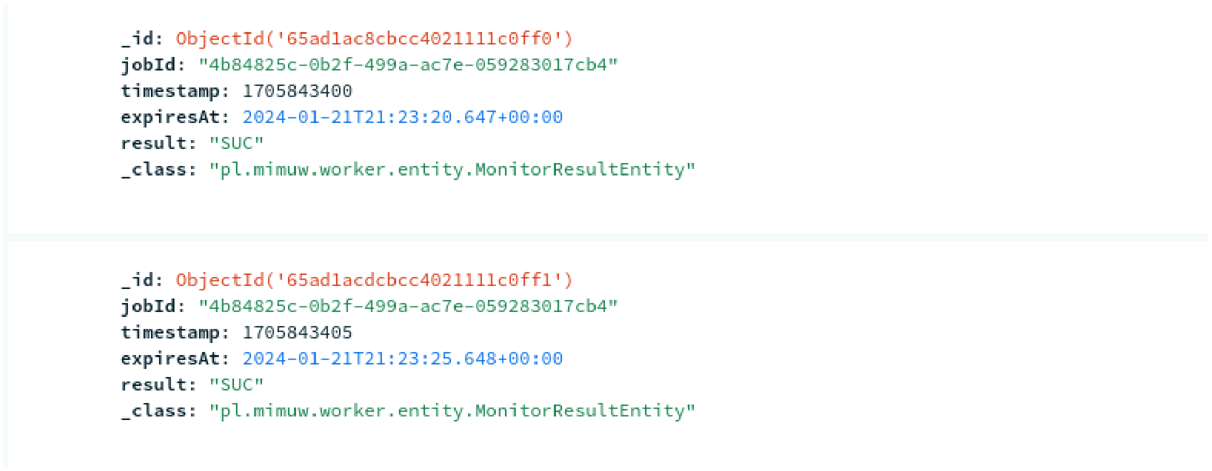
Test results:

E2E functional test:

1. Created a cloud function that can easily change returned status code. (wewu_less/handlers/tester.py)
2. Added job using **WeWu API** with POST:

```JavaScript
{
 "serviceUrl": "https://us-east1-wewu-410223.cloudfunctions.net/wewu_tester",
 "geoRegions": ["us-east1-b"],
 "primaryAdmin": {"email": "******@gmail.com"},
 "secondaryAdmin": {"email": "******@gmail.com"},
 "pollFrequencySecs": 5,
 "alertingWindowNumberOfCalls": 10,
 "alertingWindowCallsFailCount": 6,
 "ackTimeout": 20
}
```

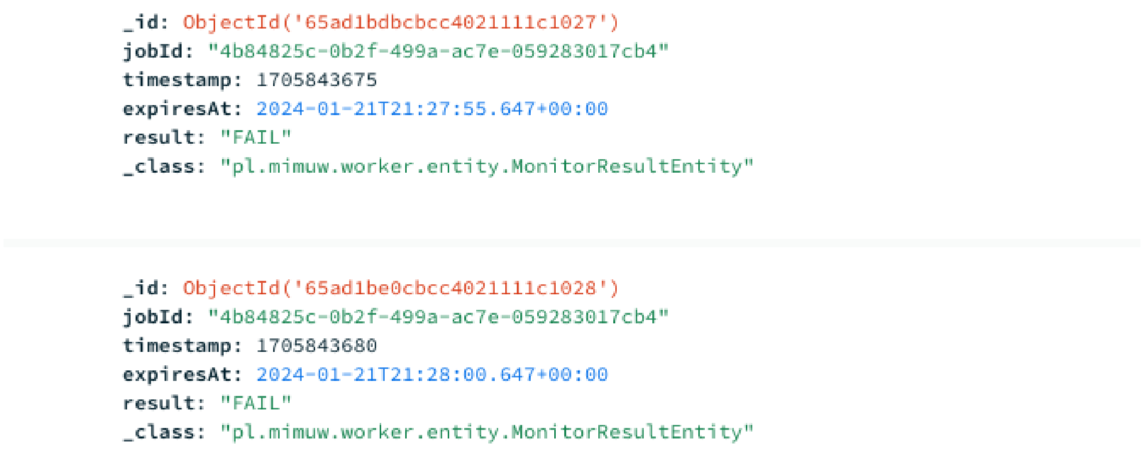
3. Pinging results when tester returns 200:



```
_id: ObjectId('65ad1ac8cbcc4021111c0ff0')
jobId: "4b84825c-0b2f-499a-ac7e-059283017cb4"
timestamp: 1705843400
expiresAt: 2024-01-21T21:23:20.647+00:00
result: "SUC"
_class: "pl.mimuw.worker.entity.MonitorResultEntity"
```

```
_id: ObjectId('65ad1acdcbcc4021111c0ff1')
jobId: "4b84825c-0b2f-499a-ac7e-059283017cb4"
timestamp: 1705843405
expiresAt: 2024-01-21T21:23:25.648+00:00
result: "SUC"
_class: "pl.mimuw.worker.entity.MonitorResultEntity"
```

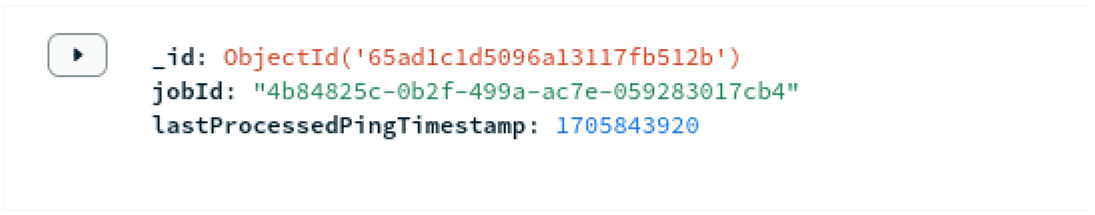
4. Results after POST request to tester service (now it returns 500):



```
_id: ObjectId('65ad1bdbcbcc4021111c1027')
jobId: "4b84825c-0b2f-499a-ac7e-059283017cb4"
timestamp: 1705843675
expiresAt: 2024-01-21T21:27:55.647+00:00
result: "FAIL"
_class: "pl.mimuw.worker.entity.MonitorResultEntity"
```

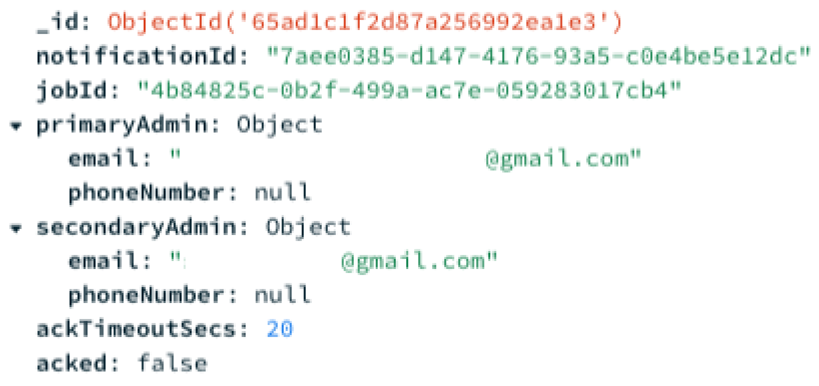
```
_id: ObjectId('65ad1be0cbcc4021111c1028')
jobId: "4b84825c-0b2f-499a-ac7e-059283017cb4"
timestamp: 1705843680
expiresAt: 2024-01-21T21:28:00.647+00:00
result: "FAIL"
_class: "pl.mimuw.worker.entity.MonitorResultEntity"
```

5. We can check in the database that the scheduled buzzator added a record about last detected failure of the job, to avoid spamming alerts if the monitored service is still down. Also, it sent an event to inform the primary admin about the failure.

