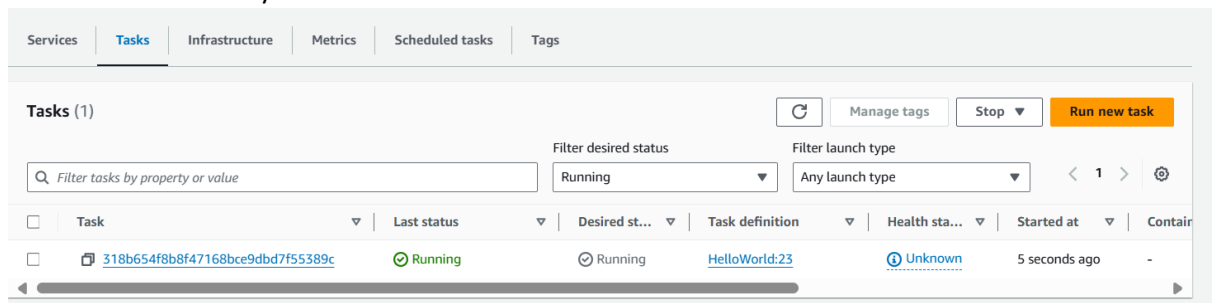


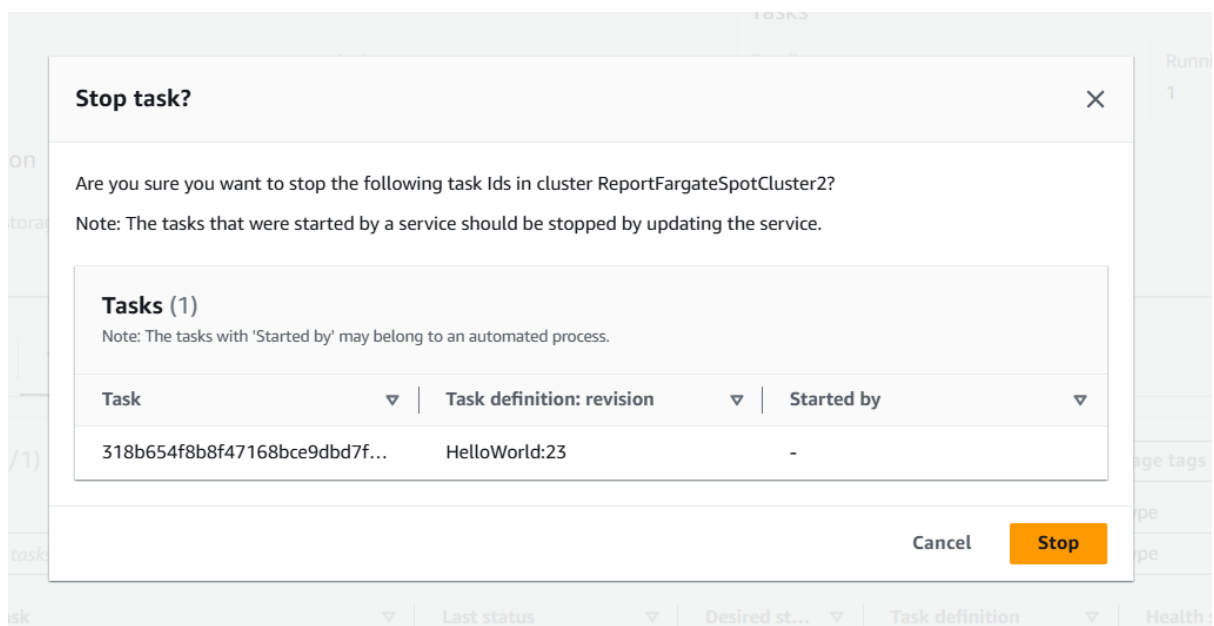
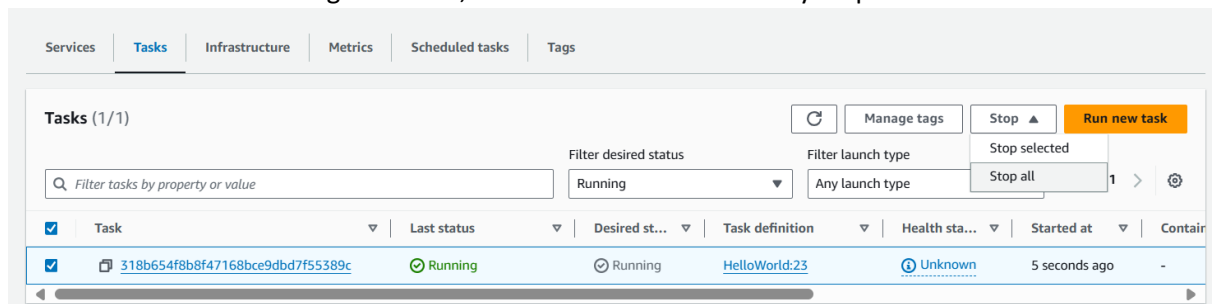
1. Any task that is running on Amazon ECS Spot capacity provider is subject to interruption. There are two possibilities for task non-completion.
 - a. The running Amazon ECS Fargate task is interrupted. Whenever the task is interrupted there will 120 seconds post which task will be killed.
 - b. There is possibility that Amazon ECS Fargate task is not available for invocation.

