# Application Services

## SQS – Simple Queue Service

* SQS is a distributed web service that gives you access to a message queue that can be used to store messages while waiting for a computer to process them.
* Amazon SQS is a distributed queue system that enables web service apps to quickly and reliably queue messages
* A Queue is a temporary repository for messages that are waiting processing
* SQS helps decouple the components of an application so they can run independently.
* Messages can be retrieved via SQS API
* FIFO (first-in-first-out) queues - \*preserve - the exact order in which messages are sent and received. If you use a FIFO queue, you don't have to place sequencing information in your messages. \*Standard queues - provide a \*loose-FIFO capability - that attempts to preserve the order of messages. However, because standard queues are designed to be massively scalable using a highly distributed architecture, receiving messages in the exact **order they are sent is not guaranteed**.
* Standard queues provide at-least-once delivery, which means that each message is delivered at least once.
* One or more producers can send messages to a FIFO queue. Messages are stored in the order that they were successfully received by Amazon SQS.
* SQS common use case is a distributed, decoupled application whose multiple components and modules need to communicate with each other, but can’t do the same amount of work simultaneously.
* Batch operations (SendMessageBatch, DeleteMessageBatch, and ChangeMessageVisibilityBatch) all cost the same as other Amazon SQS requests. By grouping messages into batches, you can reduce your Amazon SQS costs.
* Some AWS or external services that send notifications to Amazon SQS might not be compatible with FIFO queues
* SQS message queues can receive notifications from Amazon SNS topics.
* Deliver same message to multiple SQS Queues – by creating a SNS Topic. And then have multiple SQS queues subscribe to the topic. A message published to a SNS Topic will be delivered to all SQS queues by SNS
* You can delete all messages in an Amazon SQS message queue using the PurgeQueue action, while retaining the queue and its attributes.
* For security, encrypt messages before they are placed in Queue
* Messages deleted from FIFO queues are never seen / introduced again. On rare occasions, this might happen in standard queues.
* When you issue a DeleteMessage request on a previously-deleted message, Amazon SQS returns a success response.
* SQS is not HIPAA compliant. Send messages to SQS via S3 which is HIPAA compliant.
* Queue can store unlimited messages. Limit on number of inflight messages
* Queue names have 80 character limit.
* You cannot share SQS messages between regions
* The producer and consumer can run at their own independent throughput.
* The queue acts as a buffer between consumer and producer. Ensures delivery of messages at least once. Ensure your application isn’t affected by processing the same message multiple times.
* Allows multiple readers and writers. Single queue can be used simultaneously by various applications – helps scale out applications
* Use SQS for - *Decoupling the components of an application, configuring individual message delay, dynamically increasing concurrency or throughput at read time, Scaling transparently*
* Does not guarantee FIFO messages. If order is important, add sequencing information in each message.
* Only an AWS account owner (or an AWS account that the account owner has delegated rights to) can perform operations on an Amazon SQS message queue
* Amazon SQS FIFO queues don't serve messages from the same message group to more than one consumer at a time. However, if your FIFO queue has **multiple message groups**, you can take advantage of parallel consumers, allowing Amazon SQS to **serve messages from different message groups to different consumers**.
* Must use a FIFO dead letter queue with a FIFO queue. (Similarly, you can use only a standard dead letter queue with a standard queue.)
* The name of a FIFO queue must end with .fifo suffix. To determine whether a queue is FIFO, you can check whether the queue name ends with the suffix.
* SQS can do auto-scaling. If queue grows beyond a threshold, instantiate new web/app servers. Use Auto scaling + SQS to achieve this.
* Message can contain **up to 10 metadata attributes**.
* SQS is **pull based**.
* You are billed at 64KB Chunks
* SQS Message size **up to 256KB of text in any format**. May consist of 1-10 messages. (Larger via SDK)
* **Message life**: 1 minute 🡪 14 days (default is 4 days)
* **Visibility Timeout**: time message is invisible in the SQS queue after a reader picks up the message- If job is completed, the message is deleted else it will reappear after the time out. **Max 12 hours**
* **SQS long polling:** Does not return a response till there is a message in a queue so it does not keep polling and costing more
* In almost all cases, Amazon SQS long polling is preferable to short polling. Use short polling if a single application thread is polling multiple – queues.
* Types of queues
  + **Standard Queues (Default)**
    - unlimited TPS
    - Guaranteed delivery at least once
    - May deliver more than once
    - Best effort ordering
  + **FIFO Queues (May not be tested)**
    - 300 TPS
    - Guaranteed ordering
    - No duplicates
    - Message available until consumed and deleted

