Entrega 5 - Escenario y Pruebas de Estrés API REST y Batch

A. Plan de pruebas

- Parámetros de configuración:

Se usan instancias de (micro)servicios Python, desplegadas en App Engine modo flexible, no se entrega una configuración de empleo de hardware virtual (configuración por defecto), sólo se restringe el número de instancias a 1 siguiendo las instrucciones recibidas. Ver Documento de Arquitectura en la wiki para mayor detalle a nivel de arquitectura.

Se usa bucket de Cloud Storage para almacenar todos los archivos de audio a procesar y procesados, con las siguientes características:

- Location type: “Region”, us-central1 (Iowa); reduce latencia a cambio de reducir disponibilidad y replicación
- Storage class: Standard
- Access control: Fine-grained, se marca “Enforce public access prevention on this bucket”

Se usa Cloud Pub/Sub con tiempo de Acknowledge de mensajes de 600s (10 minutos). Se eliminan los mensajes exitosamente procesados/ACK.

Se usa base de datos PostgreSQL mediante Cloud SQL, 2 vCPUs, 8GB de RAM, disco SSD de 10GB

- Entorno de prueba:

Se arma una VM para lanzar las pruebas con Locust. Para reducir sesgos derivados de un trayecto de red grande, se ubica dicha VM en el mismo proyecto GCP de los componentes de la solución en nube. Esta máquina es del tipo N1 (f1-micro), 1 vCPU, 0.614 GiB RAM.

- Escenarios de prueba:

  Para la actividad se definen 2 escenario:

  1. Se desea probar cuál es la máxima, cantidad de requests HTTP por minuto que soporta la aplicación web con usuarios (al menos un usuario) que cuenten con 30 archivos disponibles.
  2. Se desea probar cuál es la máxima cantidad de archivos que pueden ser procesados por minuto en la aplicación local. Se usará una herramienta escrita en Python basada en Locust, ya que dio problema Apache Bench con requests tipo POST con archivo de audio.

Se usará una herramienta escrita en Python basada en Locust, dando continuidad a pruebas anteriores. Se dispara desde la instancia descrita en la sección Entorno de prueba.

- Criterios de aceptación:

  Los criterios de aceptación se definen según cada escenario de prueba

  1. El tiempo de respuesta promedio de la aplicación (carga de archivo) debe ser de máximo 1.500 ms, es decir 1.5s, y con porcentaje de errores menor a 1%. Si este tiempo no se cumple, o si se generan más de 1% de errores en los requests disparados se concluye que el sistema NO soporta la cantidad de requests de la prueba.
  2. Se lanzarán archivos de audio de peso mínimo de 5MB para cargar al sistema hasta llegar al punto de quiebre, que es que un archivo cargado se demore 10 minutos en ser procesado por el encolador (según la base de datos PostgreSQL).

B. Análisis de capacidad

El análisis de capacidad mediante pruebas de estrés se realizará mediante los 2 escenarios de pruebas mencionados anteriormente. Se ejecuta la prueba de estrés con el script de Python basado en Locust desde VM en cloud. Más información, ver repositorio de herramienta Locust:

https://github.com/lsolier/load-testing

- Escenario 1:

Archivo de configuración de prueba:

La prueba consiste en enviar 30 requests simultáneas con una tasa de nacimiento de requests de 30 por segundo (para arrancar con carga máxima); se ejecutará la prueba por un lapso de 1 minuto apuntando al IP que expone el servicio Google App Engine (GAE en adelante) del API REST.

```
luis_sajami@load-testing:~/load-testing$ cat locust.conf
locustfile = locustfile.py # file to run
headless = true # to run with the locust UI
host = https://devopslsolier.uc.r.appspot.com/ # base url for the domains
users = 30 # amout of users that will run
master = false # to set master/worker state
spawn-rate = 30 # spawn rate per second
run-time = 1m # load testing  runtime
luis_sajami@load-testing:~/load-testing$
```

```
[2022-12-06 04:42:49,261] load-testing/INFO/locust.main: Time limit reached. Stopping Locust.
[2022-12-06 04:42:49,262] load-testing/INFO/locust.runners: Stopping 30 users
[2022-12-06 04:42:49,305] load-testing/INFO/locust.runners: 30 Users have been stopped, 0 still running
[2022-12-06 04:42:49,306] load-testing/INFO/locust.main: Running teardowns...
[2022-12-06 04:42:49,641] load-testing/INFO/locust.main: Shutting down (exit code 0), bye.
[2022-12-06 04:42:49,641] load-testing/INFO/locust.main: Cleaning up runner...
 Name                                                          # reqs      # fails  |     Avg     Min     Max  Median  |   req/s failures/s
--------------------------------------------------------------------------------------------------------------------------------------------
 POST //api/auth/login                                            93     0(0.00%)  |    4026      93   10667    2700  |    3.17    0.00
 POST //api/tasks                                                 74     0(0.00%)  |    5431     533   11091    4800  |    2.52    0.00
--------------------------------------------------------------------------------------------------------------------------------------------
 Aggregated                                                      167     0(0.00%)  |    4649      93   11091    4000  |    5.70    0.00

Response time percentiles (approximated)
 Type     Name                                                             50%    66%    75%    80%    90%    95%    98%    99%  99.9% 99.99%   100% # reqs
--------|----------------------------------------------------------------|---------|------|------|------|------|------|------|------|------|------|------|------|
 POST     //api/auth/login                                               2700   5600   6700   7400   8700   9700  10000  11000  11000  11000  11000     93
 POST     //api/tasks                                                    5000   6600   8400   9200   9900  10000  11000  11000  11000  11000  11000     74
--------|----------------------------------------------------------------|---------|------|------|------|------|------|------|------|------|------|------|------|
 None     Aggregated                                                    4000   6000   7400   8100   9500  10000  11000  11000  11000  11000  11000    167
```

Rows 25 CSV JSON CSV JSON

id	date_created	date_updated	format_input	format_output	path_input	path_output	status	folder	file_name	id
3527	2022-12-06 03:59:42.957561	2022-12-06 04:02:01.028695	mp3	ogg	2/61d23e83-b0...	2/61d23e83-b0...	processed	61d23e83-b0e8...	test2	2
3446	2022-12-06 03:59:19.057951	2022-12-06 04:02:54.043134	mp3	ogg	2/b5fd9940-76...	2/b5fd9940-76...	processed	b5fd9940-76d9-...	test2	2
3402	2022-12-06 03:59:03.636864	2022-12-06 04:05:51.833496	mp3	ogg	2/084fd615-d8f...	2/084fd615-d8f...	processed	084fd615-d8f4-...	test2	2
3522	2022-12-06 03:59:41.558136	2022-12-06 04:06:29.64384	mp3	ogg	2/ab2d45f4-a1b...	2/e84f6265-10...	processed	ab2d45f4-a1b3-...	test2	2
3427	2022-12-06 03:59:12.482679	2022-12-06 04:06:29.671929	mp3	ogg	2/ab0f9e5a-13b...	2/ab0f9e5a-13b...	processed	ab0f9e5a-13ba-...	test2	2
3496	2022-12-06 03:59:33.747965	2022-12-06 04:07:27.790171	mp3	ogg	2/82eadce3-10...	2/82eadce3-10...	processed	82eadce3-1064-...	test2	2
3380	2022-12-06 03:58:55.126939	2022-12-06 04:09:09.861044	mp3	ogg	2/10ede926-92...	2/10ede926-92...	processed	10ede926-92bb...	test2	2
3542	2022-12-06 03:59:48.199268	2022-12-06 04:10:50.408572	mp3	ogg	2/06a12996-5f...	2/06a12996-5f...	processed	06a12996-5f6a-...	test2	2

Conclusión:

Después de ejecutar una carga de 30 usuario en paralelo y generar 167 request durante 1 minuto, 93 request hacia el enpoint login y 74 request hacia el endpoint task.

Se evidencio que el rendimiento del worker se vio degradado logrando solo realizar la conversión de 8 archivos en 10 minutos, comparado con las pruebas de carga en compute engine su rendimiento no es el deseable.