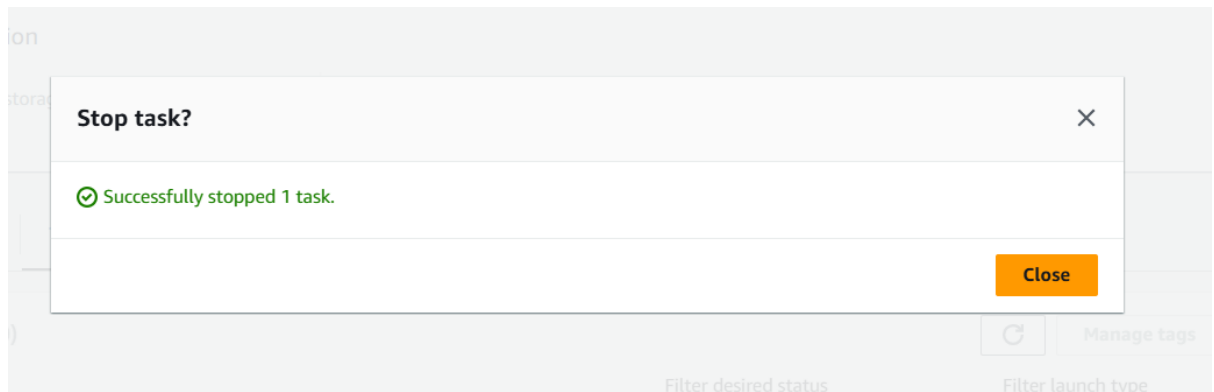
In the write up we shall detail how to simulate both the situation in testing environment and how to remediate.

2. How to simulate Amazon ECS Fargate task interruption while the task is running. When any Amazon ECS Fargate task is interrupted, the SIGTERM state change event is sent from task kernel to process. The SIGTERM event has to be captured in the code and do needful. Here we are simulating the process by killing the task manually.
 - a. Run the task manually from Amazon ECS cluster

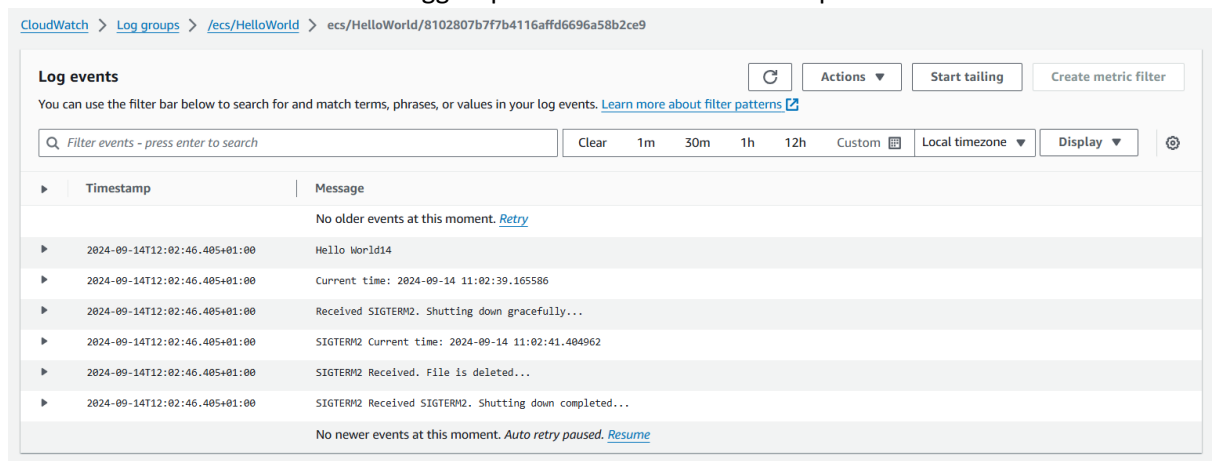


- b. While the task is on running condition, select the task and manually stop the task





- c. Validate in Amazon CloudWatch Loggroup that task has been interrupted.