Pricing

* First 1 million SQS Requests per month are free.
* $0.50 per 1 million SQS requests per month thereafter + data transfer charges (unless to lambda or EC2)
* 64KB chunk = 1 request. So, a message of 256KB = 4 requests.

Exam Tip - De-couple ➔ SQS

EC2 instances pull SQS messages from a standard SQS queue on a FIFO (First In First out) basis. – False

## SWS – Simple Workflow Service

* SWS is a web service that makes it easy to coordinate work across distributed application components. Co-ordinate tasks & workflows.
* Amazon SWF API actions are task-oriented. Amazon SQS API actions are message-oriented.
* With SWF – it is a robust platform for development, message passing between tasks
* Amazon uses SWS to process orders on its website.
* No EC2 components involved.
* It can also involve human actors.
* Task is assigned only once and never duplicated
* Workflow and activity types and its execution are scoped into a domain.
* **Domain** isolates a set of types, executions and task lists wfrom other.
* **Domains** can be registered on AWS Mgmt Console or by RegisterDomain action in the SWF API
* **Workers** are programs that interact with SWF to get tasks, process task and return results
* SWS Actors
  + WF Starters – e-commerce application
  + WF Deciders – Control flow of activity tasks.
  + WF Activity workers – Carry out actual task

Trick Question – when to use SQS or SWS

| **Attribute** | **SQS** | **SWS** |
| --- | --- | --- |
| Retention | 14 days | 1 year |
| API | Message Oriented | Task Oriented |
| Assignment | Might be assigned multiple times | Only once |
| State | Write code to implement tracking | Keeps Track of State & Events |

## SNS – Simple Notification Service (not heavily Tested)

* Makes it easy to setup, operate and send notifications from the cloud.
* Immediate delivery to subscribers or other applications
* JSON Formatted
* Publish-Subscribe paradigm: **notifications “pushed”** to subscribers
* SNS consists of Topics and you can publish messages to topics.
* You can send notifications to:
  + Emails
  + SMS
  + SQS queues
  + http endpoint
  + trigger Lambda functions
  + Apple, google, Windows etc.
* Lambda function can then manipulate information and then send to other SNS Topics
* You can group multiple recipients using **topics**. Recipients can subscribe to topics to receive notifications.
* Stored across multiple AZs for redundancy
* Flexible message delivery over multiple protocols.
* Is used in conjunction with CloudWatch and AutoScaling.

