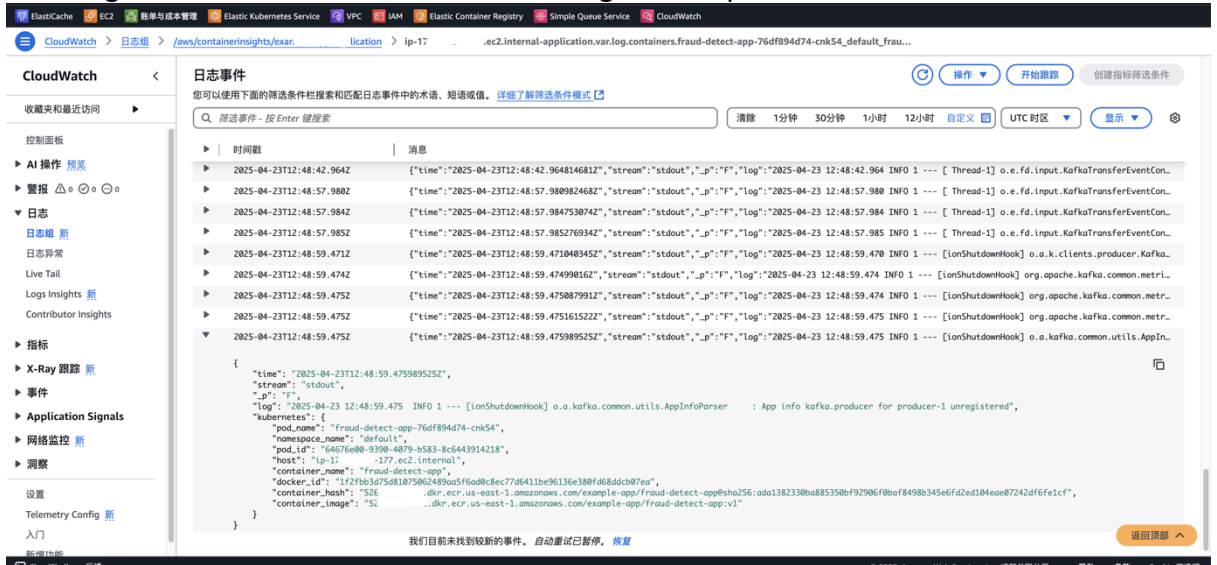


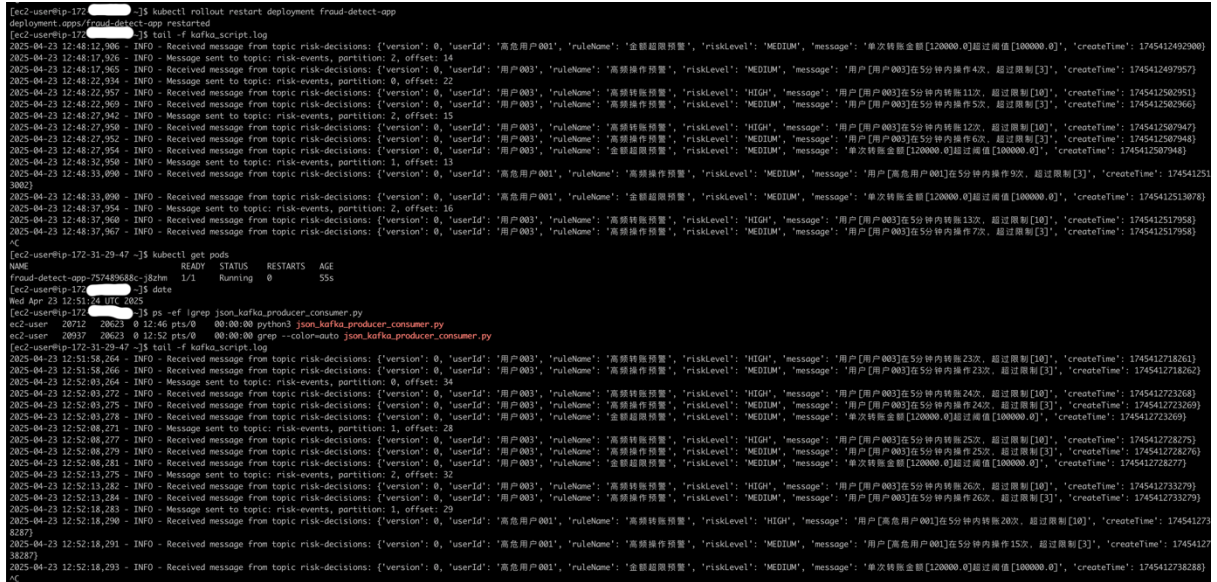
# Resilience Test Report

Test case for pod restart:

1. Modified script json\_kafka\_producer\_consumer.py to send message in a infinite loop. Uploaded the test script json\_kafka\_producer\_consumer.py to the Test Server. Started the test.
2. Reviewed the CloudWatch logs and confirmed the service was running properly by checking the decision results in the console logs before pod restart.



3. Restarted the pod and confirmed the client logs were printed as expected.



4. Checked the server logs in CloudWatch logs again. Confirmed that the pod ID was different indicating the service was re-deployed successfully. Confirmed that the decision results were published expectedly after the pod was restarted and the messages piled up in the message queue backlog during the restarting period were

consumed again.

The screenshot shows the AWS CloudWatch console interface. The left sidebar contains navigation options like '控制面版', 'AI 操作 预览', '警报', '日志', 'Live Tail', 'Logs Insights', 'Contributor Insights', '指标', 'X-Ray 跟踪', '事件', 'Application Signals', '网络监控', and '洞察'. The main content area is titled '日志事件' (Log Events) and displays a list of log entries for the 'fraud-detect-app' service. The log entries show timestamps and log messages, including a detailed log entry for a Kafka message processing event. The log entry for '2025-04-23 12:58:43.166Z' contains a JSON object with fields like 'time', 'stream', 'p', 'log', 'riskLevel', 'message', 'kubernetes', 'pod\_name', 'namespace\_name', 'pod\_id', 'host', 'container\_name', 'docker\_id', 'container\_hash', and 'container\_image'.

5. Note that the service k8s deployment is set with readinessProbe and livenessProbe. The pod will restart automatically if the service is unhealthy.