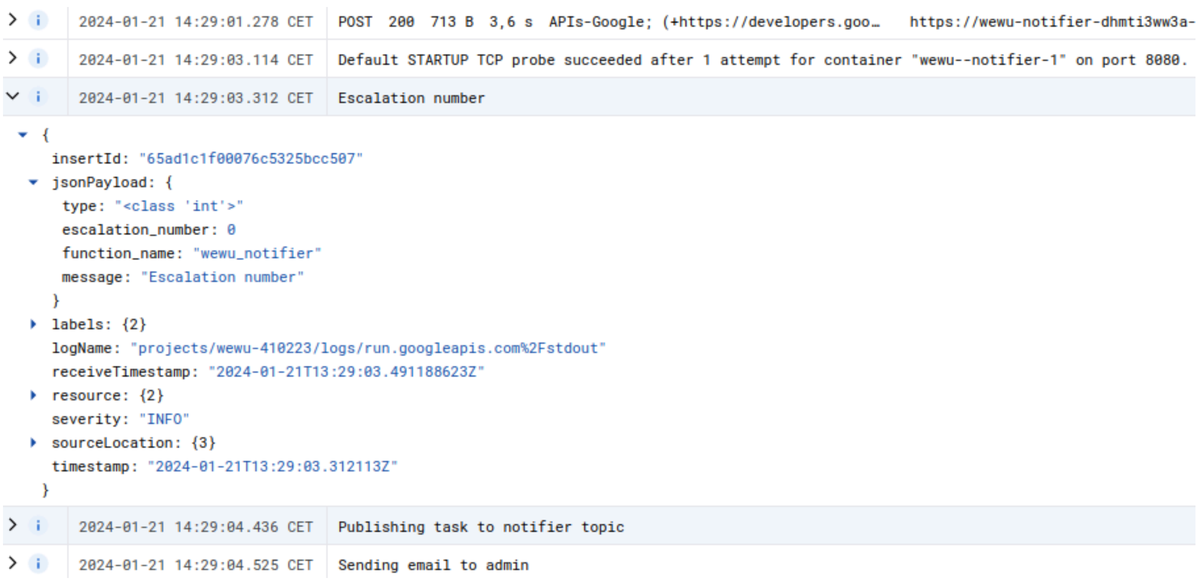


```
_id: ObjectId('65ad1c1d5096a13117fb512b')
jobId: "4b84825c-0b2f-499a-ac7e-059283017cb4"
lastProcessedPingTimestamp: 1705843920
```

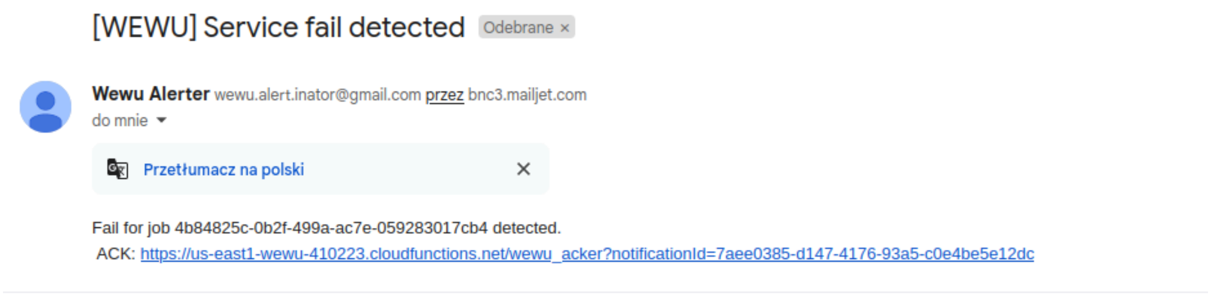
6. The notifier received the event about sending an alert to the admin and saved the information about the notification in the database. After that it sent an email to ******@gmail.com:



```
  _id: ObjectId('65ad1c1f2d87a256992ea1e3')
  notificationId: "7aee0385-d147-4176-93a5-c0e4be5e12dc"
  jobId: "4b84825c-0b2f-499a-ac7e-059283017cb4"
▾ primaryAdmin: Object
    email: "[redacted]@gmail.com"
    phoneNumber: null
▾ secondaryAdmin: Object
    email: "[redacted]@gmail.com"
    phoneNumber: null
  ackTimeoutSecs: 20
  acked: false
```



```
> i  2024-01-21 14:29:01.278 CET  POST  200  713 B  3,6 s  APIs-Google; (+https://developers.goo…  https://wewu-notifier-dhmti3ww3a-
> i  2024-01-21 14:29:03.114 CET  Default STARTUP TCP probe succeeded after 1 attempt for container "wewu--notifier-1" on port 8080.
v i  2024-01-21 14:29:03.312 CET  Escalation number
 ▾ {
     insertId: "65ad1c1f00076c5325bcc507"
   ▾ jsonPayload: {
       type: "<class 'int'>"
       escalation_number: 0
       function_name: "wewu_notifier"
       message: "Escalation number"
     }
   ▸ labels: {2}
     logName: "projects/wewu-410223/logs/run.googleapis.com%2Fstdout"
     receiveTimestamp: "2024-01-21T13:29:03.491188623Z"
   ▸ resource: {2}
     severity: "INFO"
   ▸ sourceLocation: {3}
     timestamp: "2024-01-21T13:29:03.312113Z"
   }
> i  2024-01-21 14:29:04.436 CET  Publishing task to notifier topic
> i  2024-01-21 14:29:04.525 CET  Sending email to admin
```