## Elastic Transcoder

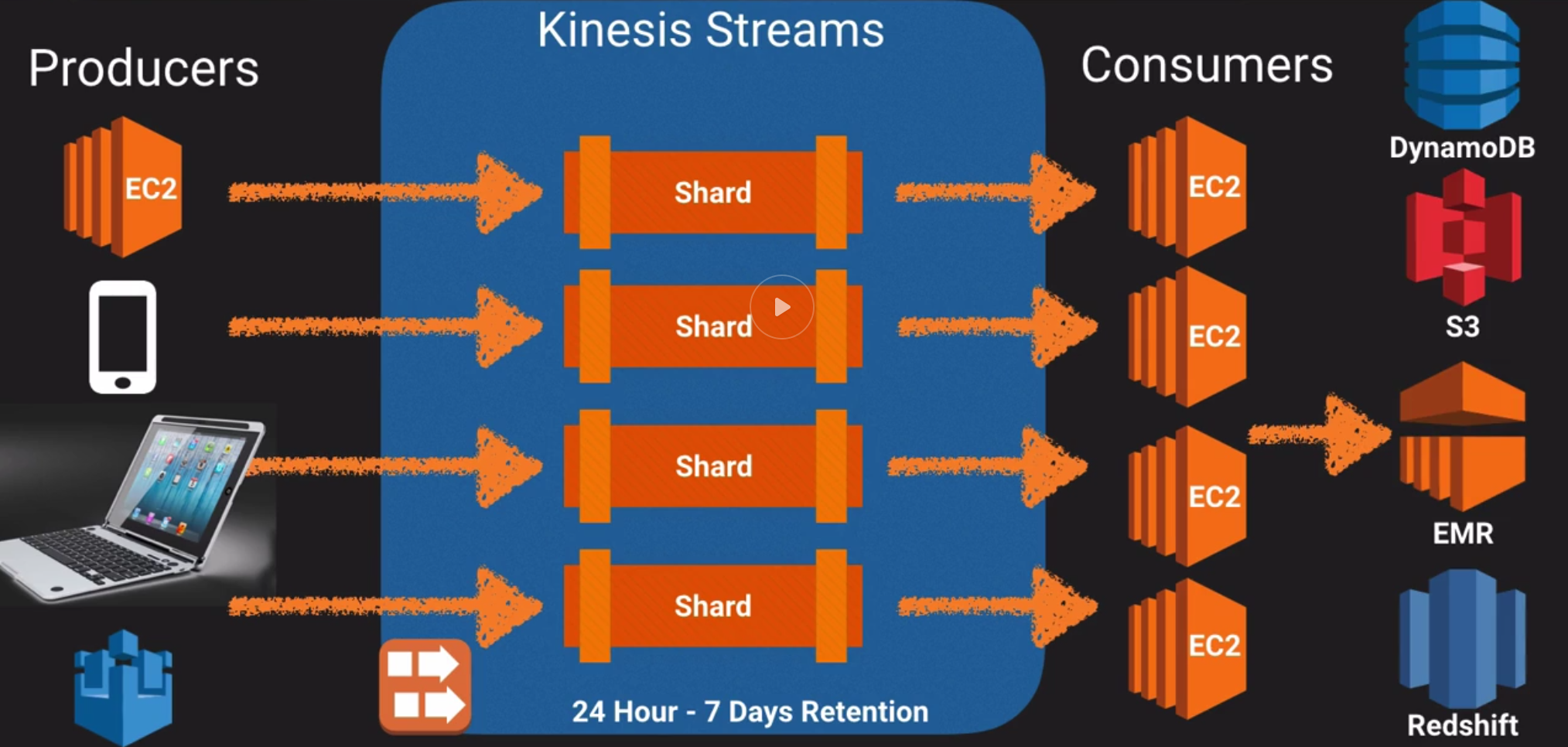
* Allows to convert media files from source to different media formats optimized for the cloud. Don’t have to guess settings for different types
* You pay the minutes you transcode and the resolution
* S3 → Lambda Function → E. Transcoder → S3

