SQS

Amazon CloudWatch Logs is a service provided by Amazon Web Services (AWS) that enables you to monitor, store, and access log files from various AWS resources and applications. Here are some key points about AWS CloudWatch Logs

Log Aggregation: CloudWatch Logs allows you to aggregate logs from different sources, including AWS services (such as Amazon EC2 instances, AWS Lambda functions, and Amazon RDS databases) as well as custom applications running on-premises or in the cloud.

Centralized Log Storage: Logs are stored centrally in CloudWatch Logs, making it easier to search, analyze, and retrieve log data. You can access logs using the AWS Management Console, CLI (Command Line Interface), SDKs (Software Development Kits), or the CloudWatch Logs API.

Real-time Monitoring: CloudWatch Logs provides real-time log monitoring capabilities, allowing you to set up metric filters and alarms based on log events. You can define custom metrics and trigger actions or notifications based on specific log patterns or events.

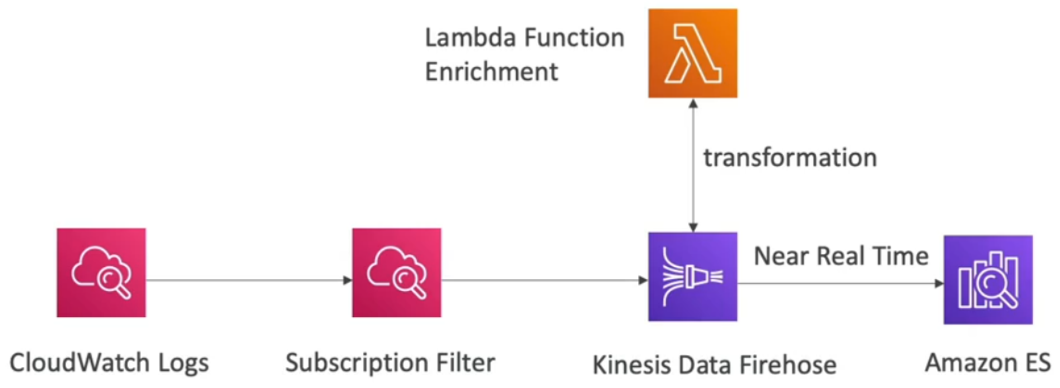
Log Retention: You can specify the retention period for log data in CloudWatch Logs. Log data can be stored for a specific duration, ranging from a few days to several years, depending on your requirements.

Integration with Other AWS Services: CloudWatch Logs integrates with other AWS services, enabling you to use log data for various purposes. For example, you can set up log-based alarms to trigger Amazon SNS notifications or invoke AWS Lambda functions based on specific log events.

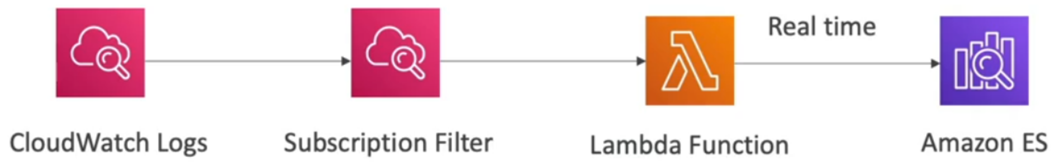
Log Insights: CloudWatch Logs provides Log Insights, a feature that allows you to run advanced queries and perform interactive analysis on log data using the CloudWatch Logs Insights Query Language. This helps you identify patterns, troubleshoot issues, and gain insights from your log data.

Log Export: CloudWatch Logs allows you to export log data to other services for further processing or archival purposes. For example, you can export log data to Amazon S3, Amazon Elasticsearch Service, or Amazon Kinesis Data Firehose

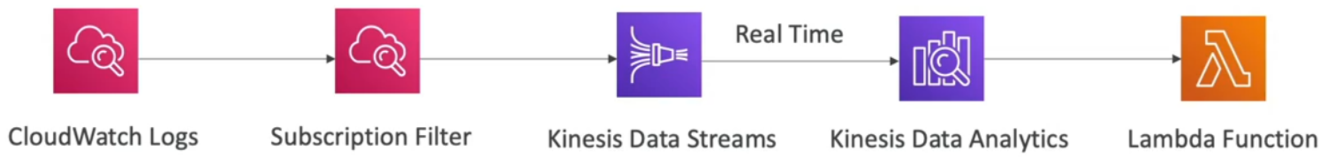
CloudWatch Logs Subscription Filter Patterns Near Real Time into Amazon ES



