Media2Cloud: Implementation Guide
Copyright © Amazon Web Services, Inc. and/or its affiliates. All rights reserved.

Amazon's trademarks and trade dress may not be used in connection with any product or service that is not Amazon's, in any manner that is likely to cause confusion among customers, or in any manner that disparages or discredits Amazon. All other trademarks not owned by Amazon are the property of their respective owners, who may or may not be affiliated with, connected to, or sponsored by Amazon.
# Table of Contents

Home ................................................................................................................................. 1
Overview ............................................................................................................................. 2
Cost ..................................................................................................................................... 2
Architecture .......................................................................................................................... 2
Ingest Workflow .................................................................................................................. 3
Analysis Workflow .............................................................................................................. 3
Labeling Workflow ............................................................................................................. 4
Web Interface ..................................................................................................................... 5
Solution Components ........................................................................................................ 7
Ingest Process ..................................................................................................................... 7
Analysis Process ................................................................................................................ 7
Labeling Process ................................................................................................................ 7
Error Handling ................................................................................................................... 8
Proxy Files ........................................................................................................................ 8
Web Interface ..................................................................................................................... 5
Amazon DynamoDB .......................................................................................................... 8
Amazon SNS ....................................................................................................................... 9
Considerations .................................................................................................................... 10
Limits .................................................................................................................................. 10
Custom Sizing .................................................................................................................... 10
Integrated Partners .......................................................................................................... 11
  Cloudfirst.io .................................................................................................................... 11
  Levels Beyond ............................................................................................................... 11
  Nomad CMS .................................................................................................................. 11
  EditShare ....................................................................................................................... 12
  eMAM ............................................................................................................................ 12
  Evertz ............................................................................................................................. 12
  IMT ................................................................................................................................. 12
  Starchive ....................................................................................................................... 13
  Trackit ............................................................................................................................ 13
Regional Deployment ......................................................................................................... 13
Template .............................................................................................................................. 14
Deployment .......................................................................................................................... 15
  What We'll Cover ........................................................................................................... 15
  Step 1. Launch the Stack ................................................................................................. 15
  Step 2. Upload a Video or Image File ............................................................................. 17
  Step 3. Create and View Metadata ............................................................................... 18
  Step 4. Configure the Labeling Work Team ................................................................. 18
  Step 5. Create Your Face Collection ............................................................................ 18
Security ................................................................................................................................ 20
Server-Side Encryption .................................................................................................... 20
Amazon CloudFront ......................................................................................................... 20
Resources ............................................................................................................................ 21
Appendix A: File Paths in Amazon S3 .............................................................................. 22
Appendix B: Ingest State Machine ..................................................................................... 24
Appendix C: Analysis State Machine ................................................................................ 26
  Video Analysis Sub-state Machine ............................................................................. 26
  Audio Analysis Sub-state Machine ............................................................................. 27
  Image Analysis Sub-state Machine ............................................................................. 29
Appendix D: Labeling State Machine ................................................................................ 30
Appendix E: State Machine Error Handling ..................................................................... 31
Appendix F: Lifecycle Policy .............................................................................................. 32
Appendix G: Amazon SNS Notifications .......................................................................... 33
  Ingest State Machine Notification Message ............................................................... 33
Media2Cloud

AWS Implementation Guide

AWS Solutions Builder Team

January 2019 (last update (p. 47): January 2021)

This implementation guide discusses architectural considerations and configuration steps for deploying the Media2Cloud solution in the Amazon Web Services (AWS) Cloud. It includes links to an AWS CloudFormation template that launches, configures, and runs the AWS services required to deploy this solution using AWS best practices for security and availability.

The guide is intended for IT infrastructure architects and developers who have practical experience working with video workflows and architecting in the AWS Cloud.
Overview

Amazon Web Services (AWS) offers highly scalable, elastic, and secure cloud services that help customers more easily ingest, store, process, deliver, and manage content in the cloud. Moving your digital asset management to the cloud enables you to take advantage of the latest innovations in asset management and supply chain applications. However, it can be a challenging and slow process to migrate your existing video archives to the cloud.

To help streamline and automate the migration process, AWS offers the Media2Cloud solution. This solution sets up a serverless end-to-end ingest workflow to move your video assets and associated metadata to the cloud. During the migration, the solution analyzes and extracts valuable metadata from your video and images using Amazon Rekognition, Amazon Transcribe, and Amazon Comprehend. Media2Cloud also includes a simple web interface that enables you to immediately start ingesting and analyzing your content to enrich its value.

Media2Cloud is designed to provide a serverless framework for accelerating the setup and configuration of a content ingest and analysis process. We recommend that you use this solution as a baseline and customize it to meet your specific needs.

Cost

You are responsible for the cost of the AWS services used while running the Media2Cloud solution. The total cost for running this solution depends on the amount of data being ingested and analyzed, running the solutions Amazon Elasticsearch Service cluster, and the size and length of media files analyzed with Amazon Rekognition, Amazon Transcribe, and Amazon Comprehend. For full details, see the pricing webpage for each AWS service you will be using in this solution. For customers who want to process large-scale video archives, we recommend that you contact your AWS account representative for at-scale pricing.

Architecture Overview

Deploying the Media2Cloud solution builds the following environment in the AWS Cloud.

![Figure 1: Media2Cloud architecture on AWS](image-url)
The AWS CloudFormation template deploys three workflows: one that ingests source videos and images, one that analyzes and extracts machine learning metadata from your content, and one that creates and manages labeling jobs. When you upload a video or image to the ingest Amazon Simple Storage Service (Amazon S3) bucket, the ingest workflow creates a standardized proxy file and thumbnails for analysis. The analysis workflow analyzes the videos and images and extracts metadata using AWS-native AI services. The labeling workflow uses Amazon SageMaker Ground Truth to create labeling jobs for your workforce that are used to tag faces to your face collection.

The ingest Amazon S3 bucket has an Amazon S3 lifecycle policy that allows the solution to move uploaded videos and images to Amazon Simple Storage Service Glacier (Amazon S3 Glacier) for archiving.

Ingest Workflow

The ingest workflow includes AWS Step Functions and AWS Lambda, which orchestrate the ingest workflow and start AWS Elemental MediaConvert to create standardized proxy files and thumbnails of the uploaded videos and images for analysis. If the media content is in Amazon S3 Glacier or S3 Glacier Deep Archive storage, the workflow automatically restores the media content from archive storage to Amazon S3 storage. Proxy files are created and stored in a proxy Amazon S3 bucket. Media information such as bitrate, formats, audio channels container format, the asset unique identifier, and MD5 checksum are processed via AWS Lambda and stored in Amazon DynamoDB. Technical metadata extracted from videos or images are indexed in an Amazon Elasticsearch cluster. When video processing is completed, Amazon Simple Notification Service (Amazon SNS) sends notifications to subscribed users who may use the notification to start other workflows. For example, third party partner solutions (p. 11) such as Media Asset Manager (MAM) and Archive System can subscribe to the Amazon SNS topic and then integrated the derived information into their workflows. When an ingest Amazon SNS notification is received, the automated system can import the files into its system. For more information, see Appendix G (p. 33).