## API Gateway

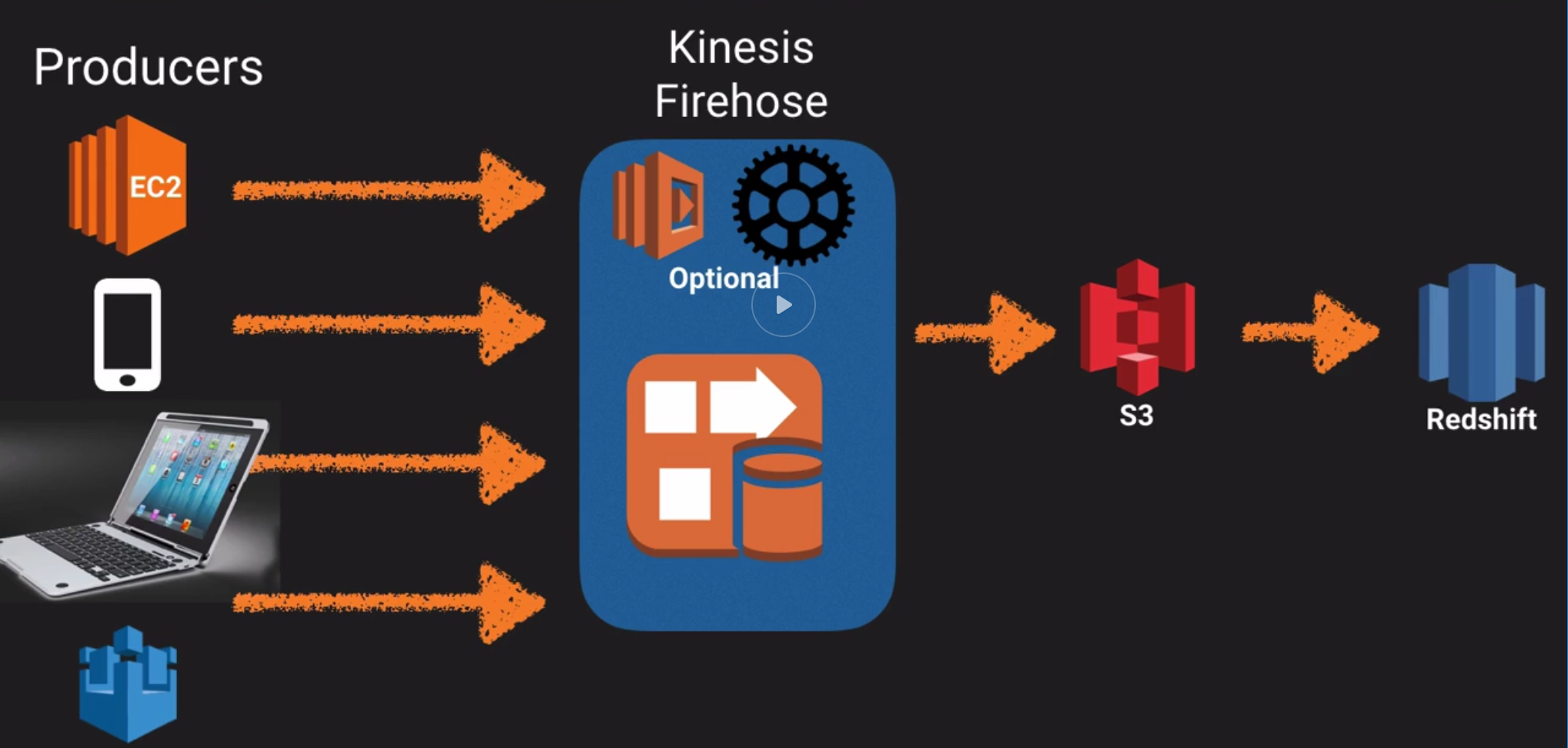
* Managed web service which enables developers to publish, monitor and secure APIs at any scale.
* Create an API that acts as “front door” for applications to access data, business logic or any functionality from your backend services
* **API Caching** – Cache your endpoint’s responses. Reduces load on endpoints based on duration of TTLs
* Low cost & Efficient. Scales automatically
* Throttle requests as required to prevent attacks.
* Log requests to CloudWatch.
* For application built on top of **multiple domains**, **you need to enable CORS on API Gateway**.
* Same Origin Policy: Web browser permits scripts contained in a first web page to access data in a second web page if both have the same origin
* CORS relaxes the requirement
* Allows restricted resources on a web page to be requested from another domain
* *Error: Origin policy cannot be read at the remote resource? – Enable CORS*