CloudWatch Logs Subscription Filter Patterns Real Time Load into Amazon ES



CloudWatch Logs Subscription Filter Patterns Real Time Analytics



Subscription Filter is a feature that allows you to specify a filter pattern for a log group and define a destination for the matching log events. It provides a way to selectively stream log events from CloudWatch Logs to other AWS services or external systems for further processing or analysis

Filter Pattern: You define a filter pattern using pattern syntax. The filter pattern can include various operators, wildcards, and placeholders to match specific log events based on their content. For example, you can filter events based on keywords, specific fields, or regular expressions.

Destination: Once you have defined the filter pattern, you specify a destination for the matching log events. The destination can be another AWS service, such as Amazon Kinesis Data Firehose, AWS Lambda, or an Amazon Simple Notification Service (SNS) topic. Alternatively, you can choose to send the events to a Lambda function in a different AWS account or even to a third-party system using a custom endpoint.

Processing and Analysis: The log events that match the filter pattern are sent to the specified destination for further processing or analysis. This allows you to perform actions such as storing logs in a data lake, indexing them in a search engine, triggering notifications, or invoking custom logic for real-time processing.

SQS

sqs is not a object transfer just transfer strings what ever you send it will converted into strings

Amazon Simple Queue Service (SQS) is a fully managed message queuing service provided by Amazon Web Services (AWS). It enables you to decouple the components of your applications by allowing them to communicate asynchronously through messages. Here are some key points about SQS:

Message Queue: SQS provides a reliable and scalable message queuing system. Messages are stored in a queue until they are consumed by the receiving component. This decoupling allows components to work independently and at their own pace.

Fully Managed: SQS is a fully managed service, which means that AWS takes care of the underlying infrastructure, scaling, and maintenance of the message queue. You don't need to provision or manage servers to use SQS.

Distributed and Highly Available: SQS is designed to be distributed and highly available. Messages are stored redundantly across multiple availability zones, ensuring durability and fault tolerance.

Standard Queue and FIFO Queue: SQS provides two types of queues: Standard Queue and FIFO (First-In-First-Out) Queue. Standard Queue offers high throughput and best-effort ordering of messages, while FIFO Queue guarantees that the order of messages is preserved and that each message is processed exactly once.

Message Retention: Messages stored in SQS queues are retained for a configurable period of time. This allows consumers to retrieve and process messages even if they become temporarily unavailable or if there is a temporary spike in message load.

Visibility Timeout: When a message is retrieved from an SQS queue, it becomes invisible to other consumers for a specified duration called the "visibility timeout." This ensures that only one consumer processes a message at a time. If the processing is not completed within the visibility timeout, the message becomes visible again and can be picked up by another consumer.

integration: SQS integrates seamlessly with other AWS services, making it a fundamental component in building scalable and loosely coupled systems. For example, you can integrate SQS with AWS Lambda, Amazon SNS, and Amazon EC2 to create event-driven architectures and distributed systems.

SDKs and APIs: AWS provides SDKs and APIs for various programming languages, making it easy to interact with SQS and integrate it into your applications. You can send, receive, and delete messages using these SDKs and APIs.

SQS is widely used in various scenarios such as decoupling microservices, handling asynchronous workflows, buffering requests, and enabling distributed processing. It helps ensure scalability, reliability, and fault tolerance in your applications by providing a robust messaging infrastructure.

