Audit Log GCS Sink Service Operations Guide

1. Useful Links

List of all the frequently used system related documents link

Name	Link	Description
ADT	API Logs for Product Metrics - ADT	Design Document Of Product Metrics.
GCS Sink Service	Kafka Connect GCS Sink Service	GCS Sink Service Document.
Integrating Audit Logging Service	Audit Logging Service Integration Guide	Document to integrate the audit logging service.
Visualising Product Metrics	Visualising Product Metrics	Data discovery access and accessing data stored in GCS via data discovery.
KCAAS DX overview	Kafka connect overview link	Kafka connect as a service overview link

---

2. Connector APIs

List the APIs exposed by the connector along with usage, sample payloads, and response structures.

Curl	Description	Sample Response

Cluster / **Curl**
EUS: curl --location 'http://audit-log-gcs-sink.data-ventures-luminate-cperf.eus2-prod-a30.cluster.k8s.us.walmart.net/connectors/audit-log-gcs-sink-connector/status'
SCUS: curl --location 'http://audit-log-gcs-sink.data-ventures-luminate-cperf.scus-prod-a63.cluster.k8s.us.walmart.net/connectors/audit-log-gcs-sink-connector/status'
	Status of the connector	{
		"connector": {
		"state": "RUNNING",
		"worker_id": "11.19.25.41:8083"
		},
		"name": "audit-log-gcs-sink-connector",
		"type": "sink",
		"tasks": [
		{
		"id": 0,
		"state": "RUNNING",
		"worker_id": "11.19.25.41:8083"
		}
		]
		}

Cluster	Curl
EUS	curl --location 'http://audit-log-gcs-sink.data-ventures-luminate-cperf.eus2-prod-a30.cluster.k8s.us.walmart.net/connectors/audit-log-gcs-sink-connector/config'
SCUS	curl --location 'http://audit-log-gcs-sink.data-ventures-luminate-cperf.scus-prod-a63.cluster.k8s.us.walmart.net/connectors/audit-log-gcs-sink-connector/config'

Configurations of the connector


```
{
"connector.class": "io.lenses.streamreactor.connect.gcp.storage.sink.GCPStorageSinkConnector",
"errors.log.include.messages": "true",
"build.id": "0.0.289",
"transforms.InsertRollingRecordTimestamp.timezone": "GMT",
"tasks.max": "1",
"timezone": "GMT",
"topics": "api_logs_audit_prod",
"connect.gcpstorage.gcp.auth.mode": "FILE",
"connect.gcpstorage.gcp.file": "*****",
"transforms": "InsertRollingRecordTimestamp, FilterUS",
"locale": "en",
"errors.deadletterqueue.context.headers.enable": "true",
"transforms.InsertRollingRecordTimestamp.type": "io.lenses.connect.smt.header.InsertRollingRecordTimestampHeaders",
"connect.gcpstorage.gcp.project.id": "wmt-dv-luminate-prod",
"connect.gcpstorage.kcql": "INSERT INTO `audit-api-logs-us-prod:us_dv_audit_log_prod.db/api_logs` SELECT * FROM `api_logs_audit_prod` PARTITIONBY service_name, _header.date, endpoint_name STOREAS `PARQUET` PROPERTIES(\n 'flush.size'='50000000',\n 'flush.count'='5000',\n 'flush.interval'='600',\n 'key.suffix'='_eus2'\n );\n",
"transforms.FilterUS.type": "com.walmart.audit.log.sink.converter.AuditLogSinkUSFilter",
"errors.deadletterqueue.topic.name": "api_logs_audit_prod_DLQ",
"connect.gcpstorage.indexes.name": ".indexes-eus2",
"name": "audit-log-gcs-sink-connector",
"errors.tolerance": "all",
"transforms.InsertRollingRecordTimestamp.date.format": "yyyy-MM-dd",
"errors.log.enable": "true"
}
```