- Escenario 2:

Archivo de configuración de prueba:

La prueba consiste en enviar 30 requests simultáneas con una tasa de nacimiento de requests de 30 por segundo (para arrancar con carga máxima); se ejecutará la prueba por un lapso de 1 minuto apuntando al IP que expone el servicio GAE del API REST.

```
luis_sajami@load-testing:~/load-testing$ cat locust.conf
locustfile = locustfile.py # file to run
headless = true # to run with the locust UI
host = https://devopslsolier.uc.r.appspot.com/ # base url for the domains
users = 30 # amout of users that will run
master = false # to set master/worker state
spawn-rate = 30 # spawn rate per second
run-time = 1m # load testing  runtime
luis_sajami@load-testing:~/load-testing$
```

```
[2022-12-06 04:42:49,261] load-testing/INFO/locust.main: Time limit reached. Stopping Locust.
[2022-12-06 04:42:49,262] load-testing/INFO/locust.runners: Stopping 30 users
[2022-12-06 04:42:49,305] load-testing/INFO/locust.runners: 30 Users have been stopped, 0 still running
[2022-12-06 04:42:49,306] load-testing/INFO/locust.main: Running teardowns...
[2022-12-06 04:42:49,641] load-testing/INFO/locust.main: Shutting down (exit code 0), bye.
[2022-12-06 04:42:49,641] load-testing/INFO/locust.main: Cleaning up runner...
 Name                                                          # reqs      # fails  |     Avg     Min     Max  Median  |   req/s failures/s
--------------------------------------------------------------------------------------------------------------------------------------------
 POST //api/auth/login                                            93     0(0.00%)  |    4026      93   10667    2700  |    3.17    0.00
 POST //api/tasks                                                 74     0(0.00%)  |    5431     533   11091    4800  |    2.52    0.00
--------------------------------------------------------------------------------------------------------------------------------------------
 Aggregated                                                      167     0(0.00%)  |    4649      93   11091    4000  |    5.70    0.00

Response time percentiles (approximated)
 Type     Name                                                           50%    66%    75%    80%    90%    95%    98%    99%  99.9% 99.99%   100% # reqs
--------|--------------------------------------------------------------|---------|------|------|------|------|------|------|------|------|------|------|
 POST     //api/auth/login                                            2700   5600   6700   7400   8700   9700  10000  11000  11000  11000  11000     93
 POST     //api/tasks                                                 5000   6600   8400   9200   9900  10000  11000  11000  11000  11000  11000     74
--------|--------------------------------------------------------------|---------|------|------|------|------|------|------|------|------|------|------|
 None     Aggregated                                                 4000   6000   7400   8100   9500  10000  11000  11000  11000  11000  11000    167
```

Rows 26 CSV JSON CSV JSON

id	date_created	date_updated	format_input	format_output	path_input	path_output	status	folder	file_name
3527	2022-12-06 03:59:42.957561	2022-12-06 04:02:01.028695	mp3	ogg	2/61d23e83-b0...	2/61d23e83-b0...	processed	61d23e83-b0e8...	test2
3446	2022-12-06 03:59:19.057951	2022-12-06 04:02:54.043134	mp3	ogg	2/b5fd9940-76...	2/b5fd9940-76...	processed	b5fd9940-76d9-...	test2
3402	2022-12-06 03:59:03.636864	2022-12-06 04:05:51.833496	mp3	ogg	2/084fd615-d8f...	2/084fd615-d8f...	processed	084fd615-d8f4-...	test2
3522	2022-12-06 03:59:41.558136	2022-12-06 04:06:29.64384	mp3	ogg	2/ab2d45f4-a1b...	2/e84f6265-10...	processed	ab2d45f4-a1b3-...	test2
3427	2022-12-06 03:59:12.482679	2022-12-06 04:06:29.671929	mp3	ogg	2/ab0f9e5a-13b...	2/ab0f9e5a-13b...	processed	ab0f9e5a-13ba-...	test2
3496	2022-12-06 03:59:33.747965	2022-12-06 04:07:27.790171	mp3	ogg	2/82eadce3-10...	2/82eadce3-10...	processed	82eadce3-1064-...	test2
3380	2022-12-06 03:58:55.126939	2022-12-06 04:09:09.861044	mp3	ogg	2/10ede926-92...	2/10ede926-92...	processed	10ede926-92bb...	test2
3542	2022-12-06 03:59:48.199268	2022-12-06 04:10:50.408572	mp3	ogg	2/06a12996-5f...	2/06a12996-5f...	processed	06a12996-5f6a-...	test2
3544	2022-12-06 03:59:48.720924	2022-12-06 04:11:23.304788	mp3	ogg	2/346d6961-c1...	2/346d6961-c1...	processed	346d6961-c165...	test2
3552	2022-12-06 03:59:51.123127	2022-12-06 04:12:16.389935	mp3	ogg	2/3895f38f-a83...	2/3895f38f-a83...	processed	3895f38f-a832-...	test2
3517	2022-12-06 03:59:40.17743	2022-12-06 04:13:52.866557	mp3	ogg	2/df891ac8-7be...	2/df891ac8-7be...	processed	df891ac8-7be5-...	test2
3423	2022-12-06 03:59:10.981018	2022-12-06 04:15:06.704974	mp3	ogg	2/f26f6441-363...	2/f26f6441-363...	processed	f26f6441-3633-...	test2
3477	2022-12-06 03:59:28.509077	2022-12-06 04:15:50.790405	mp3	ogg	2/e84f6265-10...	2/e84f6265-10...	processed	e84f6265-1044...	test2
3550	2022-12-06 03:59:50.420937	2022-12-06 04:16:44.937102	mp3	ogg	2/baa11f00-001...	2/baa11f00-001...	processed	baa11f00-001e-...	test2
3450	2022-12-06 03:59:20.284135	2022-12-06 04:18:56.014454	mp3	ogg	2/3dfab698-6ed...	2/3dfab698-6ed...	processed	3dfab698-6edb-...	test2