Analysis Workflow

The analysis workflow includes AWS Step Functions and AWS Lambda which leverage Amazon Rekognition, Amazon Transcribe, and Amazon Comprehend to analyze and extract machine learning
metadata from the proxy files. A Lambda function transforms the metadata results into WebVTT subtitle tracks, chapter markers, key phrases, labels, sentiments, entities, and locations. The machine learning metadata results are stored in the proxy Amazon S3 bucket and indexed in an Amazon ES cluster. When the analysis is completed, Amazon SNS sends notifications to subscribed users. For more information, see Appendix G (p. 33). For more information on the state machines used in the analysis workflow, see Analysis Process (p. 7).

**Figure 3: Media2Cloud analysis workflow**

**Labeling Workflow**

The labeling workflow includes AWS Step Functions and AWS Lambda, which create and manage labeling jobs. The labeling job creates a dataset, including faces, for indexing. Annotation results are indexed to Amazon Rekognition face collection. Amazon SNS sends a notification to the responsible work team that will work on the labeling job. After the work team completes the job, the labeling state machine collects the annotations and indexes the annotation results to the Amazon Rekognition face collection. The unique face IDs are stored in an Amazon DynamoDB table.
Figure 4: Media2Cloud labeling workflow

Web Interface

The solution also deploys an Amazon Cognito user pool, an Amazon CloudFront distribution, a web Amazon S3 bucket, and a simple web interface. The web interface makes it easy to upload, browse, search video and image files, extract metadata, and create and manage your labeling workforce. The web interface leverages Amazon Cognito for user authentication and is powered by web assets hosted in the web Amazon S3 bucket. Amazon CloudFront is used to provide public access to the solution’s website bucket contents.

An Amazon API Gateway RESTful API is used for searching results stored in the Amazon ES cluster, and AWS IoT Core is used as a publish/subscribe message broker to periodically update workflow progress to connected web clients.
Figure 5: Media2Cloud web interface
Solution Components

Ingest Process

When a new video or image is uploaded to the ingest Amazon S3 bucket through the web interface, the ingest process starts. The workflow generates an asset unique identifier, computes and validates an MD5 checksum, and extracts media information such as bitrate, formats, audio channels container format for video, or EXIF information such as GPS location, model, and make for image. The workflow creates a proxy file and thumbnails using AWS Elemental MediaConvert. The proxy files and thumbnail images are stored in the proxy Amazon S3 bucket. The technical metadata are indexed to an Amazon Elasticsearch Service (Amazon ES) cluster.

When the workflow is completed, the source files are tagged to allow the Amazon S3 lifecycle policy to move files to Amazon S3 Glacier storage class for archiving.

Analysis Process

The Media2Cloud solution provides the following preset options for the analysis process when you deploy the template: Default, All, and Audio and Text.

- **Default** enables celebrity recognition, labels, transcription, key phrases, entities, and text processes.
- **All** enables all detections including celebrity recognition, labels, transcription, key phrases, entities, text, faces, face matches, person, moderation, sentiment, and topic processes.
- **Audio and Text** enables transcription, key phrases, entities, and text processes.

Four state machines are deployed to process the analysis.

- The video analysis state machine analyzes and extracts AI/ML metadata from the video proxy using Amazon Rekognition video APIs.
- The audio analysis state machine analyzes and extracts AI/ML metadata from the audio stream of the proxy file using Amazon Transcribe and Amazon Comprehend.
- The image analysis state machine analyzes and extracts image metadata with Amazon Rekognition image APIs.
- The analysis monitoring state machine monitors the video analysis, audio analysis, and image analysis state machines and periodically reports the analysis process status to the web interface by sending the status to an AWS IoT Core MQTT topic. The machine learning metadata results are stored in the proxy Amazon S3 bucket and indexed in an Amazon ES cluster.

Labeling Process

The labeling workflow manages the lifecycle of the labeling job from its creation to indexing the results. Using the web interface, you can crop faces from videos or images and either label them immediately or place all the cropped faces in a queue for batch processing.

If you select batch processing, use Amazon SageMaker Ground Truth private workforce to send a batch job to your labeling work team for processing. Your labeling team can be composed of staff within your
organization as well as external workers (such as contractors and interns). Your labeling team receives an email notification containing the access details for the labeling job. When the job is completed, the workflow collects and indexes the annotated results in the Amazon Rekognition face collection. The indexed faces are stored in an Amazon DynamoDB table.

Error Handling

The Media2Cloud solution applies a catch and retry concept for error handling to the state machines to improve the resiliency of the solution by retrying the state execution multiple times. When the state execution exhausts the retries, it stops the execution and generates an error.

The solution also uses Amazon CloudWatch Events to respond to execution errors caused by the state machines (ingest, analysis, and labeling). The error handling Lambda function processes the error by analyzing the execution history of the failed state machine and sends an Amazon Simple Notification Service (Amazon SNS) notification to subscribers.

Proxy Files

When a new video is uploaded to Amazon S3, the Media2Cloud solution automatically converts the video to .mp4 format and creates a compressed version of the video known as a proxy file. For this solution, proxy files are used to enable users to upload videos of various sizing and formatting, without being subject to Amazon Rekognition and Amazon Transcribe limits. Additionally, the proxy files can be used as reference proxies in a Media Asset Manager (MAM) for search, discovery, and proxy editing.

Web Interface

The Media2Cloud solution deploys a web interface that makes it easy to upload, browse, search video and image files, index faces to create your own face collection, and view artificial intelligence and machine learning information. This web interface can be used as a reference for building your own end-to-end ingest and analysis workflow applications. The interface automatically subscribes to the AWS IoT Core message broker to display the ingest, analysis, and labeling process status and progress. You can use the web interface to search results in the Amazon ES cluster and start workflows.

The web interface includes an HTML5 video player by VideoJS that can play the MP4 proxy video files, generated by the ingest workflow and displays Machine Learning (ML) metadata created by the analysis workflow by using Amazon S3 signed URLs.

Amazon DynamoDB

The solution deploys the following Amazon DynamoDB tables which are configured to use on-demand capacity and encryption at rest using SSE.