Cluster	Curl
EUS	curl --location 'http://audit-log-gcs-sink.data-ventures-luminate-cperf.eus2-prod-a30.cluster.k8s.us.walmart.net/connectors/restart_all'
SCUS	curl --location 'http://audit-log-gcs-sink.data-ventures-luminate-cperf.scus-prod-a63.cluster.k8s.us.walmart.net/connectors/restart_all'

Restarting the connectors


```
{
"connector.class": "io.lenses.streamreactor.connect.gcp.storage.sink.GCPStorageSinkConnector",
"errors.log.include.messages": "true",
"build.id": "0.0.289",
"transforms.InsertRollingRecordTimestamp.timezone": "GMT",
"tasks.max": "1",
"timezone": "GMT",
"topics": "api_logs_audit_prod",
"connect.gcpstorage.gcp.auth.mode": "FILE",
"connect.gcpstorage.gcp.file": "*****",
"transforms": "InsertRollingRecordTimestamp, FilterUS",
"locale": "en",
"errors.deadletterqueue.context.headers.enable": "true",
"transforms.InsertRollingRecordTimestamp.type": "io.lenses.connect.smt.header.InsertRollingRecordTimestampHeaders",
"connect.gcpstorage.gcp.project.id": "wmt-dv-luminate-prod",
"connect.gcpstorage.kcql": "INSERT INTO `audit-api-logs-us-prod:us_dv_audit_log_prod.db/api_logs` SELECT * FROM `api_logs_audit_prod` PARTITIONBY service_name, _header.date, endpoint_name STOREAS `PARQUET` PROPERTIES(\n 'flush.size'='50000000',\n 'flush.count'='5000',\n 'flush.interval'='600',\n 'key.suffix'='_eus2'\n );\n",
"transforms.FilterUS.type": "com.walmart.audit.log.sink.converter.AuditLogSinkUSFilter",
"errors.deadletterqueue.topic.name": "api_logs_audit_prod_DLQ",
"connect.gcpstorage.indexes.name": ".indexes-eus2",
"name": "audit-log-gcs-sink-connector",
"errors.tolerance": "all",
"transforms.InsertRollingRecordTimestamp.date.format": "yyyy-MM-dd",
"errors.log.enable": "true"
}
```

Cluster	Curl
EUS	curl --location 'http://audit-log-gcs-sink.data-ventures-luminate-cperf.eus2-prod-a30.cluster.k8s.us.walmart.net/connectors/log_all/TRACE'
SCUS	curl --location 'http://audit-log-gcs-sink.data-ventures-luminate-cperf.scus-prod-a63.cluster.k8s.us.walmart.net/connectors/log_all/TRACE'

Update the logging level for the connectors. Available levels - DEBUG, ERROR, INFO, WARN