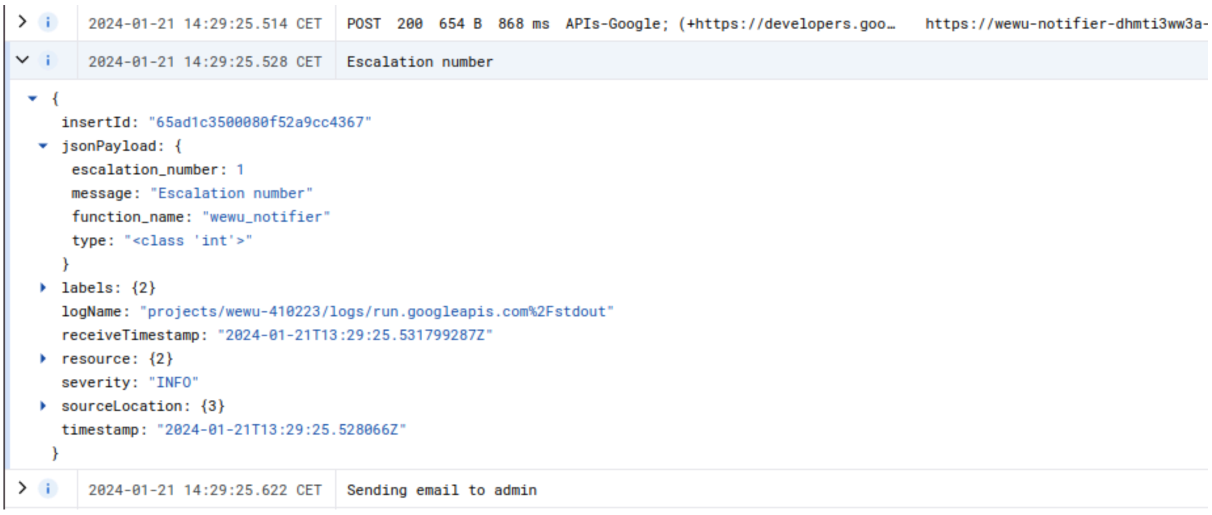
[WEWU] Service fail detected Odebrane ×

Wewu Alerter wewu.alert.inator@gmail.com przez bnc3.mailjet.com
do mnie

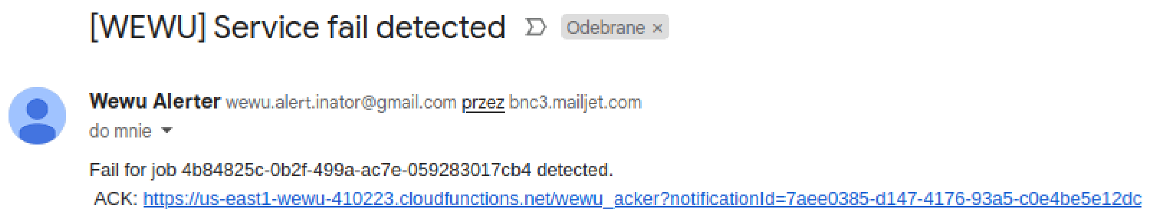
Przetłumacz na polski ×

Fail for job 4b84825c-0b2f-499a-ac7e-059283017cb4 detected.
ACK: https://us-east1-wewu-410223.cloudfunctions.net/wewu_acker?notificationId=7aee0385-d147-4176-93a5-c0e4be5e12dc

7. After roughly 20 seconds (time for ACK specified in request) another notification was sent (attached mail from ******@gmail.com):



```
>  i  2024-01-21 14:29:25.514 CET   POST  200  654 B  868 ms  APIs-Google; (+https://developers.goo…   https://wewu-notifier-dhmti3ww3a-
v  i  2024-01-21 14:29:25.528 CET   Escalation number
  {
   insertId: "65ad1c3500080f52a9cc4367"
   jsonPayload: {
    escalation_number: 1
    message: "Escalation number"
    function_name: "wewu_notifier"
    type: "<class 'int'>"
   }
   labels: {2}
   logName: "projects/wewu-410223/logs/run.googleapis.com%2Fstdout"
   receiveTimestamp: "2024-01-21T13:29:25.531799287Z"
   resource: {2}
   severity: "INFO"
   sourceLocation: {3}
   timestamp: "2024-01-21T13:29:25.528066Z"
  }
>  i  2024-01-21 14:29:25.622 CET   Sending email to admin
```



[WEWU] Service fail detected Odebrane ×

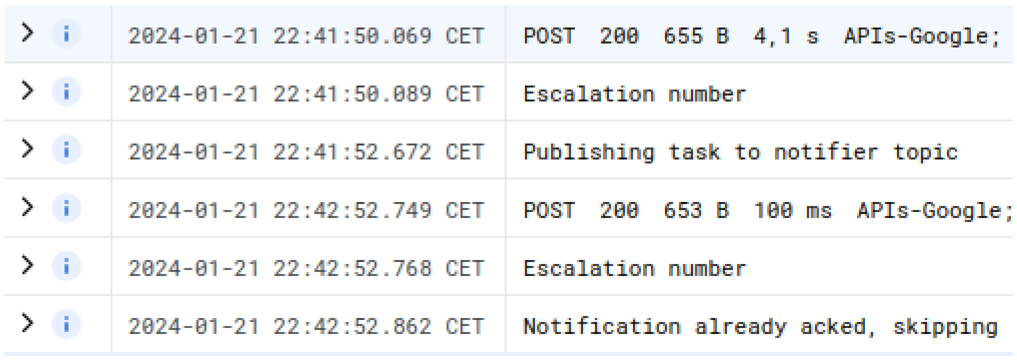
Wewu Alerter wewu.alert.inator@gmail.com przez bnc3.mailjet.com
do mnie

Fail for job 4b84825c-0b2f-499a-ac7e-059283017cb4 detected.
ACK: https://us-east1-wewu-410223.cloudfunctions.net/wewu_acker?notificationId=7aee0385-d147-4176-93a5-c0e4be5e12dc

8. After another POST request to test service (now it returns 200) no more notifications arrive and service does not detect more failures (as the service is working).

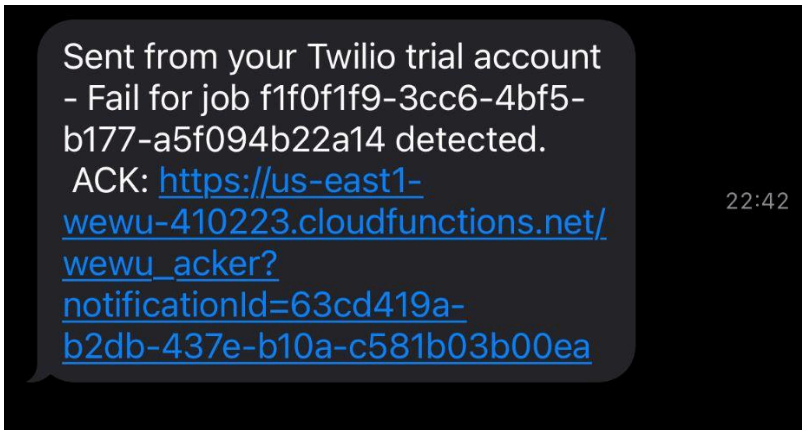
Additional Notes:

- Notification can be ACKed by clicking the link in the received alert. If the first admin will ACK the message before *ackTimeoutSecs* has passed since sending the alert, the second admin won't be notified. We can observe this behavior in the logs (*ackTimeoutSecs* for this task was set to 60):



```
>  i  2024-01-21 22:41:50.069 CET   POST  200  655 B  4,1 s  APIs-Google;
>  i  2024-01-21 22:41:50.089 CET   Escalation number
>  i  2024-01-21 22:41:52.672 CET   Publishing task to notifier topic
>  i  2024-01-21 22:42:52.749 CET   POST  200  653 B  100 ms  APIs-Google;
>  i  2024-01-21 22:42:52.768 CET   Escalation number
>  i  2024-01-21 22:42:52.862 CET   Notification already acked, skipping
```

- Notification can be also send through SMS:



Sent from your Twilio trial account - Fail for job f1f0f1f9-3cc6-4bf5-b177-a5f094b22a14 detected.
ACK: https://us-east1-wewu-410223.cloudfunctions.net/wewu_acker?notificationId=63cd419a-b2db-437e-b10a-c581b03b00ea

22:42

Load test:

Testing settings:

- Worker service can process up to 100 tasks.
- Maximum number of pods equal to 20.
- Tasks with pretty low poll frequencies (from 8s to 15s).
- WeWu-scheduler and WeWu-buzzator schedule to run every 1 minute.