- A table to store ingest information
- A table to store machine learning metadata
- A table to store indexed faces
- A table to store queued faces that are ready to process by labeling workers
- A table to temporarily store analysis results used internally by the analysis workflow
Amazon SNS

This solution deploys two Amazon Simple Notification Service (Amazon SNS) topics: one used to receive ingest, analysis, labeling, and error notifications from the workflows and one used by the Amazon SageMaker Ground Truth private workforce to send notifications to the labeling workers.
Deployment Considerations

Limits

Currently, the Media2Cloud solution has the following known limitations:

- **Maximum media duration is limited to 4 hours**
  
  Amazon Transcribe can process files up to 4 hours in length. For more information, see Amazon Transcribe Guidelines and Limits.

- **Concurrent Amazon Rekognition video processes are limited to 20 processes**
  
  Amazon Rekognition supports up to 20 concurrent video processes. For more information, see Limits in Amazon Rekognition. By default, the Media2Cloud AWS CloudFormation stack enables two Amazon Rekognition video processes, one for celebrity recognition and one for label detections. Each video analysis consumes two processes. Therefore, you can run at most 10 concurrent analysis jobs with this solution.

  You can use the web interface's Settings page to enable or disable the Amazon Rekognition video processes by turning off individual detection.

![Figure 6: Amazon Rekognition default detection settings](image)

Custom Sizing

Choose from three preset Amazon Elasticsearch Service (Amazon ES) cluster sizes to support your anticipated metadata results:
Small:
- 3 dedicated primary nodes; t2.small.elasticsearch instance type
- 2 data nodes; m4.large.elasticsearch instance type

Medium:
- 3 dedicated primary nodes; t2.medium.elasticsearch instance type
- 4 data nodes; m4.large.elasticsearch instance type

Large:
- 3 dedicated primary nodes; t2.medium.elasticsearch instance type
- 6 data nodes; m4.large.elasticsearch instance type

Integrated Partners

The Media2Cloud solution is designed to provide a standardized architecture to support AWS Technology Partners to integrate with content from AWS customers. A standardized architecture helps accelerate the migration and supply chain process, and helps Media Asset Manager (MAM) partners provide solutions for their customers.

The Media2Cloud solution integrates with the following AWS Partners:

Cloudfirst.io

Cloudfirst.io is an AWS Partner that specializes in large-scale, unstructured, active archive, and content storage management solutions for Media and Entertainment. They actively assist clients with legacy archive migrations embracing various next-generation technologies. Cloudfirst provides consulting and a product called Rapid Migrate that address the challenges of moving content out of existing LTO archives, process and move content into Amazon Simple Storage Service (Amazon S3) storage in supported content and metadata formats for Media2Cloud to initiate the ingest process.

Levels Beyond

Levels Beyond is an AWS Partner that provides a Media Asset Manager (MAM) service platform called Reach Engine. Levels Beyond can be integrated with the Media2Cloud solution through Amazon Simple Notification Service (Amazon SNS) and interface with the output to consume the JSON formatted metadata to provide customers with a rich search, discovery and management service to manage their content archives. Levels Beyond can support customers further by configuring the services to add additional metadata faceting as well as automating the processing of content for production, OTT, digital publishing and other content related services.

Nomad CMS

Nomad CMS is an AWS Partner that supports the ability to bring an OTT metadata enrichment and discovery system to existing Amazon S3 assets. Nomad augments Amazon S3 asset storage without requiring any changes to the existing asset structure or files themselves. Nomad also automatically integrates with Media2Cloud and other AWS AI/ML services. Confidence scores, labels, transcriptions, and other AI enrichment is used to tag each asset with appropriate discovery information. Searching and publishing activities are used to make the resulting metadata available to custom solutions or in support of other integration activities.
EditShare

EditShare is an AWS Partner that designs and delivers high-performance, scalable, shared storage solutions that enable media professionals to create outstanding content. EditShare’s EFSv with FLOW is a Media2Cloud enabled, end-to-end cloud production solution. It supports tiered asset storage, media management, intelligent archiving, and broad compatibility with creative tools such as the Adobe Creative Suite. Highlighting just one use case, EFSv and FLOW powered workflows have fast search and seamless switching between proxy and high-resolution editing right in the video editorial application. EditShare’s Professional Services team can offer AWS customers seamless workflows designed around their business processes, leveraging solutions from EditShare and other providers.

eMAM

eMAM is an AWS Partner that powers workflows for production, post-production, sharing, and distribution: the entire lifecycle of a digital asset. eMAM provides a web interface designed to support non-technical users, providing a collaboration nexus for editors and designers using integrations into Apple Final Cut and Adobe Creative Cloud applications. eMAM is flexible, with easy configuration and scalability for the entire range of use cases and verticals, to provide customers with choice and control. eMAM provides a range of options for deployment including AWS cloud and hybrid solutions. eMAM is available as a permanent license or as a subscription in the AWS Marketplace with SaaS/PaaS-Server options.

Evertz

Evertz is an AWS Partner that provides the Mediator-X, a cohesive, highly scalable, infrastructure agnostic platform for Media Asset Management, Transmission Playout and Non Linear delivery applications. Evertz Mediator-X allows customers to manage their Cloud Content Factory at scale using a rich feature set of integrations and options under the functional blocks of acquisition, processing, management, production, playout, and delivery. Utilizing Media2Cloud and other AWS services, customers can gather and store both metadata and content in highly durable cloud storage, use the intuitive user-interface to visualize machine learning data alongside other customer-specific metadata or pull data from API endpoints within the Mediator-X platform.

IMT

IMT is an AWS Partner that provides SoDA, a new way to control data movement between storage tiers, on-site and in the cloud. Since its inception 13 years ago, IMT has grown to become a leading next-gen Systems Integrator supporting over 800+ customers in Media & Entertainment, broadcast, sports, and corporate video in North America. SoDA is IMT’s Intelligent Data Management Software that can be leveraged as a simple data migration tool to help customers move off from legacy archives, as well as broker data movement to and from the cloud to various endpoints. Designed to work with all types of storage—on-premises, hybrid, and AWS—users can define rich, flexible policies or manually transfer data. SoDA plugs into multiple MAM solutions to empower end users to control their own data movement.