[
"com.walmart.audit.log.sink.converter.AuditLogSinkUSFilter",
"com.walmart.audit.log.sink.converter.CustomHeaderConverter",
"io.confluent.connect.avro.AvroConverterConfig",
"io.confluent.connect.avro.AvroData",
"io.confluent.connect.avro.AvroDataConfig",
"io.confluent.kafka.schemaregistry.rules.NoneAction",
"io.confluent.kafka.serializers.AbstractKafkaSchemaSerDe",
"io.confluent.kafka.serializers.KafkaAvroDeserializerConfig",
"io.confluent.kafka.serializers.KafkaAvroSerializerConfig",
"io.lenses.streamreactor.common.errors.ErrorPolicy$",
"io.lenses.streamreactor.common.errors.ThrowErrorPolicy",
"io.lenses.streamreactor.common.util.AsciiArtPrinter",
"io.lenses.streamreactor.connect.cloud.common.sink.seek.IndexManager",
"io.lenses.streamreactor.connect.cloud.common.sink.transformers.TopicsTransformers$",
"io.lenses.streamreactor.connect.cloud.common.sink.writer.NoWriter",
"io.lenses.streamreactor.connect.cloud.common.sink.writer.Uploading",
"org.apache.kafka.common.utils.Utils",
"org.apache.kafka.connect.cli.AbstractConnectCli",
"org.apache.kafka.connect.connector.policy.AllConnectorClientConfigOverridePolicy",
"org.apache.kafka.connect.connector.policy.NoneConnectorClientConfigOverridePolicy",
"org.apache.kafka.connect.connector.policy.PrincipalConnectorClientConfigOverridePolicy",
"org.apache.kafka.connect.data.Values$ValueParser",
"org.apache.kafka.connect.json.JsonConverterConfig",
"org.apache.kafka.connect.mirror.MirrorCheckpointConnector",
"org.apache.kafka.connect.mirror.MirrorSourceConnector",
"org.apache.kafka.connect.rest.basic.auth.extension.BasicAuthSecurityRestExtension",
"org.apache.kafka.connect.runtime.AbstractHerder",
"org.apache.kafka.connect.runtime.WorkerConnector",
"org.apache.kafka.connect.runtime.WorkerInfo",
"org.apache.kafka.connect.runtime.WorkerSinkTask",
"org.eclipse.jetty.util.log",
"org.eclipse.jetty.util.thread.QueuedThreadPool",
"org.eclipse.jetty.util.thread.ReservedThreadExecutor",
"org.eclipse.jetty.util.thread.ShutdownThread",
"org.eclipse.jetty.util.thread.ThreadPoolBudget",
"org.eclipse.jetty.util.thread.strategy.EatWhatYouKill",
"org.reflections",
"org.reflections.Reflections",
"root",
"shaded.parquet.org.apache.thrift.transport.TIOStreamTransport"
]

3. Logs and Health Metrics

How to access logs (e.g., K8s) and what metrics to monitor (e.g., consumer lag, memory utilization).

Name	Description
Topic **Cluster** / **Link** **EUS** — api_logs_audit_prod - EUS **SCUS** — api_logs_audit_prod - SCUS	**api_logs_audit_prod** is the topic in prod EUS and SCUS cluster. Separate topics are present in both cluster.

Consumer group

Cluster	Link
EUS	connect-audit-log-gcs-sink-connector - EUS
SCUS	connect-audit-log-gcs-sink-connector - SCUS

connect-audit-log-gcs-sink-connector is the consumer group in prod EUS and SCUS cluster. Separate consumer groups are present in both cluster. We have alerts set up. When consumer lag is more than **50K we get WARNING alert**. When it is more than **75K we get critical alert.**
Flush properties are as follows. Here once any of the threshold are met the connector will start writing the record on to GCS

Flush property	Threshold
size	50 MB
interval	10 Mins
count	5000 records

K8s Dashboard

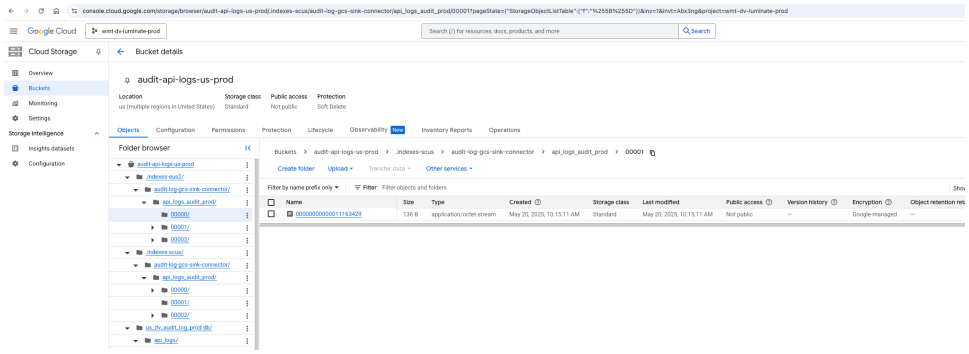
Cluster	Link
EUS	K8s Dashboard EUS - audit-log-gcs-sink
SCUS	K8s Dashboard SCUS - audit-log-gcs-sink

K8s dashboard in EUS and SCUS cluster. Service name is **audit-log-gcs-sink.** This can be used to monitor and download the logs. We can also monitor the pod health at that time. For a particular time range metrics Grafana is the go to dashboard.