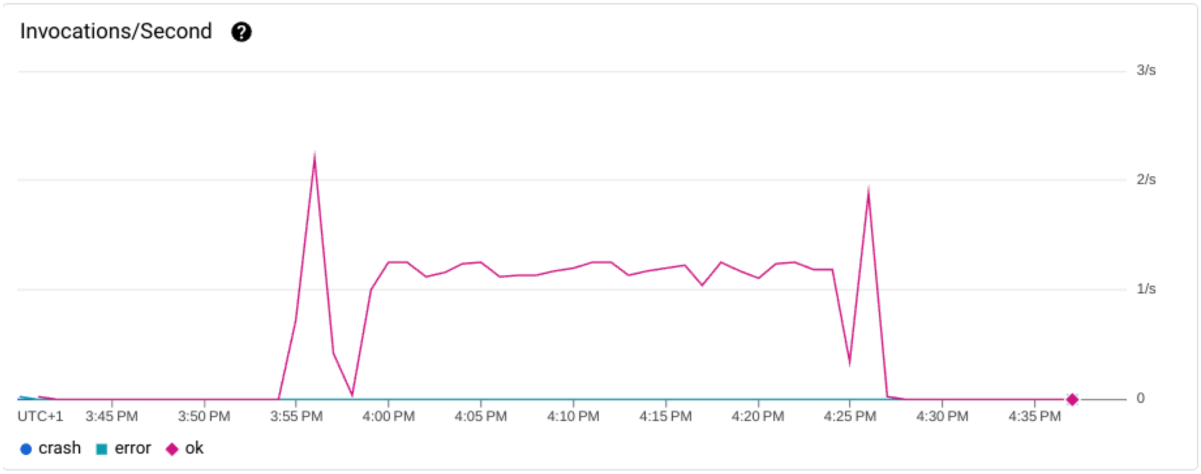
Test process:

- Create an initial load of tasks equal to 200 (send requests to the API) and wait 2 min.
- Add 15 new tasks every 6 seconds (send requests to the API).

We used a python script - link.
Note: We requested to monitor only services with high-availability (like google.com or facebook.com). At the end we ran the script for a while again to add some more requests, which explains the spike we can observe in the metrics. Below you can find results gathered from the metrics we find the most informative about the state of our service:

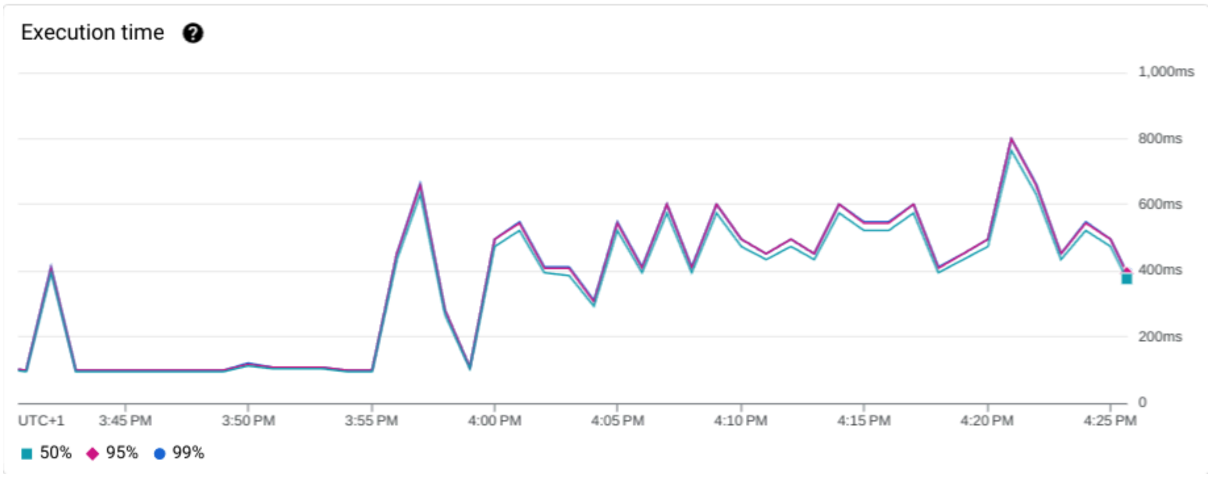
WeWu-API:

This metric reflects characteristics of our test. Spike at the beginning and at the end. Some idle time between spikes and requests coming at a consistent rate.



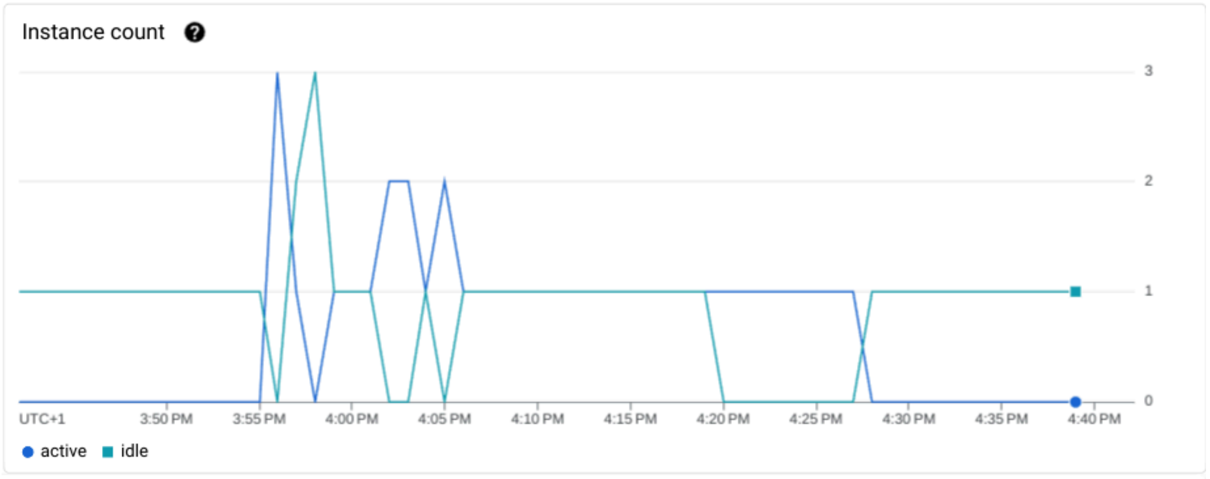
WeWu-scheduler:

Scheduler execution time is related to the amount of requests to the api as well as all currently monitored services. Here we can see it's execution time is longer at the beginning, later it has nothing to schedule for a while (as all tasks are processed by workers). After that its execution time lengthens after requests start coming at a consistent rate. Important difference between scheduler and api is that after the test, when no more requests are coming, it still has old jobs to schedule again.