Starchive

Starchive is an AWS Partner that offers a command center for today’s content producers. Starchive brings the power of digital asset management to the entrepreneur and small/medium business with the elegance of a modern consumer SaaS application and at a fraction of the cost of comparable solutions. Starchive helps users find the signal in the noise of their digital chaos and get back to work building their brand, business, and bottom line. In a world where every individual has the power to create and the opportunity to consume digital media 24/7—every business has the mandate to be a content powerhouse to thrive. Learn more about how Starchive used Media2Cloud to help Essence Magazine support their 50-year anniversary by improving the accessibility to their historical archive.
TrackIt

TrackIt is an AWS Advanced Consulting Partner with decades of experience in the Media & Entertainment industry and a wealth of cloud technology design and deployment work performed for many media-centric companies. TrackIt has experience building advanced pipelines that include AI/ML tools and integration with asset management systems, along with transcoding, rendering, VOD, OTT, live streaming, cloud-based editorial, and collaborative online tools. Learn more about how TrackIt used Media2Cloud to help Jukin Media improve the utility of their archive.

Regional Deployment

This solution uses the Amazon Rekognition, Amazon Comprehend, and Amazon Transcribe services which are currently available in specific AWS Regions only. Therefore, you must launch this solution in an AWS Region where these services are available. For the most current service availability by region, refer to the AWS Regional Services List.
AWS CloudFormation Template

This solution uses AWS CloudFormation to automate the deployment of the Media2Cloud solution on the AWS Cloud. It includes the following AWS CloudFormation template, which you can download before deployment:

**media2cloud-deploy.template**: Use this template to launch the solution and all associated components. The default configuration deploys Amazon Simple Storage Service (Amazon S3) buckets, an Amazon Elasticsearch cluster, AWS Lambda functions, AWS Step Functions state machines, an Amazon API Gateway RESTful API, an Amazon CloudFront distribution, Amazon Cognito user pools, Amazon Simple Notification Service (Amazon SNS) topics, AWS IoT Core, Amazon DynamoDB tables, and an Amazon SageMaker Ground Truth private workforce. You can also customize the template based on your specific needs.
Automated Deployment

Before you launch the automated deployment, please review the architecture, configuration, and other considerations discussed in this guide. Follow the step-by-step instructions in this section to configure and deploy the Media2Cloud solution into your account.

Time to deploy: Approximately 25 minutes

What We'll Cover

The procedure for deploying this architecture on AWS consists of the following steps. For detailed instructions, follow the links for each step.

Step 1. Launch the Stack (p. 15)
- Launch the AWS CloudFormation template into your AWS account.
- Enter values for required parameters: Stack Name and Email Address.
- Review the other template parameters, and adjust if necessary.

Step 2. Upload a Video or Image File (p. 17)
- Upload a file using the web interface to begin the ingest workflow.

Step 3. Create and View Metadata (p. 18)
- Create the metadata from the uploaded file.

Step 4. Configure the Labeling Work Team (p. 18)
- Configure your labeling work team.

Step 5. Create Your Face Collection (p. 18)
- Index faces to create grow your face collection to improve face analysis results.

Step 1. Launch the Stack

This automated AWS CloudFormation template deploys the Media2Cloud solution in the AWS Cloud.

Note
You are responsible for the cost of the AWS services used while running this solution. See the Cost (p. 2) section for more details. For full details, see the pricing webpage for each AWS service you will be using in this solution.

1. Sign in to the AWS Management Console and click the button below to launch the media2cloud AWS CloudFormation template.
You can also download the template as a starting point for your own implementation.

2. The template is launched in the US East (N. Virginia) Region by default. To launch this solution in a different AWS Region, use the region selector in the console navigation bar.

   **Note**
   This solution uses the Amazon Rekognition, Amazon Comprehend, and Amazon Transcribe services which are currently available in specific AWS Regions only. Therefore, you must launch this solution in an AWS Region where these services are available. For the most current service availability by Region, refer to the [AWS Regional Services List](#).

3. On the **Create stack** page, verify that the correct template URL shows in the **Amazon S3 URL** text box and choose **Next**.

4. On the **Specify stack details** page, assign a name to your solution stack.

5. Under **Parameters**, review the parameters for the template, and modify them as necessary.

   This solution uses the following default parameters.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Email</td>
<td>&lt;Requires Input&gt;</td>
<td>Email address of the user that will be created in the Amazon Cognito identity pool and subscribed to the Amazon Simple Notification Service (Amazon SNS) topic. Subscribed users will receive ingest, analysis, labeling, and error notifications. After launch, two emails will be sent to this address: one with instructions for logging in to the web interface and one confirming the Amazon SNS subscription.</td>
</tr>
<tr>
<td>Price Class</td>
<td>Use Only U.S., Canada and Europe</td>
<td>A dropdown box with price class for the edge location from which Amazon CloudFront serves your requests. Choose Use Only U.S., Canada and Europe; Use U.S., Canada, Europe, Asia and Africa; or Use All Edge Locations. For more information, see Choosing the Price Class.</td>
</tr>
</tbody>
</table>
### Table of Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Elasticsearch Cluster Size</td>
<td>Small</td>
<td>A drop-down box with three Amazon Elasticsearch Service (Amazon ES) cluster sizes: Small, Medium, Large.</td>
</tr>
<tr>
<td>Default Language Code</td>
<td>en-US</td>
<td>The default language code used by Amazon Transcribe and Amazon Comprehend to process speech to text and Neutral Language Processing (NLP) analysis</td>
</tr>
<tr>
<td>Analysis Feature(s)</td>
<td>Default</td>
<td>A drop-down box with three presets: Default, All, and Audio and Text. For more information about the presets, see Analysis Process (p. 7).</td>
</tr>
<tr>
<td>Anonymous Usage</td>
<td>Yes</td>
<td>Send anonymous usage data to AWS to help understand solution usage across our customer base as a whole. To opt out to this feature, select No.</td>
</tr>
</tbody>
</table>

6. Choose **Next**.
7. On the **Configure stack options** page, choose **Next**.
8. On the **Review** page, review and confirm the settings. Be sure to check the box acknowledging that the template will create AWS Identity and Access Management (IAM) resources.
9. Choose **Create stack** to deploy the stack.

You can view the status of the stack in the AWS CloudFormation console in the **Status** column. You should see a status of **CREATE_COMPLETE** in approximately 25 minutes.

---

### Step 2. Upload a Video or Image File

After the solution successfully launches, you can start uploading video or image files for processing. The solution sends two emails: one with the subscription confirmation for the Amazon SNS topic to send ingest, analysis, labeling, and error notifications, and one with instructions for signing into the solution’s provided web interface.

1. In the **M2CStatus** email, select **Confirm subscription** to subscribe to the Amazon SNS topic.
2. In the second email, follow the instructions to sign in to the website.