3. Code for Gracious Exit

We have introduced the function which will receive the SIGTERM when Amazon ECS Fargate task is interrupted.

```

10
11 def sigterm_handler(signum, frame):
12     print("Received SIGTERM2. Shutting down gracefully...")
13     current_time = datetime.datetime.fromtimestamp(time.time())
14     print(f"SIGTERM2 Current time: {current_time}")
15     time.sleep(5)
16     current_time = datetime.datetime.fromtimestamp(time.time())
17     s3 = boto3.client('s3')
18     s3.delete_object(Bucket=bucket_name, Key=file_name)
19     print("SIGTERM2 Received. File is deleted...")
20     print("SIGTERM2 Received SIGTERM2. Shutting down completed...")
21     # Perform cleanup operations here
22     sys.exit(0)
23

```

4. The Amazon ECS Fargate task definition for timeout

When a Amazon ECS Fargate Task is interrupted, the process gets 120 seconds to gracefully exit the process. In the function, all necessary work can be undone so that freshly task can be started.

To simulate this we have introduced some delays in the process.

- a. Manually introduce delay in code

We have introduced a delay in the code so that process runs for longer time when task can be interrupted by manually stopping the task.

```
110
111     print("Hello World14")
112     current_time = datetime.datetime.fromtimestamp(time.time())
113     print(f"Current time: {current_time}")
114     time.sleep(30)
115     current_time = datetime.datetime.fromtimestamp(time.time())
116     print(f"Current time+: {current_time}")
117     print(sum)
```

- b. Increase Amazon ECS Fargate task timeout

Any Amazon ECS Fargate Task by default gets 30 second to complete before stopping. This is a configurable and we have configured it in task definition. This gives the time to SIGTERM handler to complete the graceful exit. When the timeout is configured with less second the SIGTERM handler couldn't finish the execution.

```
21         "startTimeout": 5,
22         "stopTimeout": 30,
23         "logConfiguration": {
```

- c. Add delay in in SIGTERM code

Finally we have added the delay in SIGTERM handler to validate the SIGTERM handler gets enough time to complete the graceful exit.

```
11 def sigterm_handler(signum, frame):
12     print("Received SIGTERM2. Shutting down gracefully...")
13     current_time = datetime.datetime.fromtimestamp(time.time())
14     print(f"SIGTERM2 Current time: {current_time}")
15     time.sleep(5)
16     current_time = datetime.datetime.fromtimestamp(time.time())
```

CloudWatch > Log groups > /ecs/HelloWorld > ecs/HelloWorld/8102807b7f7b4116affd6696a58b2ce9

| Log events | | Actions | Start tailing | Create metric filter |
|---|--|---------|---------------|----------------------|
| You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns | | | | |
| Filter events - press enter to search | | Clear | 1m | 30m |
| | | | 1h | 12h |
| | | | Custom | Local timezone |
| | | | Display | |
| Timestamp | Message | | | |
| | No older events at this moment. Retry | | | |
| 2024-09-14T12:02:46.405+01:00 | Hello World14 | | | |
| 2024-09-14T12:02:46.405+01:00 | Current time: 2024-09-14 11:02:39.165586 | | | |
| 2024-09-14T12:02:46.405+01:00 | Received SIGTERM2. Shutting down gracefully... | | | |
| 2024-09-14T12:02:46.405+01:00 | SIGTERM2 Current time: 2024-09-14 11:02:41.404962 | | | |
| 2024-09-14T12:02:46.405+01:00 | SIGTERM2 Received. File is deleted... | | | |
| 2024-09-14T12:02:46.405+01:00 | SIGTERM2 Received SIGTERM2. Shutting down completed... | | | |
| No newer events at this moment. Auto retry paused. Resume | | | | |

5. How to simulate when the Amazon ECS Fargate task(spot) not able to invoke for capacity non-availability.

In our solution, the docker container in Amazon ECS Fargate Spot is invoked through Amazon EventBridge scheduler. In earlier section, we have seen how explained how task interruption can be alerted and remediated.

In the following section, we shall explain how to get notification when Amazon EventBridge Scheduler not able to invoke the Amazon ECS Fargate Task on spot for non availability.

- a. Amazon Schedule redrive if task not available. If the task fails after retry it will send to Amazon SQS DLQ which will send mail through Amazon SNS notification.

Retry policy and dead-letter queue (DLQ)

Retry policy

Info

By default, EventBridge Scheduler attempts to retry failed invocations for up to 24 hours. You can specify the maximum age of the event and the maximum number of times to retry.