WeWu-copy-and-paste-inator:

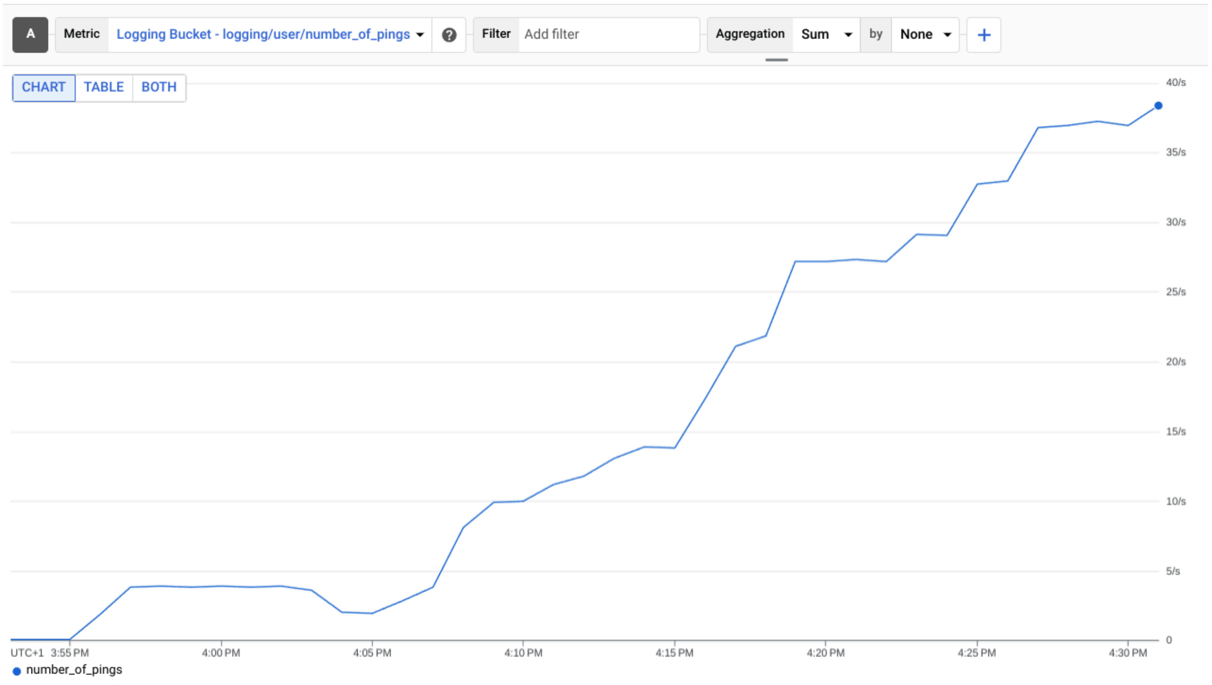
This is probably not the most interesting chart, but it clearly shows the scaling capabilities of Cloud Functions – when we created the first 200 messages the number of instances raised in order to match the current demand to process all the incoming events.

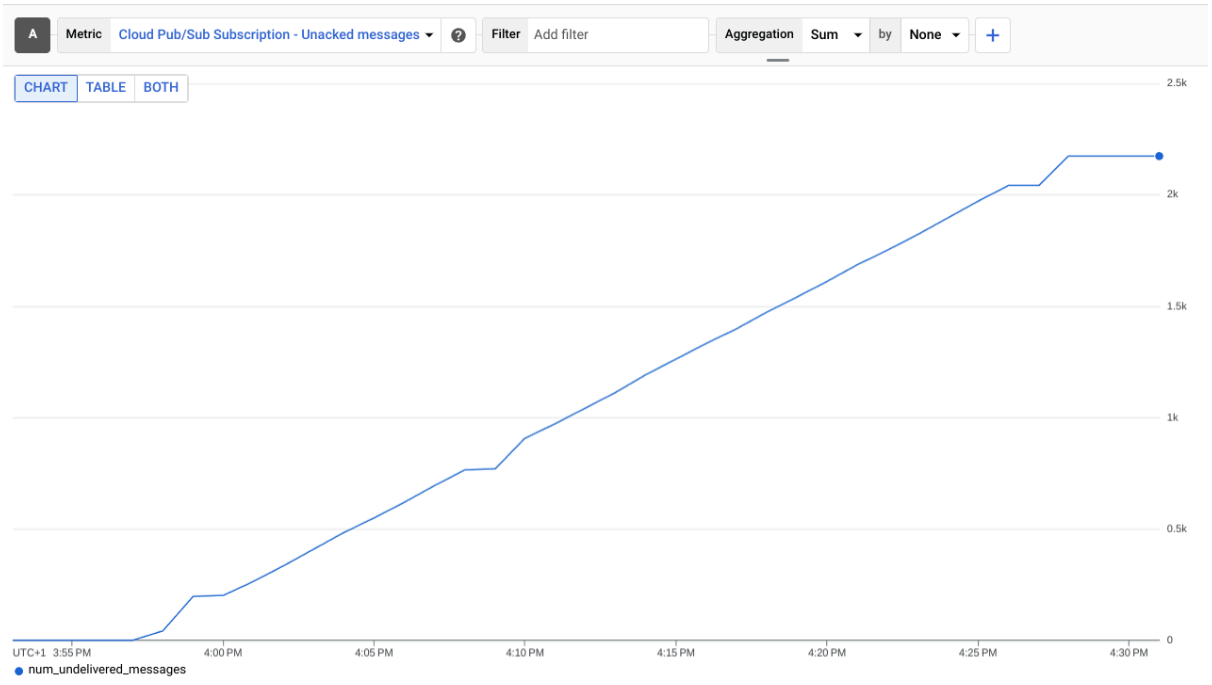


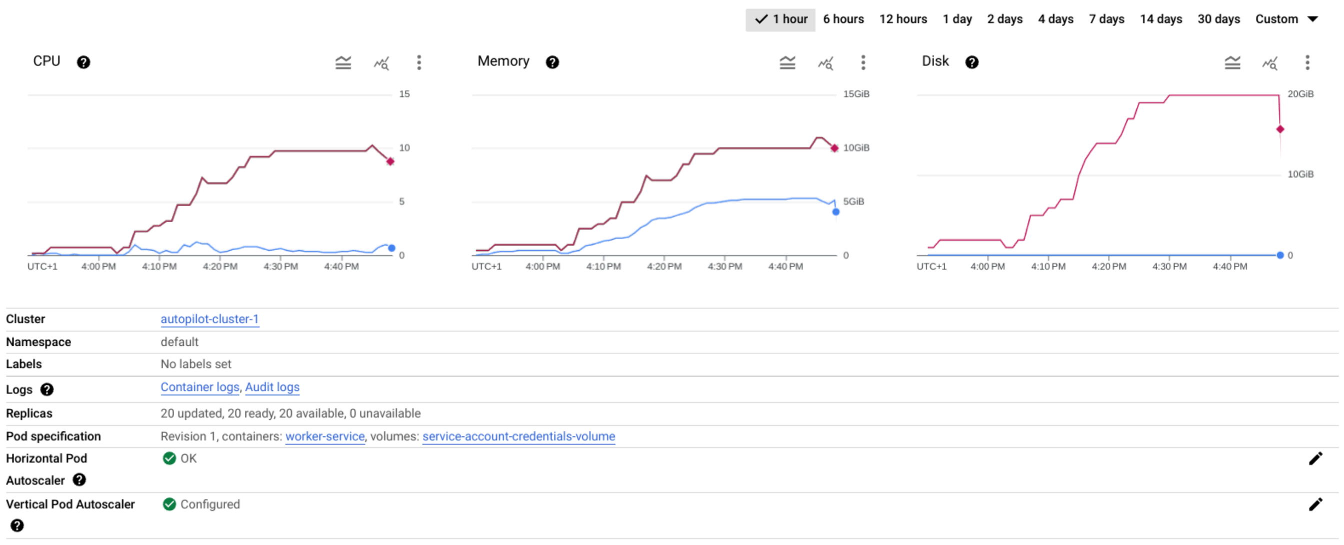
WeWu-worker:

In the worker, we wanted to observe two main things: the workers take on new tasks, and with the increase of the load – the number of pods increase accordingly to the specified autoscaler. On the first two charts we can clearly see the correlation between the number of monitoring tasks in the Pub/Sub and the number of pings sent per second, which indicates that workers correctly pull new tasks if possible. The creation of the new pods we could observe in the real-time, but it is also reflected on the cluster CPU/memory limits since it was scaling up in the meantime to meet the resources requirements requested by the pods.

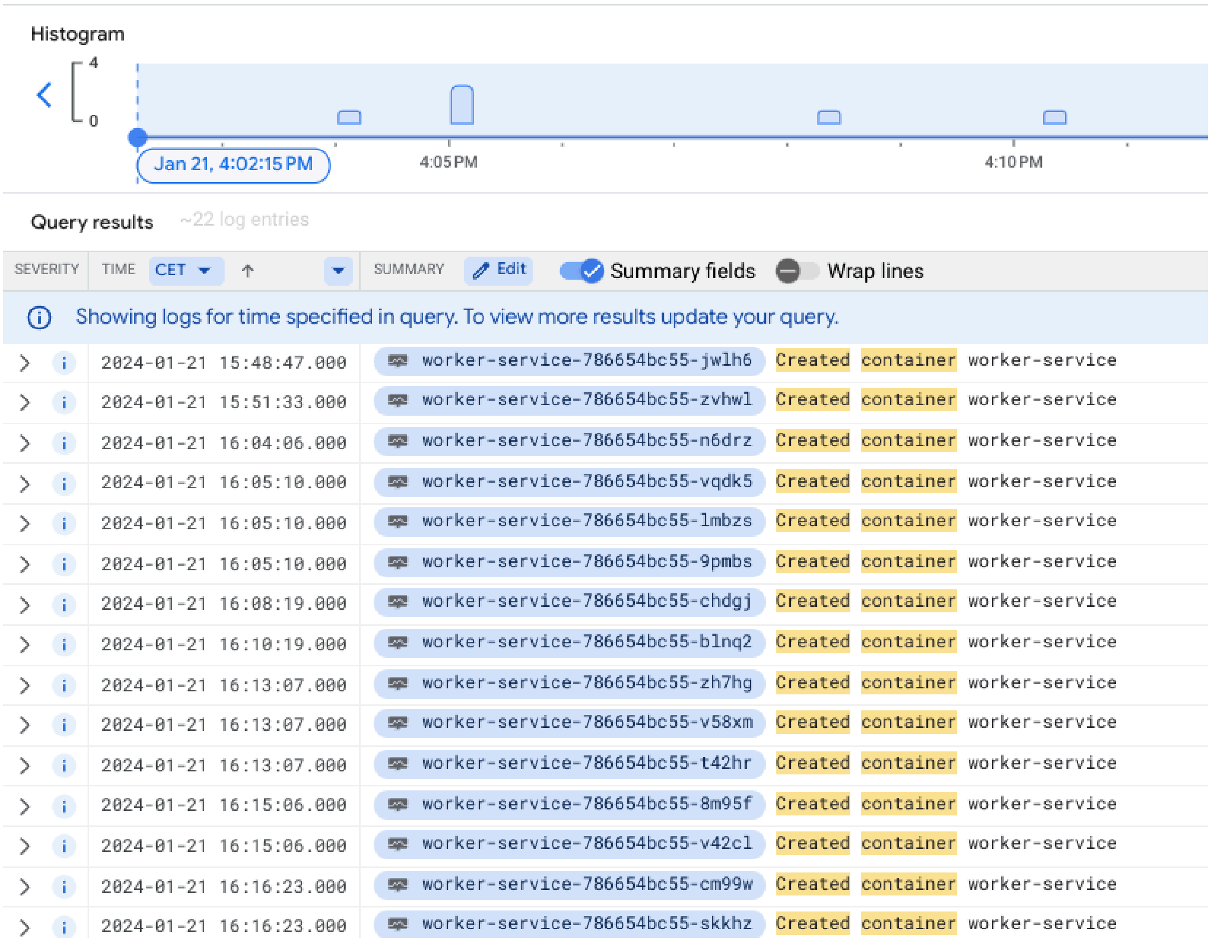
Additionally, we can look up the GKE logs in order to check if the new pods were requested in the sensible time intervals.







Cluster	autopilot-cluster-1
Namespace	default
Labels	No labels set
Logs	Container logs, Audit logs
Replicas	20 updated, 20 ready, 20 available, 0 unavailable
Pod specification	Revision 1, containers: worker-service, volumes: service-account-credentials-volume
Horizontal Pod Autoscaler	✓ OK
Vertical Pod Autoscaler	✓ Configured