GCS Bucket US - audit-api-logs-us-prod

Sink service stores the records in this GCS bucket. We can check here for latest updates on offsets commits based on **Last modified** time. Actual records will be stored in respective partitions
EUS offsets maintained here by connector **audit-api-logs-us-prod/.indexes-eus2/audit-log-gcs-sink-connector /api_logs_audit_prod/00000**

SCUS offsets maintained here by connector **audit-api-logs-us-prod/.indexes-scus/audit-log-gcs-sink-connector /api_logs_audit_prod/00001**



Grafana Golden Signals - Grafana Golden Signals

Application metrics can be found here i.e, health, latency, memory utilization etc.. Any deviation here then we need to take appropriate config updates in **kitt.yaml .**

We can select the appropriate clusters in the dropdown.

Kafka Broker Metrics - Managed Broker Metrics

Metrics related to Kafka broker can be found here. This provides a very high level analysis at the broker level

4. Troubleshooting the Issues

Common issues and how to debug them using logs, configurations, or metrics. There are chances the lag is due to rebalancing, connector being killed because of an error, GCS not responding etc.. Here are the standard steps to be followed in order to address any issue

<table>
<tr><th>Step</th><th>Description</th><th>Action</th></tr>
<tr><td>1</td><td>Consumer groups analysis<table><tr><th>Cluster</th><th>Link</th></tr><tr><td>EUS</td><td>connect-audit-log-gcs-sink-connector - EUS</td></tr><tr><td>SCUS</td><td>connect-audit-log-gcs-sink-connector - SCUS</td></tr></table></td><td>Check the logs in K8s for any exception, error, memory related errors. Analyse the consumer groups. Is it in rebalancing state ? Inactive state ? Some underlying infra changes too can cause consumer rebalancing. If the application is not able to heal from here then proceed to next steps.</td></tr>
<tr><td>2</td><td>Check the connector status, config. Set of APIs to support this analysis
Refer pointer 2. APIs of the Connector</td><td>Check the connector task status. Due to some internal errors the connector task can be killed and it may require restart as well. Even if the consumer is rebalancing for more than 10 minutes then better to restart the connector task. Enable the TRACE log level. On restarting wait for 10 minutes to see if consumer is able to rejoin the group and start consuming the message (Lag should be reduced / We can see latest offsets commit as GCS bucket). If it fails again then health related metrics to be checked.</td></tr>
<tr><td>3</td><td>Grafana golden signals analysis - Grafana Golden Signals</td><td>We can get to know the health metics of the connector here. Check for memory utilization, CPU utilization and take appropriate action updating configs in kitt.yaml</td></tr>
<tr><td>4</td><td>Heap dump and Heap memory related analysis</td><td>On analysing the heap dump / if in case of heap memory error try to update the Xmx, Xms in property KAFKA_HEAP_OPTS of kitt.yaml</td></tr>
</table>

---

5. Issues Observed in Prod

List of known or recurring issues observed in production with logs and mitigation steps.