☒ Retry

Maximum age of event - optional
The maximum amount of time to keep unprocessed events. The maximum and default value is 24 hours.

hour(s) minute(s)

Retry attempts - optional
The maximum number of times to retry when a target returns an error. The maximum value is 185 times.

times

Dead-letter queue (DLQ)
Standard Amazon SQS queues that EventBridge Scheduler uses to store events that couldn't be delivered successfully to a target.

☐ None

☒ Select an Amazon SQS queue in my AWS account as a DLQ

☐ Specify an Amazon SQS queue in other AWS accounts as a DLQ

SQS queue
Select an SQS queue

▼

We shall define redrive, SNS notification in case the task is not able to place.

Further we shall simulate the task non availability.

6. Simulation of Amazon ECS task on spot non available.

We shall create a schedule 5 minutes in advance with Amazon ECS Fargate task definition version. Once the schedule is active and ready to be launched, we shall deregister the task definition version. As a result the task will fail to get invoked and message will be passed to Amazon SQS DLQ which can trigger notification.

The screenshot shows the 'Schedule pattern' configuration page in the AWS console. It has two tabs: 'Occurrence' and 'Info'. Under 'Occurrence', there are two radio buttons: 'One-time schedule' (which is selected) and 'Recurring schedule'. Below this, the 'Date and time' section includes a date input set to '2024/09/14', a calendar icon, a time input set to '12:35', and a time zone dropdown set to '(UTC+01:00) Europe/London'. A 'Flexible time window' section has a dropdown menu currently set to 'Off'. Instructions for each field are provided below the inputs.

The Amazon EventBridge schedule is set at 12:35

The screenshot shows the 'RunTask' configuration page for an Amazon ECS target. It includes a 'Universal target definition' toggle. The 'ECS cluster*' section has a dropdown menu showing 'arn:aws:ecs:...' and a 'Create new ECS cluster' link. The 'ECS task*' section has a dropdown menu showing 'HelloWorld' and a 'Create new ECS task' link. Below these, there are radio buttons for 'Latest' and 'Revision' (which is selected). A dropdown menu for the task version is set to '5'. The 'Task count' section has an input field set to '1'.

The version is set at 5.

HelloWorld (1/19) Info

Deploy

Actions

Create new revision

Filter task definition revisions by value

Filter statusActive

Deregister

Delete

< 1 >

| Task definition: revision | Status |
|--|--------|
| <input type="checkbox"/> HelloWorld:16 | ACTIVE |
| <input type="checkbox"/> HelloWorld:15 | ACTIVE |
| <input type="checkbox"/> HelloWorld:14 | ACTIVE |
| <input type="checkbox"/> HelloWorld:13 | ACTIVE |
| <input type="checkbox"/> HelloWorld:12 | ACTIVE |
| <input type="checkbox"/> HelloWorld:11 | ACTIVE |
| <input type="checkbox"/> HelloWorld:10 | ACTIVE |
| <input type="checkbox"/> HelloWorld:9 | ACTIVE |
| <input type="checkbox"/> HelloWorld:8 | ACTIVE |
| <input type="checkbox"/> HelloWorld:7 | ACTIVE |
| <input type="checkbox"/> HelloWorld:6 | ACTIVE |
| <input checked="" type="checkbox"/> HelloWorld:5 | ACTIVE |

The selected version is deregistered.

Deregister

X

When a task definition revision is deregistered, the revision transitions to an INACTIVE state. Existing tasks and services that use the inactive task definition revision continue to run without disruption.

Inactive revisions can't be used to run new tasks or create new services, and you can't update an existing service to use an inactive revision.

Are you sure you want to deregister the following task definition:revision?

Task definitions (1)

The following task definition revision is going to be deregistered.

< 1 >

Task definition: revision

HelloWorld:5

Cancel

Deregister

Amazon SQS > Queues

Queues (1)

Edit

Delete

Send and receive messages

Actions

Create queue

Search queues by prefix

< 1 >

| Name | Type | Created | Messages available | Messages in flight | Encryption | Content-based deduplication |
|---|----------|------------------------|--------------------|--------------------|--------------------------|-----------------------------|
| <input type="radio"/> EventBridgeSigTermDLQ | Standard | 2024-09-12T18:31:01:00 | 2 | 0 | Amazon SQS key (SSE-SQS) | - |

SQS Message available after Amazon EventBridge failed to invoke the Amazon ECS Fargate task

There is 2 messages available as redrive is implemented after failure.

7. Clean Up