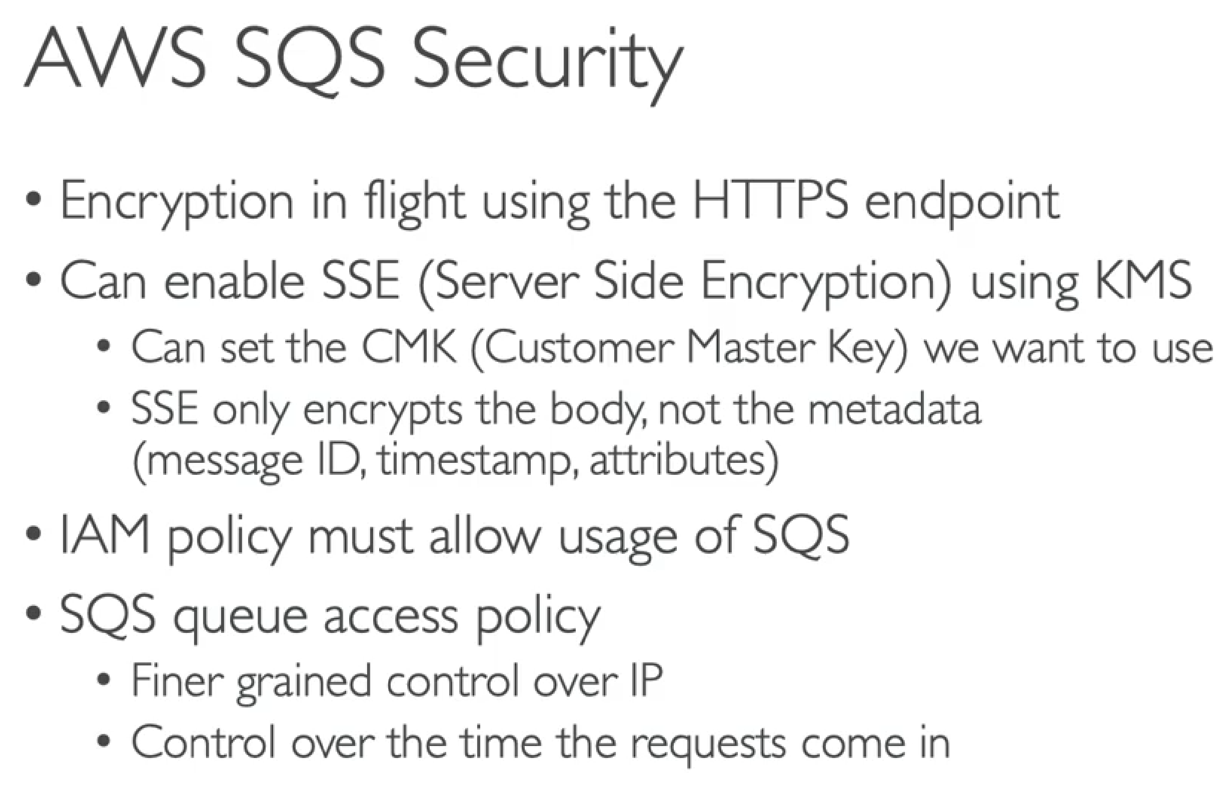
It's important to note that SQS is a fully managed service, which means you pay for the messages sent, received, and stored in the queues, as well as for API requests and data transfer.

you have a 2gb data you can upload usinf sqs but it first save on s3 bucket and add copy link to conusmer to producer

persecond 3000 message per second you can also retrieve the data 4days to 14 days