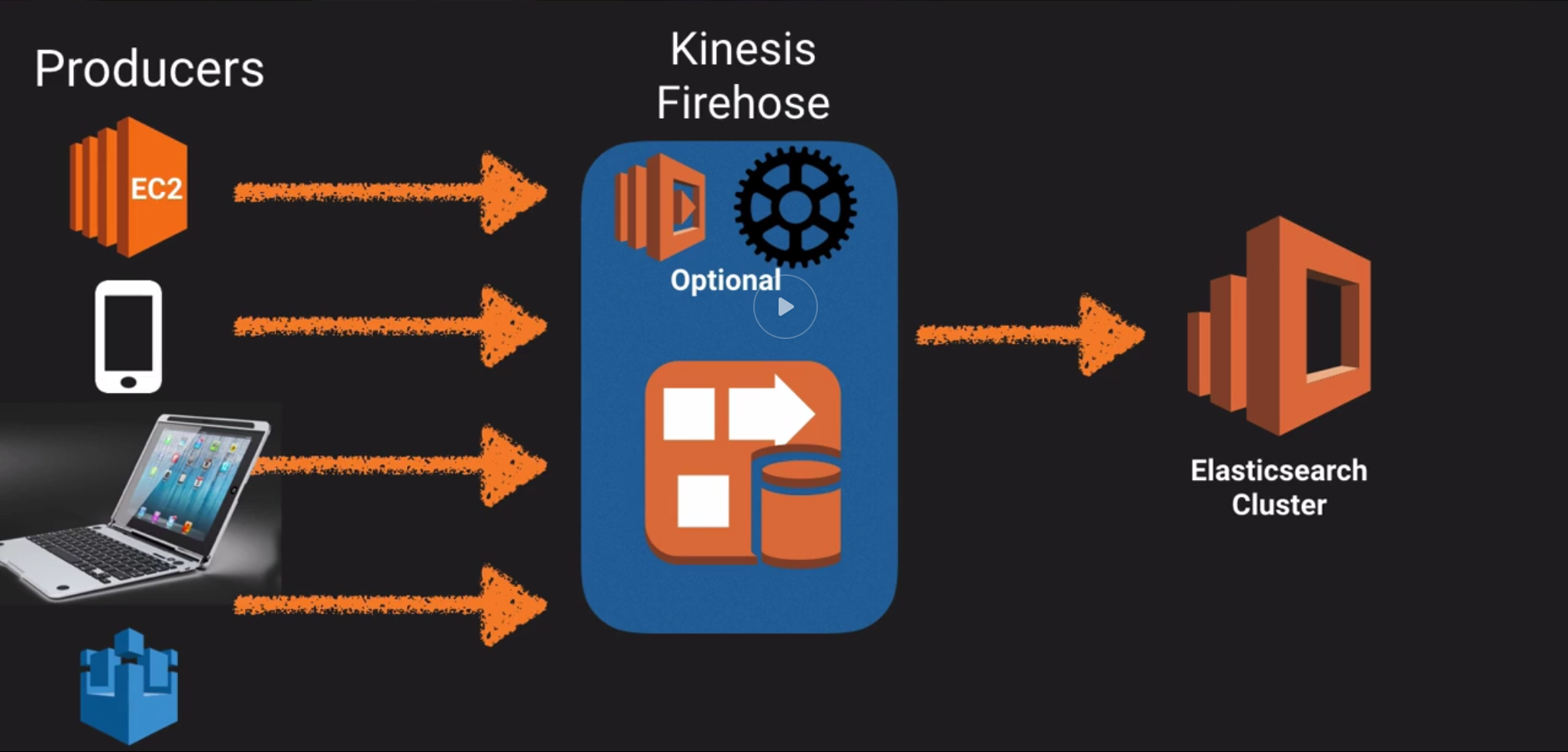
## Amazon Kinesis

* Streaming data is something which is generated by thousands of data sources – stock prices, game information, social network data, geo-spatial data, purchases from online stores, IoT sensor data.
* Kinesis is an AWS platform to load and analyze streaming data and the ability to build your own custom applications
* **Kinesis Streams**

