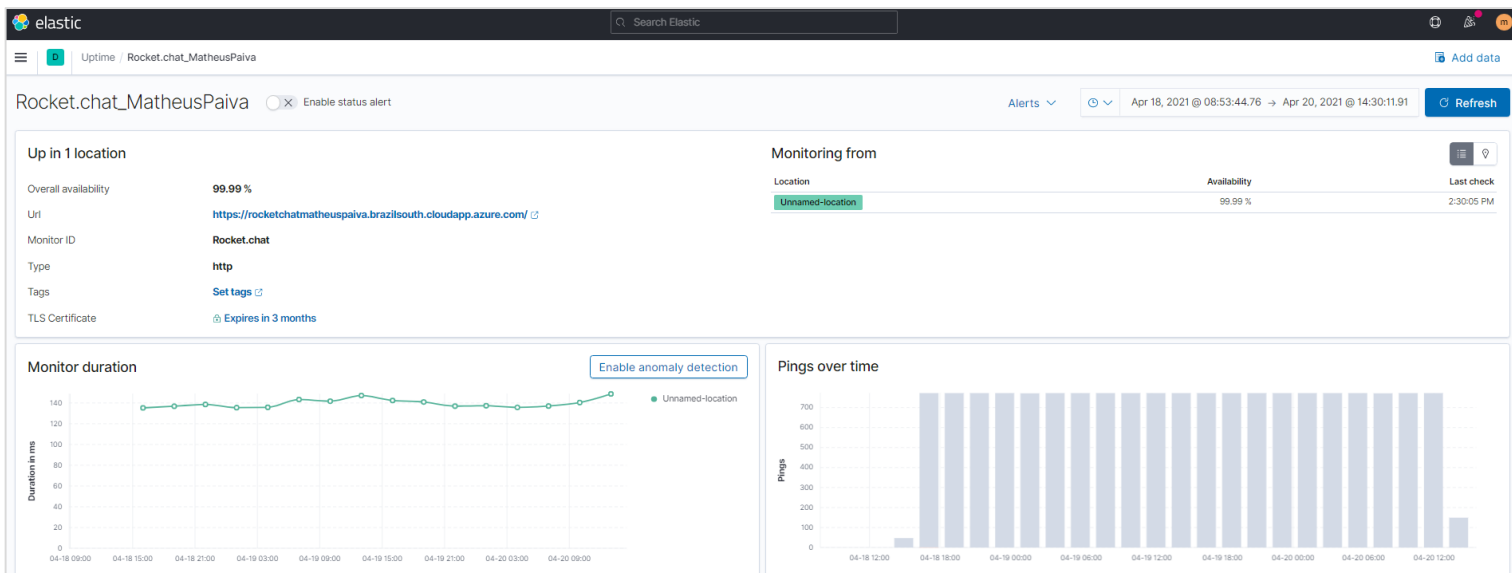
The heartbeat monitoring in ELK is now available and now I can monitor the Uptime of my Rocket.chat server.

https://i-o-optimized-deployment-2e5e8c.kb.us-west1.gcp.cloud.es.io:9243/app/uptime/monitor/Um9ja2V0LmNoYXQ=

Login as Elastic Search user

User: Test

Password: RC\$_17042021



Connect to my Rocket.chat server via Rocket.chat App (Android)