   You will be prompted to change the password the first time you sign in.
3. Choose **Sign in** on the upper right corner of the page and sign in using your recently created password.
4. Navigate to the **Demo** tab.
5. Choose the plus (+) sign, upload a video or image file, and choose Quick upload. On the next screen, choose Start Upload.

Once the ingest process is completed, a thumbnail image of the video or image is created. You can hover on the thumbnail image and select Play now to view the media file.

Step 3. Create and View Metadata

1. Hover over the created video or image proxy, right-click to access the menu options, and select Create metadata.

Once the process is completed, you can view the metadata.

2. Choose Play now.

3. In the Play window, scroll to the carousel to view the archive and media information (or EXIF information for images), transcription, Amazon Rekognition, and Amazon Comprehend results.

4. To view the metadata results for each result, select each individually.

Step 4. Configure the Labeling Work Team

The solution integrates with Amazon SageMaker Ground Truth private workforce. You can use the web interface to configure and manage a work team. You can invite or remove workers from your work team.

Use the following procedure to add a member to the work team.

1. In the solution’s web interface, navigate to the Settings page. Under Ground Truth Workforce settings, choose Run wizard.

2. In the wizard dialog box, choose Start.

3. In the Choose a work team window, select the solution’s default work team and choose Next.

4. Add or remove members.

   • To add a member, in the Manage team member window, enter the member’s email address and choose Add member.

     The labeling team member receives an invitation email with instructions to sign in to the labeling portal.

   • To remove a member, in the Manage team member window, enter the member’s email address and choose Remove member.

Step 5. Create Your Face Collection

The web interface allows you to create your own Amazon Rekognition face collection and index and store faces in the collection to improve the analysis results.

1. In the web interface, hover over a created video or image and choose Play now.

2. Scroll through the carousel until the option to create a Snapshot appears.

3. To crop a face on the video or image, choose Snapshot.

4. To index the face, choose Index now or Queue for later.

   • To index the face immediately, choose Index now. In the who is this person? field, enter the first and last name of the cropped face and choose Index now. The cropped face is indexed with the name you entered in your Amazon Rekognition face collection.
• To create a list of faces to be sent to Amazon SageMaker Ground Truth for labeling by your labeling work team, choose **Queue for later**. This option temporarily stores the face in a DynamoDB table, enabling you to accumulate faces from the same video, from other videos, and from images. When you are ready to label the faces, choose **Send to Ground Truth** to send all the accumulated faces to the labeling workflow. The members of your work team will receive notification containing the access details to perform the labeling job.

5. After the faces are indexed, choose **Re-analyze** to analyze the video or image using the newly indexed faces in your face collection so that all unidentified faces are recognized and indexed.
Security

When you build systems on AWS infrastructure, security responsibilities are shared between you and AWS. This shared model can reduce your operational burden as AWS operates, manages, and controls the components from the host operating system and virtualization layer down to the physical security of the facilities in which the services operate. For more information about security on AWS, visit the AWS Security Center.

Server-Side Encryption

AWS highly recommends that customers encrypt sensitive data in transit and at rest. This solution automatically encrypts media files and metadata at rest with Amazon Simple Storage Service (Amazon S3) Server-Side Encryption (SSE). The solution's Amazon Simple Notification Service (Amazon SNS) topics and Amazon DynamoDB tables are also encrypted at rest using SSE.

Amazon CloudFront

This solution deploys a static website hosted in an Amazon S3 bucket. To help reduce latency and improve security, this solution includes an Amazon CloudFront distribution with an origin access identity, which is a special CloudFront user that helps restrict access to the solution’s website bucket contents. For more information, see Restricting Access to Amazon S3 Content by Using an Origin Access Identity.
Additional Resources

AWS services

- AWS CloudFormation
- AWS Lambda
- Amazon Rekognition
- Amazon Cognito
- Amazon Simple Storage Service
- Amazon Transcribe
- Amazon Simple Notification Service
- Amazon Comprehend
- Amazon API Gateway
- Amazon Elasticsearch Service
- AWS Step Functions
- Amazon CloudFront
- Amazon DynamoDB
- AWS IoT Core
- AWS Elemental MediaConvert
- Amazon SageMaker Ground Truth
Appendix A: File Paths in Amazon S3

The Media2Cloud solution creates three different Amazon Simple Storage Service (Amazon S3) buckets to store the assets that are created:

- A web bucket that stores the static HTML, CSS, and JavaScript files for the web interface
- An ingest bucket that stores your original source files
- A proxy bucket that stores all the files and assets generated by the solution including:
  - Video proxies and thumbnail images generated by AWS Elemental MediaConvert
  - Mediainfo XML output generated by mediainfo and/or EXIF
  - JSON documents generated by exiftool
  - Machine learning metadata generated by AWS Machines Learning services
  - Additional WebVTT tracks and analysis JSON documents created by the solution

The following table shows the file types and Amazon S3 file paths.

<table>
<thead>
<tr>
<th>File Type</th>
<th>File path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Web static assets</td>
<td>S3://&lt;web-bucket&gt;/</td>
</tr>
<tr>
<td>Uploaded file</td>
<td>S3://&lt;ingest-bucket&gt;/&lt;file-basename&gt;/&lt;filename&gt;</td>
</tr>
<tr>
<td>Technical metadata such as mediainfo and EXIF results</td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/mediainfo</td>
</tr>
<tr>
<td></td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/imageinfo</td>
</tr>
<tr>
<td>Proxy files, thumbnails generated by MediaConvert</td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/transcoded</td>
</tr>
<tr>
<td>All AI/ML analysis results</td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis</td>
</tr>
<tr>
<td>Raw AI/ML analysis results from Amazon AI services</td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/raw/&lt;date-time&gt;/comprehend/</td>
</tr>
<tr>
<td></td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/raw/&lt;date-time&gt;/rekognition/</td>
</tr>
<tr>
<td></td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/raw/&lt;date-time&gt;/transcribe/</td>
</tr>
<tr>
<td>WebVtt tracks generated by analysis state machine</td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/vtt/celeb/&lt;name&gt;.vtt</td>
</tr>
<tr>
<td></td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/vtt/label/&lt;name&gt;.vtt</td>
</tr>
<tr>
<td></td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/vtt/transcribe/&lt;name&gt;.vtt</td>
</tr>
<tr>
<td>File Type</td>
<td>File path</td>
</tr>
<tr>
<td>-----------------------------------------------</td>
<td>---------------------------------------------------------------------------</td>
</tr>
<tr>
<td>Metadata JSON documents generated by analysis state machine</td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/vtt/celeb/&lt;name&gt;.vtt</td>
</tr>
<tr>
<td></td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/vtt/entity/&lt;name&gt;.vtt</td>
</tr>
<tr>
<td></td>
<td>S3://&lt;proxy-bucket&gt;/&lt;uuid&gt;/&lt;filename&gt;/analysis/vtt/keyphrase/&lt;name&gt;.vtt</td>
</tr>
</tbody>
</table>
Appendix B: Ingest State Machine

The Media2Cloud solution ingests videos and images to extract media information and generate proxies using AWS Step Functions state machines and an AWS Lambda function. When a new video or image file is uploaded through the web interface, the solution sends an HTTP request to the Amazon API Gateway RESTful API endpoint to start the ingest process. A Lambda function invokes the ingest state machine. The state machine progress and status are sent to an AWS IoT topic that enables the web interface to refresh the results.