Conclusión:

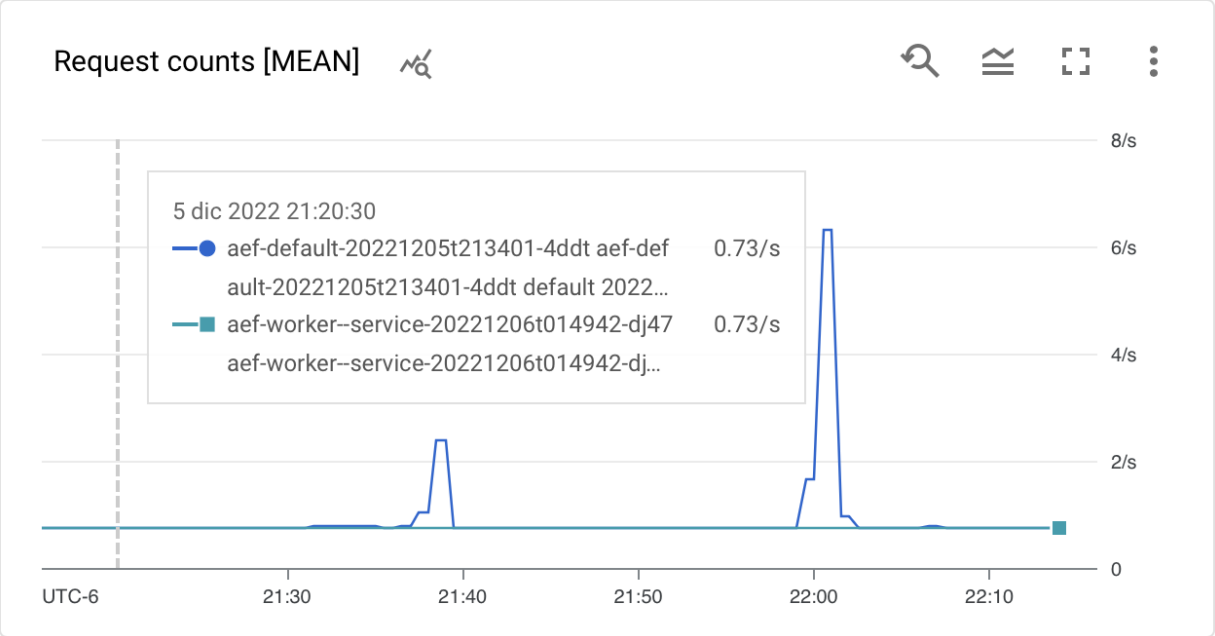
Después de ejecutar una carga de 30 usuario en paralelo y generar 167 request durante 1 minuto, 93 request hacia el enpoint login y 74 request hacia el endpoint task.

Se verifico que para convertir un archivo le tomo aproximadamente 2 minutos, con lo cual se concluye que NO supero la prueba de carga esperada.