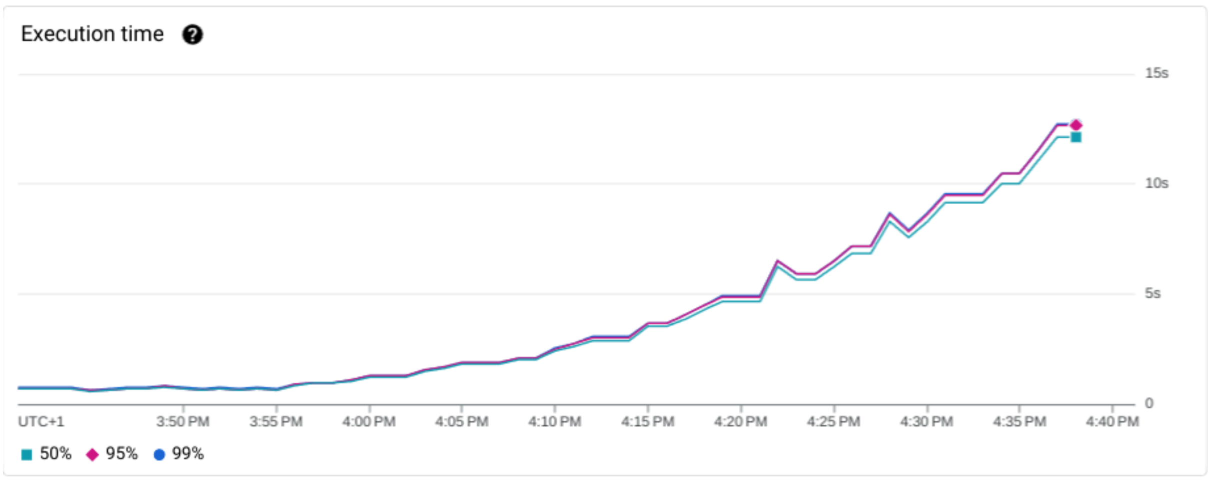


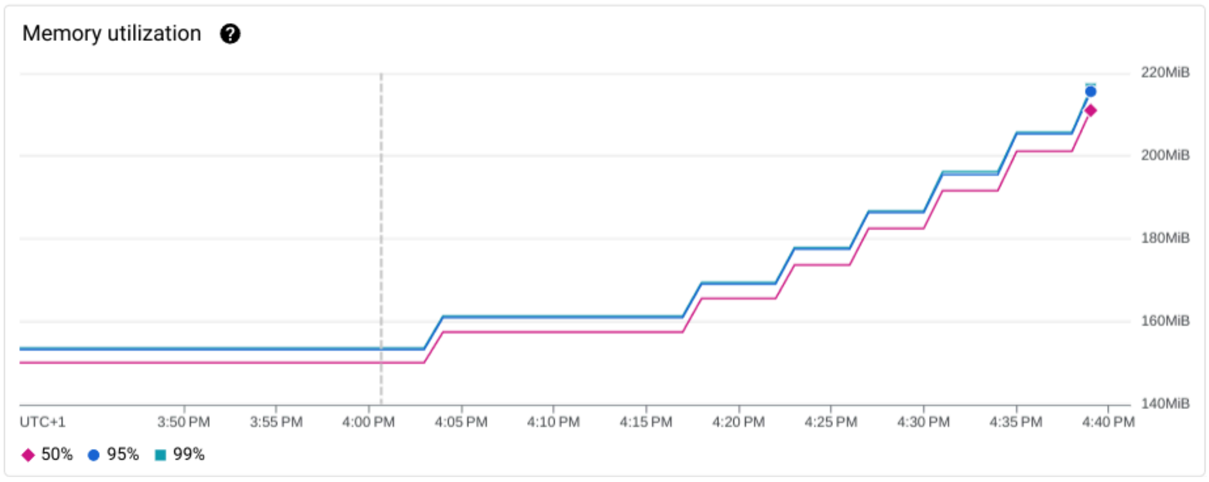
Showing logs for time specified in query. To view more results update your query.

```
2024-01-21 15:48:47.000  worker-service-786654bc55-jwlh6  Created container worker-service
2024-01-21 15:51:33.000  worker-service-786654bc55-zvhwl  Created container worker-service
2024-01-21 16:04:06.000  worker-service-786654bc55-n6drz  Created container worker-service
2024-01-21 16:05:10.000  worker-service-786654bc55-vqdk5  Created container worker-service
2024-01-21 16:05:10.000  worker-service-786654bc55-lmbzs  Created container worker-service
2024-01-21 16:05:10.000  worker-service-786654bc55-9pmbs  Created container worker-service
2024-01-21 16:08:19.000  worker-service-786654bc55-chdgj  Created container worker-service
2024-01-21 16:10:19.000  worker-service-786654bc55-blnq2  Created container worker-service
2024-01-21 16:13:07.000  worker-service-786654bc55-zh7hg  Created container worker-service
2024-01-21 16:13:07.000  worker-service-786654bc55-v58xm  Created container worker-service
2024-01-21 16:13:07.000  worker-service-786654bc55-t42hr  Created container worker-service
2024-01-21 16:15:06.000  worker-service-786654bc55-8m95f  Created container worker-service
2024-01-21 16:15:06.000  worker-service-786654bc55-v42cl  Created container worker-service
2024-01-21 16:16:23.000  worker-service-786654bc55-cm99w  Created container worker-service
2024-01-21 16:16:23.000  worker-service-786654bc55-skkhz  Created container worker-service
```

WeWu-buzzator:

We see that buzzator's execution time is directly related to the number of pings. After the start of the test its execution time rose consistently, even after the test ended, as the services were still monitored. We can see that even with this short period of time and only about 2k requests its execution time rose considerably and sharding would be necessary.





Stress test:

Testing settings:

- Worker service can process up to 100 tasks.
- Maximum number of pods equal to 20.
- Tasks with pretty high poll frequencies (from 1s to 2s, it changed in the meantime).
- WeWu-scheduler and WeWu-buzzator schedule to run every 1 minute.

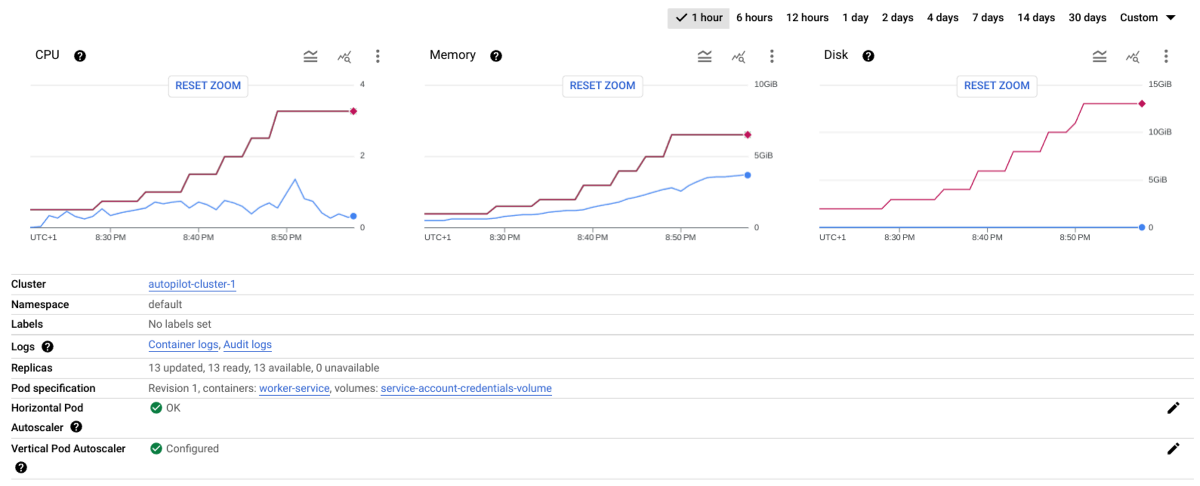
Test process:

- Create an initial load of tasks equal to 100 (send requests to the API) and wait 2 min.
- Create a POST request to test service so it will return 500 on GET requests.
- Add 5 new tasks every 1 second, ending up with a load of 300 tasks.
- Create another load of 200. Make 5 iterations of this step running script manually in irregular intervals (up to max. a few minutes).