The ingest state machine is composed of the following processes.

• **Create record** - Creates a record of the uploaded file to the ingest Amazon DynamoDB table.
• **Restore object** - Checks the storage class of the uploaded file using the S3.HeadObject API. If the file is in either the GLACIER or DEEP_ARCHIVE storage class, the Lambda function starts the restore process using the S3.RestoreObject API.
• **Compute checksum** - Incrementally computes the MD5 checksum of a 20 GB chunk using the S3.GetObject byte range.

- **Run mediaminfo** - Runs the MediaInfo tool to extract technical metadata from the video or audio file. The raw MediaInfo XML result is stored in the proxy bucket.
• **Start transcode** - Creates a job template based on the media information extracted by MediaInfo. If the video file contains multiple audio tracks (an MXF file can contain eight to 16 audio tracks), the Lambda function selects the best combination of audio tracks, and runs AWS Elemental MediaConvert to create the proxy files and thumbnails. The proxy files and thumbnail images are stored in a proxy S3 bucket.

• **Check transcode status** - Checks the transcode status by using the `MediaConvert.GetJob` API.

• **Run imageinfo** - Runs `exiftool` to extract EXIF information from the image file, generates an image proxy file, and stores the proxies to proxy bucket.

![AWS Step Functions ingest workflow](image)

**Figure 9: Ingest workflow**

• **Update record** - Collects all results from the states such as locations of the proxies, thumbnail images, and either MediaInfo (for videos) or embedded technical metadata within videos or images and updates the results to the `ingest` DynamoDB table.

• **Index ingest results** - Indexes the technical metadata to Amazon Elasticsearch Service cluster.
Appendix C: Analysis State Machine

The Media2Cloud solution includes an analysis state machine that is composed of three different AWS Step Functions sub-state machines and a set of AWS Lambda functions that start, monitor, and collect results from the sub-state machines. The analysis state machine consists of the following sub-state machines:

- A video analysis sub-state machine that manages the video-based analysis process
- An audio analysis sub-state machine that manages the audio-based analysis process
- An image analysis sub-state machine that manages the image-based analysis process

When the ingest process is completed, the web interface displays a Create metadata button to start an analysis workflow. When initiated, the web interface sends an analysis request to Amazon API Gateway where a Lambda function validates the request and starts the analysis state machine. Similar to the ingest state machine, the analysis state machine publishes progress and status to an AWS IoT topic. The web interface processes the progress status and sends updates to the web interface.

![AWS Step Functions analysis monitor workflow](image)

**Figure 10: Analysis workflow**

- **Start analysis** - Parses the input request and starts the video and audio analysis machines for a video file and/or starts the image analysis machine for an image file by calling the `StepFunctions StartExecution` API.
- **Check analysis status** - Periodically checks each of the sub-state machine's status by calling the `StepFunctions DescribeExecution` API.
- **Collect analysis results** - Collects outputs from each sub-state machine by calling the `StepFunctions DescribeExecution` API, parses, and joins the results.
- **Index analysis results** - Indexes analysis metadata collected from each sub-state machine to an Amazon Elasticsearch engine.

### Video Analysis Sub-state Machine

The video analysis sub-state machine is managed by the analysis state machine. It runs a series of Amazon Rekognition async processes to extract faces, celebrities, labels, moderation, and face match data from the video file. This sub-state machine consists of a number of parallel branches where each branch runs and monitors a specific Amazon Rekognition async process. For example, `StartCelebrityRecognition` detects celebrities from the video.
The following video analysis workflow shows the celebrity detection process.

- **Start celeb detection** - Calls the Amazon Rekognition StartCelebrityRecognition API to start the async process.
- **Check celeb detection status** - Periodically calls the Amazon Rekognition GetCelebrityRecognition API to check the status.
- **Collect celeb results** - Downloads celebrity results using the GetCelebrityRecognition API and stores raw results to an Amazon Simple Storage Service (Amazon S3) bucket in the following filepath: `<uuid>/<filename>/analysis/raw/<datetime>/rekog/celeb/`. This Lambda function also temporarily stores a list of celebrity names in an Amazon DynamoDB table, analysis-queue-table, for further processing.
- **Create celeb tracks** - Fetches the list of celebrity names from the analysis-queue-table, downloads the metadata from the Amazon S3 bucket using the Amazon S3 SelectObjectContent API, converts timecode based celebrity metadata into a WebVTT track, and updates the analysis-queue-table to remove the processed celebrity name. The processed WebVTT track and corresponding metadata are uploaded to the Amazon S3 bucket in `<uuid>/<filename>/analysis/vtt/rekog/celeb/` and `<uuid>/<filename>/analysis/metadata/rekog/celeb/`.

**Audio Analysis Sub-state Machine**

The audio analysis sub-state machine is managed by the analysis state machine. This sub-state machine runs Amazon Transcribe and Amazon Comprehend to extract transcription, entities, key phrases, sentiments, topic, and classification metadata. This sub-state machine first runs Amazon Transcribe to convert speech to text and starts a number of branches in parallel where each branch runs and monitors a specific Amazon Comprehend process.
The following audio analysis workflow shows the Lambda functions used to run the audio processes.

- **Start transcribe process** - Calls the Amazon Transcribe StartTranscriptionJob async API to start the speech to text process and transitions the transcribe status state to *Wait*.

- **Check transcribe process status** - Calls the Amazon Transcribe GetTranscriptionJob async API to check transcription job status.

- **Download transcription** - Downloads transcription results from the Amazon Transcribe service and stores the raw results to the Amazon S3 bucket in `<uuid>/<filename>/analysis/raw/<datetime>/transcribe/`. The state machine then starts the parallel branch.