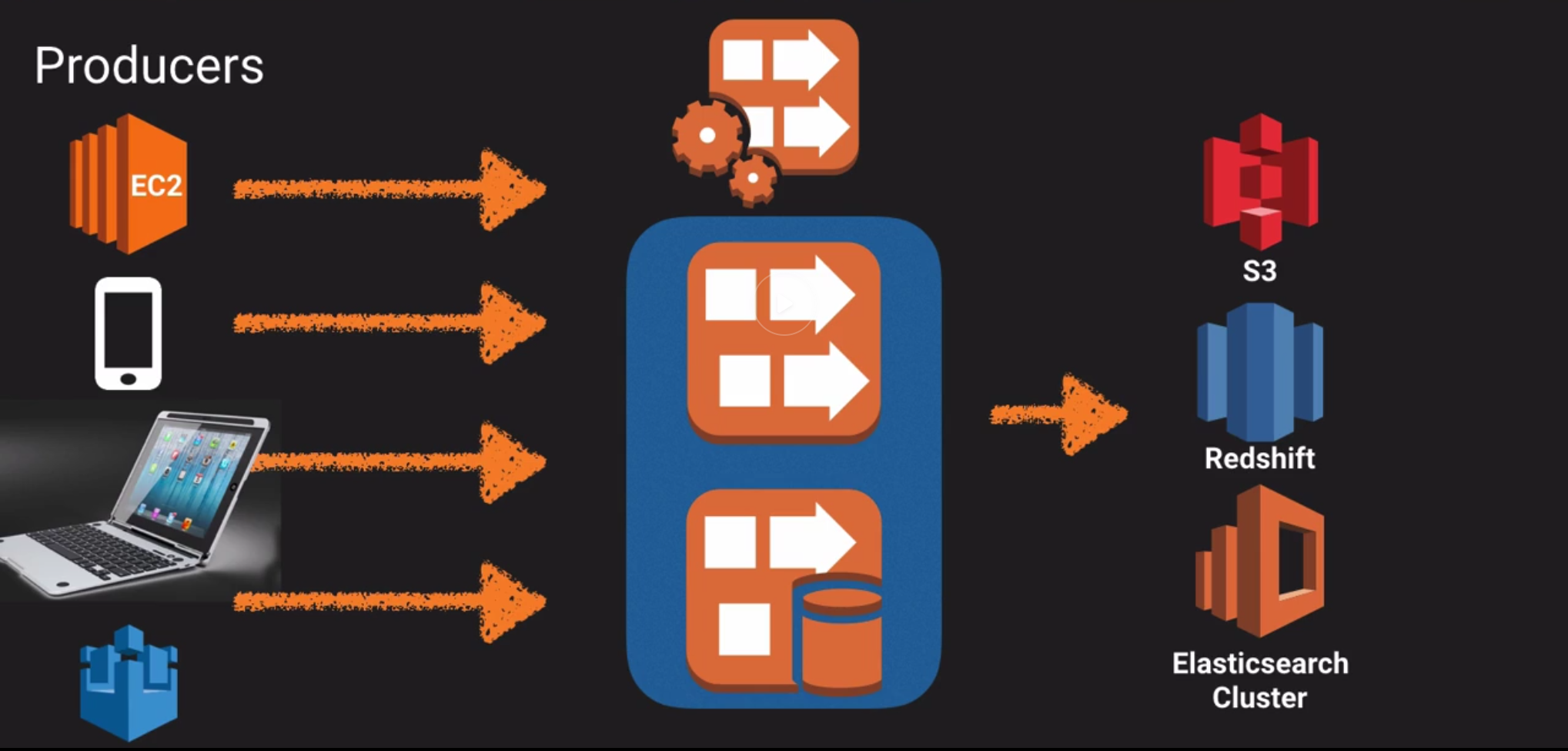


* + Allows real-time processing of streaming big data and the ability to read and replay records to multiple Amazon Kinesis Applications
  + Stores data for 24 hours to 7 days.
  + Data stored in **shards**
    - **5 TPS for Read**
    - **Data reads at 2MB per second**
    - **Write: 1000 records per second at 1MB /s**
  + Data consumers (EC2 instances) analyze the stream and then derive results/take next actions
  + Data capacity of stream is a function of the number of shards you specify for the stream.
* **Kinesis Firehose**





* + Don’t have to worry about shards, streams – completely automated.
  + No automatic data retention window. Data is either immediately analyzed or sent to S3 and then to Redshift, elastic search cluster
  + Data is immediately analyzed via **Lambda**.
* **Kinesis Analytics**



* + Run SQL type queries on top of data contained in Streams or Firehose and store the results in S3 / Redshift and Elastic Search cluster.
* *Exam Tip: Kinesis - process large streams of data. To process data - Amazon Redshift and Elastic Map Reduce*