Issue	Description	Logs	Resolution	Current config		
Consumer poll() time out	This can occur when consumer is failing to poll records in the stipulated interval. The four main properties for this to look for and update is **max.poll.records, max.poll.interval.ms, consumer.max.poll.records, consumer.max.poll.interval.ms**	[2025-05-14 15:02:15,417] WARN [audit-log-gcs-sink-connector|task-1] [Consumer clientId=connector-consumer-audit-log-gcs-sink-connector-1, groupId=connect-audit-log-gcs-sink-connector] consumer poll timeout has expired. This means the time between subsequent calls to poll() was longer than the configured max.poll.interval.ms, which typically implies that the poll loop is spending too much time processing messages. You can address this either by increasing max.poll.interval.ms or by reducing the maximum size of batches returned in poll() with max.poll.records.	To increase the worker consumer's **max.poll.interval.ms** and decrease the **max.poll.records.** Keep the same values for connector's instance i.e, **consumer.max.poll.records, consumer.max.poll.interval.ms**	Found in file -> kc_config.yaml # For worker consumer max.poll.records: 50 max.poll.interval.ms: 300000 heartbeat.interval.ms: 5000 session.timeout.ms: 15000 request.timeout.ms: 60000 # For connector instance consumer.max.poll.records: 50 consumer.max.poll.interval.ms: 300000 consumer.heartbeat.interval.ms: 5000 consumer.session.timeout.ms: 15000 consumer.request.timeout.ms: 60000		

java.lang.OutOfMemoryError: Java heap space	When there is data which can take more heap memory / memory leaks we can encounter this. Due to this the pods can be restarting intermittently.	Failed to deserialize value for header 'x-edgeconnect-session-id' on topic 'api_logs_audit_prod', so using byte array (org.apache.kafka.connect.storage.SimpleHeaderConverter:72) java.lang.OutOfMemoryError: Java heap space	Address if any memory leaks. Increase the **Xmx**, **Xms** in property **KAFKA_HEAP_OPTS** of **kitt.yaml**	`KAFKA_HEAP_OPTS: "-Xmx7g -Xms5g -XX: MetaspaceSize=96m -XX: +UseG1GC -XX: MaxGCPauseMillis=20 -XX: InitiatingHeapOccupancyPercent=35 -XX: G1HeapRegionSize=16M -XX: MinMetaspaceFreeRatio=50 -XX: MaxMetaspaceFreeRatio=80 -XX: +UseStringDeduplication"`		
GCS storage exception / Connector killed for any errors	There can be chances the GCS is not responding / timeout in getting response from GCS. By default the connector is killed and it requires manual restarts.	[2025-05-16 10:50:14,613] ERROR [audit-log-gcs-sink-connector|task-0] [audit-log-gcs-sink-connector - 1 of 1] Failed upload from data string (UploadableString(us_dv_audit_log_prod.db/api_logs/service_name=NRT/date=2025-05-16/endpoint_name=nrti_status/api_logs_audit_prod(0_000011006149)_eus2.parquet)) to Storage audit-api-logs-us-prod:.indexes-eus2/audit-log-gcs-sink-connector/api_logs_audit_prod/00000/00000000000011006149 (io.lenses.streamreactor.connect.gcp.storage.storage.GCPStorageStorageInterface:165) com.google.cloud.storage.StorageException: Unexpected end of file from server at com.google.cloud.storage.StorageException.translate(StorageException.java:179) Caused by: java.net.SocketException: Unexpected end of file from server at java.base/sun.net.www.http.HttpClient.parseHTTPHeader (Unknown Source) [2025-05-16 10:50:15,681] ERROR [audit-log-gcs-sink-connector|task-0] WorkerSinkTask {id=audit-log-gcs-sink-connector-0} Task threw an uncaught and unrecoverable exception. Task is being killed and will not recover until manually restarted. Error: org.apache.kafka.connect.errors.ConnectException: fatal:	Default behaviour of connector is to not retry when there is error and task will be killed. Check for task status and just restart the task. Currently we have update the default task behaviour to retry as mentioned here. If no succcess in that time then we need to restart the task which gets killed.	connect.gcpstorage.error.policy: RETRY connect.gcpstorage.max.retries: 5 connect.gcpstorage.retry.interval: 5000 connect.gcpstorage.http.retry.timeout.multiplier: 1.0		