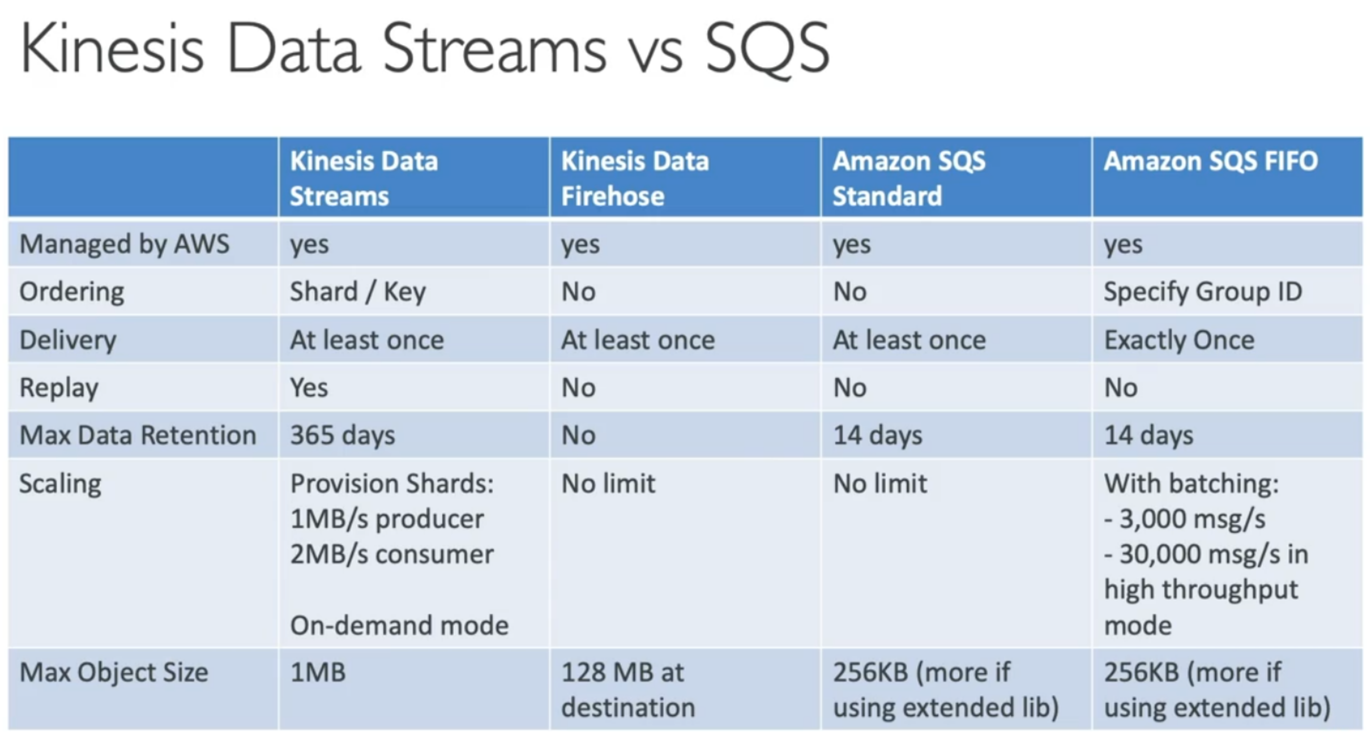
```
import boto3

# Create an SQS client
sqs = boto3.client('sqs')
# Send a message to the queue
response = sqs.send_message(
    QueueUrl='your_queue_url',
    MessageBody='Hello, SQS!'
)
# Receive messages from the queue
response = sqs.receive_message(
    QueueUrl='your_queue_url',
    MaxNumberOfMessages=10  # Number of messages to receive
)

# Process received messages
for message in response['Messages']:
    # Print the message body
    print(message['Body'])
```

```python
# Delete a message from the queue
sqs.delete_message(
    QueueUrl='your_queue_url',
    ReceiptHandle=message['ReceiptHandle']  # Receipt handle of the message
)
```



AWS SQS Security

- Encryption in flight using the HTTPS endpoint
- Can enable SSE (Server Side Encryption) using KMS
  - Can set the CMK (Customer Master Key) we want to use
  - SSE only encrypts the body, not the metadata (message ID, timestamp, attributes)
- IAM policy must allow usage of SQS
- SQS queue access policy
  - Finer grained control over IP
  - Control over the time the requests come in



Kinesis Data Streams vs SQS

	Kinesis Data Streams	Kinesis Data Firehose	Amazon SQS Standard	Amazon SQS FIFO
Managed by AWS	yes	yes	yes	yes
Ordering	Shard / Key	No	No	Specify Group ID
Delivery	At least once	At least once	At least once	Exactly Once
Replay	Yes	No	No	No
Max Data Retention	365 days	No	14 days	14 days
Scaling	Provision Shards: 1MB/s producer 2MB/s consumer On-demand mode	No limit	No limit	With batching: - 3,000 msg/s - 30,000 msg/s in high throughput mode
Max Object Size	1MB	128 MB at destination	256KB (more if using extended lib)	256KB (more if using extended lib)

SQS vs Kinesis – Use cases

- SQS use cases:
  - Order processing
  - Image Processing
  - Auto scaling queues according to messages.
  - Buffer and Batch messages for future processing.
  - Request Offloading

- Kinesis Data Streams use cases:
  - Fast log and event data collection and processing
  - Real Time metrics and reports
  - Mobile data capture
  - Real Time data analytics
  - Gaming data feed
  - Complex Stream Processing
  - Data Feed from "Internet of Things"