- **Create subtitle state** - Converts the timestamp-based transcription into a WebVTT subtitle track. The WebVTT track is uploaded to the Amazon S3 bucket in `<uuid>/<filename>/analysis/vtt/transcribe/`. 
**Media2Cloud Implementation Guide**

**Image Analysis Sub-state Machine**

![AWS Step Functions image analysis workflow](image)

**Figure 17: Image analysis workflow**

- **Start image analysis** - Runs the Amazon Rekognition RecognizeCelebrities, DetectFaces, SearchFacesByImage, DetectLabels, DetectModerationLabels, and DetectText APIs in parallel, and collects and stores the metadata to the Amazon S3 bucket in `<uuid>/<filename>/analysis/raw/<datetime>/rekog-image/<type>/`.

**Image Analysis Sub-state Machine**

The image analysis sub-state machine is managed by the analysis state machine. It runs a series of Amazon Rekognition image (synchronized) processes to extract faces, celebrities, labels, moderation, face match, and texts from the video or image file.

**Figure 16: Audio analysis workflow to comprehend entity detection**

- **Start entity detection** - Downloads the transcription from the Amazon S3 bucket, checks to ensure the transcription contains enough data to run natural language processing (NLP). If there is not enough data, the `.status` flag is set to **NO_DATA**. If there is enough data, the Lambda function calls the Amazon Comprehend BatchDetectEntities API and stores the raw metadata to the Amazon S3 bucket in `<uuid>/<filename>/analysis/raw/<datetime>/comprehend/entity/`.

- **Create entity tracks** - Converts word offset entity metadata results to timestamp-based metadata results. The result is stored to the Amazon S3 bucket in `<uuid>/<filename>/analysis/vtt/comprehend/entity/`.

---

- ![AWS Step Functions audio analysis workflow: comprehend entity detection](image)

**Figure 16: Audio analysis workflow to comprehend entity detection**

- **Start entity detection** - Downloads the transcription from the Amazon S3 bucket, checks to ensure the transcription contains enough data to run natural language processing (NLP). If there is not enough data, the `.status` flag is set to **NO_DATA**. If there is enough data, the Lambda function calls the Amazon Comprehend BatchDetectEntities API and stores the raw metadata to the Amazon S3 bucket in `<uuid>/<filename>/analysis/raw/<datetime>/comprehend/entity/`.

- **Create entity tracks** - Converts word offset entity metadata results to timestamp-based metadata results. The result is stored to the Amazon S3 bucket in `<uuid>/<filename>/analysis/vtt/comprehend/entity/`.
Appendix D: Labeling State Machine

Use the labeling state machine to index existing face images in your collection to improve the confidence level for face matching detection. To begin a labeling job, do the following:

1. Play a video or an image.
2. Choose the **Snapshot** button.
3. Crop the face to be labeled.
4. Choose **Send to Ground Truth**.

Your private work team receives a notification from Amazon SageMaker when the labeling job is ready. The worker signs in to the labeling portal using Amazon Cognito and performs the labeling job.

**Note**
The Amazon Cognito user pool for the labeling state machine is separate from the Media2Cloud Amazon Cognito user pool. Your work team members are only given access to the labeling portal to perform the labeling jobs. They do not have access to any of your AWS resources.

The labeling state machine manages the face tagging workflow where it first creates a dataset and labeling job, waits for the workers to perform the job, collects the annotations, and indexes the results in the Amazon Rekognition face collection.

![AWS Step Functions labeling workflow](image)

**Figure 18: Labeling workflow**

- **Create dataset** - Retrieves queued faces from the Amazon DynamoDB queued-faces table. If there is data, it creates a dataset manifest. If there is no data, the AWS Lambda function switches to the next state.
- **Create labeling job** - Creates a labeling job by calling the Amazon SageMaker CreateLabelingJob API, updates the DynamoDB queued-face table, and transitions the labeling state to **Wait**.
- **Check labeling job status** - Checks the labeling job status by calling the Amazon SageMaker DescribeLabelingJob API.
- **Index results state** - Downloads and parses annotations, indexes faces in your face collection by calling the Amazon Rekognition IndexFaces API, and moves queued faces from the DynamoDB queued-face table to the indexed-face table.
Appendix E: State Machine Error Handling

All state machines have a built-in retry logic using the AWS Step Functions **Catch and Retry** to improve the resiliency of the workflow.

```
"Retry": [
  {
    "ErrorEquals": [ "States.ALL" ],
    "IntervalSeconds": 1,
    "MaxAttempts": 4,
    "BackoffRate": 1.2
  },
  "Catch": [
    {
      "ErrorEquals": [ "States.ALL" ],
      "Next": "Labeling error"
    }
  ]
]
```

If all retries fail, the state machine generates an error to stop the execution. Amazon CloudWatch Events is configured to monitor for errors generated by the state machines. The CloudWatch Events pattern is defined as follows:

```
{
  "detail-type": [ "Step Functions Execution Status Change"
  ],
  "source": [ "aws.states" ],
  "detail": {
    "stateMachineArn": [
    ],
    "status": [ "FAILED",
      "ABORTED",
      "TIMED_OUT"
    ]
  }
}
```

An AWS Lambda function then uses the **StepFunctions GetExecutionHistory** API to parse the last error of the state machine and publishes the error message to an Amazon Simple Notification Service (Amazon SNS) topic.
Appendix F: Lifecycle Policy

When the Media2Cloud AWS CloudFormation template is deployed, the template updates the lifecycle policy of the Amazon S3 Glacier vault with a transition rule. The rule is set to transfer files that contain an object tag `IngestCompleted` and value tag `IngestedCompleted`, to the Amazon S3 Glacier vault. The tag is set during the `Update record` state using the `S3.putObjectTagging()` API. Note that the file transfer will occur at midnight UTC time.

![Lifecycle rule](image)

**Figure 19: Amazon S3 Glacier lifecycle policy**
Appendix G: Amazon SNS Notifications

The Media2Cloud solution sends Amazon Simple Notification Service (Amazon SNS) a publish notification to the subscriber when the ingest, analysis, and labeling state machines are completed, or when there is an error. You can automate other workflows such as importing data to a Media Asset Management (MAM) system by using the Amazon SNS topic subscription created by the solution. The following JSON notification messages can be customized to your needs.