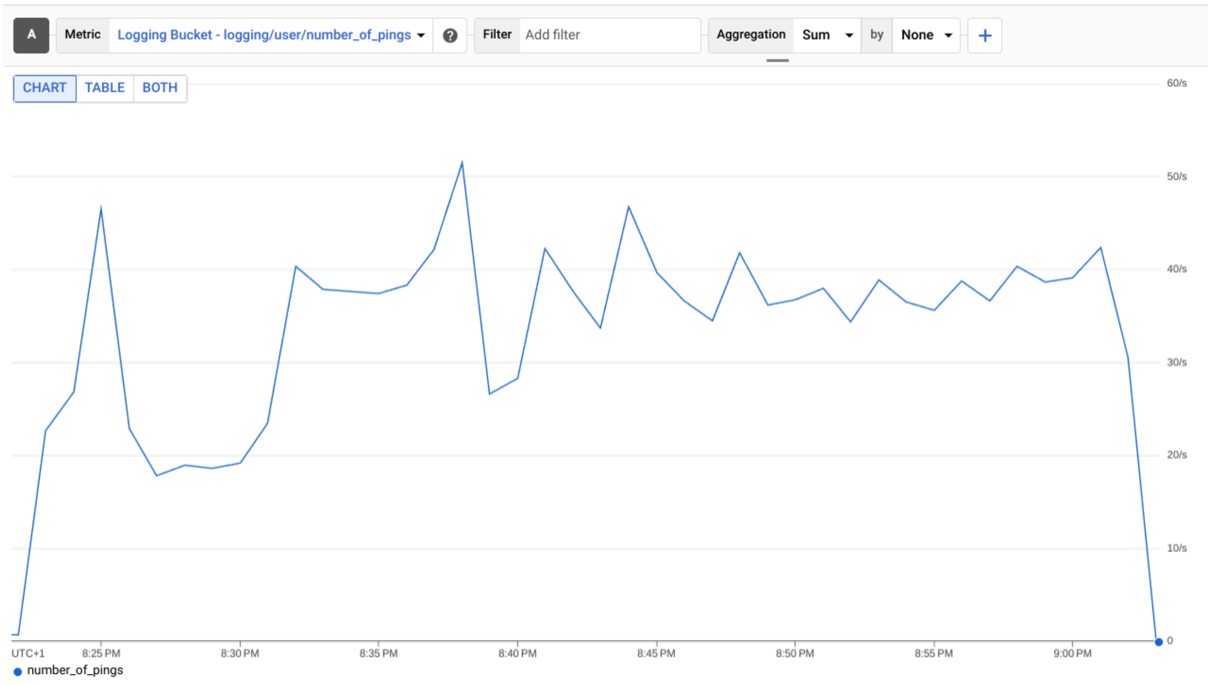
We used a python script - link.

WeWu-worker:

This time we can see that resources used by the workers were much closer to the cluster limit than in the load test. CPU usage was higher as the services had higher frequency so workers had to make more requests and spent less time idle.

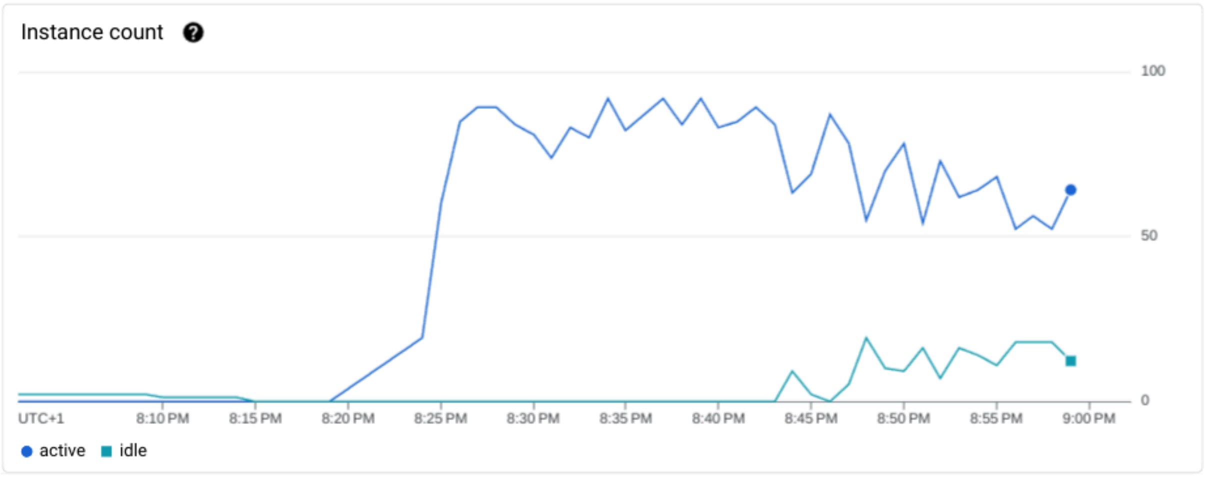


Cluster	autopilot-cluster-1
Namespace	default
Labels	No labels set
Logs	Container logs, Audit logs
Replicas	13 updated, 13 ready, 13 available, 0 unavailable
Pod specification	Revision 1, containers: worker-service, volumes: service-account-credentials-volume
Horizontal Pod Autoscaler	OK
Vertical Pod Autoscaler	Configured



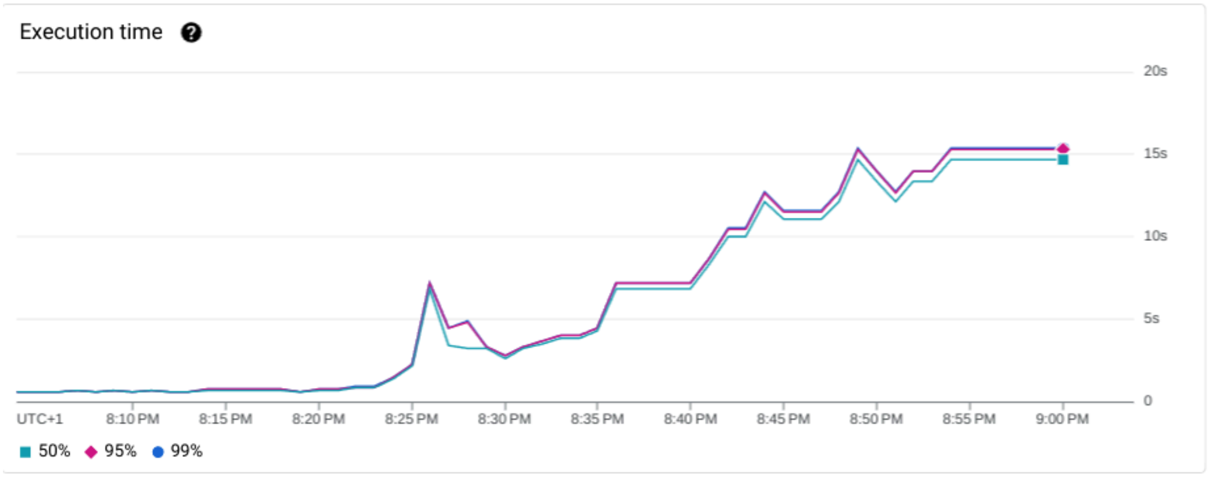
WeWu-notifier:

Increase in the number of instances of wewu_notifier was expected as there were a lot of failures and it needed to send alerts to users. Number of instances in this diagram is a little bit exaggerated as the free mail api we are using has a limit of sending notification in a time period, so the cloud function had to rerun because of unsuccessful invocation (and also handle the new alerts at the same time).



WeWu-buzzator:

Buzzator struggled a bit with a big amount of data from pinging the services so its execution time constantly rose, just like we expected.


Execution time
20s
15s
10s
5s
UTC+1
8:10 PM
8:15 PM
8:20 PM
8:25 PM
8:30 PM
8:35 PM
8:40 PM
8:45 PM
8:50 PM
8:55 PM
9:00 PM
■ 50%
◆ 95%
● 99%