Información complementaria

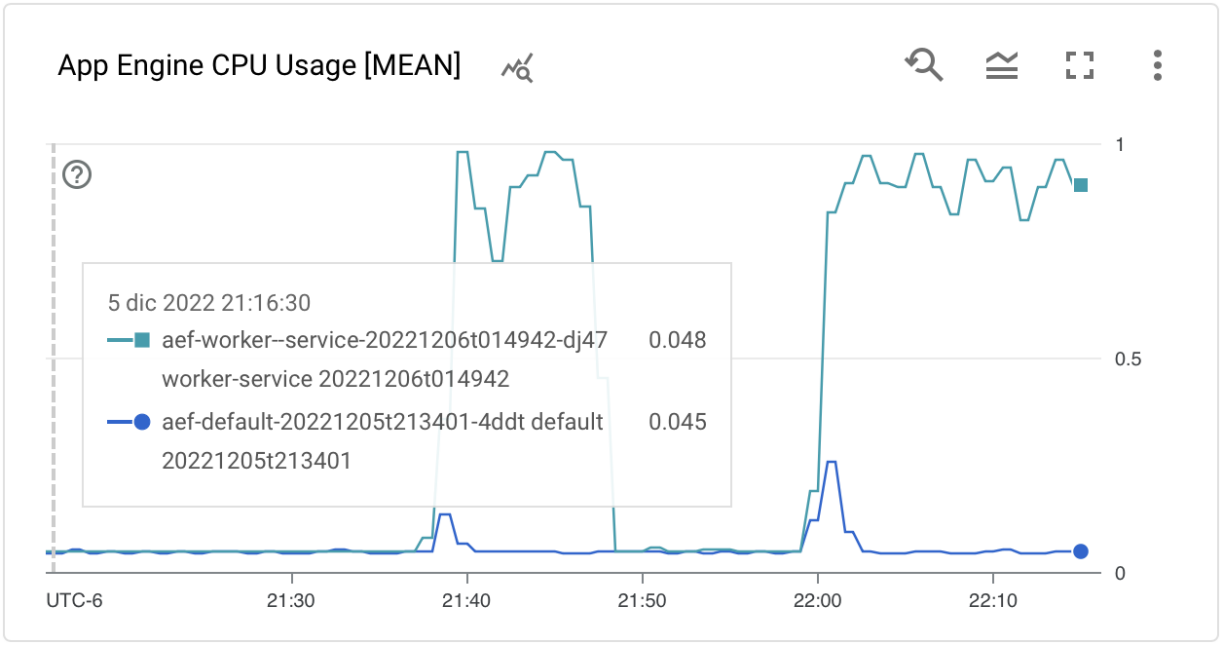
Promedio de número de peticiones

En la siguiente grafica se puede observar el número de peticiones al APi rest, con un pico máximo de 6 peticiones por segundo, luego disminuyendo paulatinamente. Para el worker no se ve tráfico de siendo el comportamiento esperado

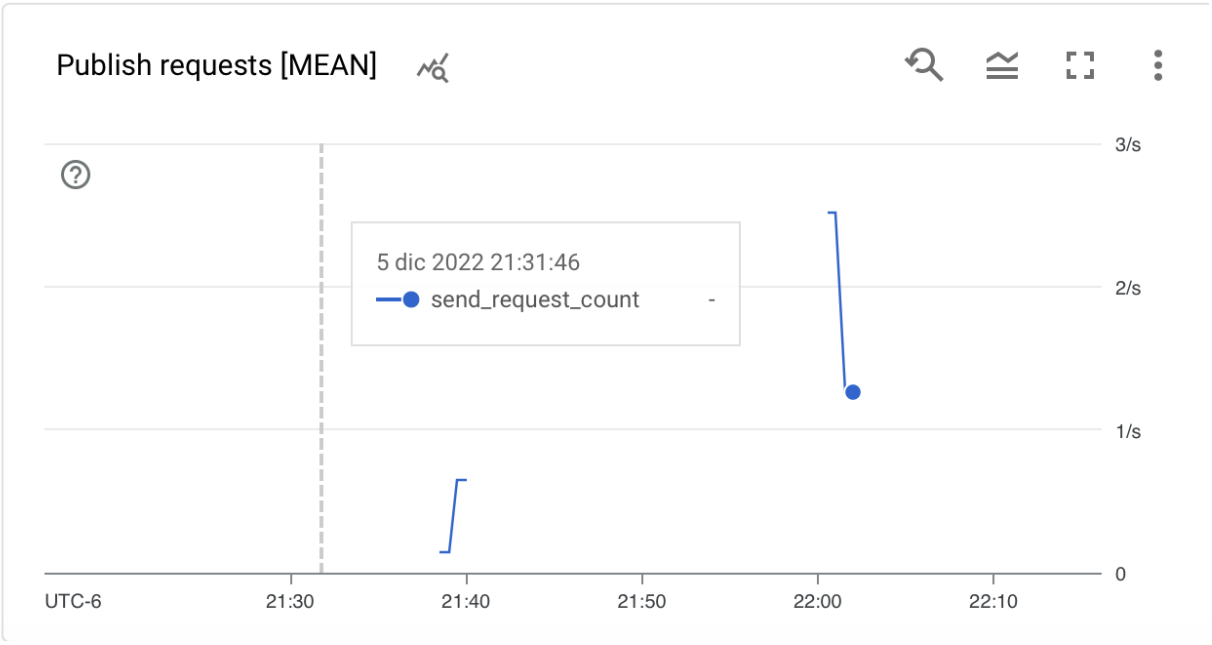


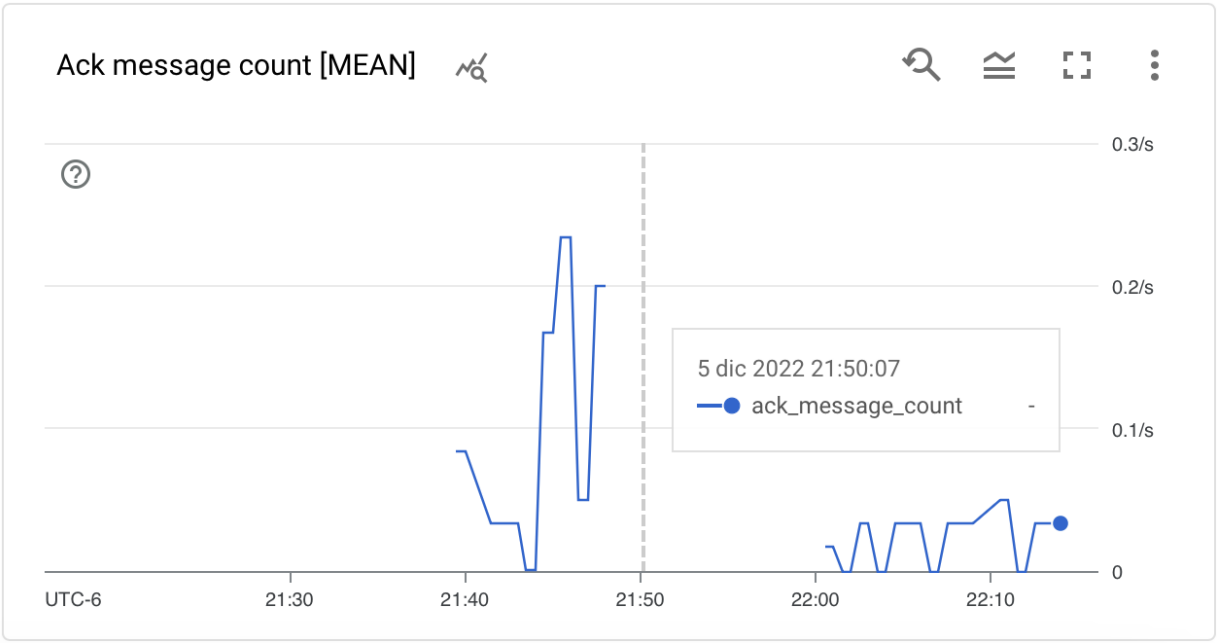
Promedio de uso de CPU instancias App Engine

La segunda grafica evidencia el uso de CPU de los componentes App Engine. La línea azul corresponde al uso del API Rest y coincide con el tráfico de peticiones de la gráfica anterior, la linea verde corresponde al worker este tiene un incremento de carga que coincide con el pico de peticiones del API rest y luego se comporta linealmente al ser un trabajo asíncrono.



Comportamiento publicador




Ack message count [MEAN]
0.3/s
0.2/s
0.1/s
0
5 dic 2022 21:50:07
ack_message_count -
UTC-6
21:30
21:40
21:50
22:00
22:10


Métricas de publicación
4/s
2/s
0
UTC-6
21:40
21:50
22:00
22:10
22:20
Cantidad de mensajes publicados: 0.48/s
Recuento de mensajes de confirmación: 0.03/s
Recuento de mensajes enviados: 0.68/s