Ingest State Machine Notification Message

```json
{
  "operation": "job-completed",
  "input": {
    "src": {
      "bucket": "<ingest-bucket>",
      "key": "demo-02/demo-02.mp4",
      "type": "video"
    },
    "restore": {
      "startTime": 1572428433387,
      "endTime": 1572428433387
    },
    "checksum": {
      "algorithm": "md5",
      "fileSize": 5885585,
      "computed": "f0c34a09ab84db214e4c7bd6a164e5e9",
      "storeChecksumOnTagging": true,
      "startTime": 1572428434655,
      "endTime": 1572428434969,
      "comparedWith": "object-metadata",
      "comparedResult": "MATCHED",
      "tagUpdated": true
    },
    "mediainfo": {
      "output": "<uuid>/demo-02/mediainfo/output.xml"
    },
    "transcode": {
      "jobId": "1572428437358-yjfe1o",
      "destination": "<uuid>/demo-02/transcoded/"
    },
    "indexer": {
      "terms": [
        "lastModified",
        "timestamp",
        "basename",
        "fileSize",
        "mime",
        "mediainfo",
        "uuid",
        "key",
        "type",
        "md5"
      ]
    }
  }
}
```
Ingest State Machine Notification Message

```
{
  "operation": "job-completed",
  "status": "COMPLETED",
  "stateMachine": "SO0050-m2csolution-ingest",
  "uuid": "<uuid>",
  "progress": 0,
  "input": {
    "src": {
      "bucket": "Ingest bucket where the uploaded file is stored",
      "key": "Object key of the uploaded file",
      "type": "Major type of the uploaded file: image or video"
    },
    "restore": {
      "startTime": "If object is in GLACIER or DEEP_ARCHIVE storage, startTime indicates the start time of the restore process",
      "endTime": "If object is in GLACIER or DEEP_ARCHIVE storage, endTime indicates the end time of the restore process"
    },
    "checksum": {
      "algorithm": "Checksum algorithm type: md5 or sha1",
      "fileSize": "File size of the uploaded file",
      "computed": "The computed MD5 or SHA1 checksum",
      "storeOnTagging": "A flag indicates if the checksum value is stored in Amazon S3 object tagging",
      "startTime": "Start time of the checksum process",
      "endTime": "End time of the checksum process",
      "comparedWith": "indicates how we compare the "computed" checksum

- **object-metadata** refers to x-amz-metadat-md5 is used for comparison
- **object-tagging** refers to an existing "computed-md5" object tag is used for comparison
- **object-etag** refers to the object ETag value is used for comparison"
    }
  }
}
```

<table>
<thead>
<tr>
<th>Key Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operation</td>
<td>Last state of the state machine: job-completed</td>
</tr>
<tr>
<td>status</td>
<td>Status of the state machine: COMPLETED</td>
</tr>
<tr>
<td>stateMachine</td>
<td>Ingest state machine name</td>
</tr>
<tr>
<td>uuid</td>
<td>UUID of the file</td>
</tr>
<tr>
<td>progress</td>
<td>Progress of the state machine</td>
</tr>
<tr>
<td>input</td>
<td>Input parameters</td>
</tr>
<tr>
<td>input.src.bucket</td>
<td>Ingest bucket where the uploaded file is stored</td>
</tr>
<tr>
<td>input.src.key</td>
<td>Object key of the uploaded file</td>
</tr>
<tr>
<td>input.src.type</td>
<td>Major type of the uploaded file: image or video</td>
</tr>
<tr>
<td>input.restore.startTime</td>
<td>If object is in GLACIER or DEEP_ARCHIVE storage, startTime indicates the start time of the restore process</td>
</tr>
<tr>
<td>input.restore.endTime</td>
<td>If object is in GLACIER or DEEP_ARCHIVE storage, endTime indicates the end time of the restore process</td>
</tr>
<tr>
<td>input.checksum.algorithm</td>
<td>Checksum algorithm type: md5 or sha1</td>
</tr>
<tr>
<td>input.checksum.fileSize</td>
<td>File size of the uploaded file</td>
</tr>
<tr>
<td>input.checksumcomputed</td>
<td>The computed MD5 or SHA1 checksum</td>
</tr>
<tr>
<td>input.checksum.storeOnTagging</td>
<td>A flag indicates if the checksum value is stored in Amazon S3 object tagging</td>
</tr>
<tr>
<td>input.checksum.startTime</td>
<td>Start time of the checksum process</td>
</tr>
<tr>
<td>input.checksum.endTime</td>
<td>End time of the checksum process</td>
</tr>
</tbody>
</table>
| input.checksum.comparedWith | indicates how we compare the "computed" checksum

- **object-metadata** refers to x-amz-metadat-md5 is used for comparison
- **object-tagging** refers to an existing "computed-md5" object tag is used for comparison
- **object-etag** refers to the object ETag value is used for comparison
### Analysis State Machine Notification Message

If all retries fail, the state machine generates an error to stop the execution. Amazon CloudWatch Events is configured to monitor for errors generated by the state machines. The CloudWatch Events pattern is defined as follows:

```
{
  "operation": "job-completed",
  "progress": 0,
  "uuid": "<uuid>",
  "status": "COMPLETED",
  "stateMachine": "SO0050-m2csolution-analysis",
  "input": {
    "aiOptions": {
      "celeb": true,
      "face": true,
      "faceMatch": true,
      "label": true,
      "moderation": true,
      "person": true,
      "text": true,
      "transcript": true,
      "entity": true,
      "keyphrase": true,
      "sentiment": true,
      "topic": true,
      "document": false,
      "languageCode": "en-US",
      "customVocabulary": "your-custom-vocabulary-en-US",
      "faceCollectionId": "your-rekognition-face-collection",
      "minConfidence": 80,
      "classification": false
    },
    "video": {
      "arn": "02b386b2-cdc0-4a36-a01c-4ec55a899e93",
      "status": "COMPLETED",
      "startTime": 1572428774779,
      "endTime": 1572428933323,
      "rekognition": {
        "celeb": {
          "id": "4d433218929e11fbf89aaf64f47aa6b0606e69c739158af3387027d6926329b6",
```
"startTime": 1572428777012,
"endTime": 1572428899621,
"output": "<uuid>/demo-02/analysis/raw/20191030T094613/rekognition/celeb/output.json",
"vtt": "<uuid>/demo-02/analysis/vtt/celeb",
"metadata": "<uuid>/demo-02/analysis/metadata/celeb"
},
"face": {
  "id": "7687e0a9edacfd7d5158a49c9f9f19c2f1238d2d4b5f7b50e0a4c9b89df36be",
  "startTime": 1572428778735,
  "endTime": 1572428850747,
  "output": "<uuid>/demo-02/analysis/raw/20191030T094613/rekognition/face/output.json"
},
"faceMatch": {
  "id": "f094dee814a3e5f7614bb194eb7405803e0f0d8a10e56e72d001fc3d93b2ffccf",
  "startTime": 1572428877721,
  "endTime": 1572428875221,
  "output": "<uuid>/demo-02/analysis/raw/20191030T094613/rekognition/faceMatch/output.json",
  "vtt": "<uuid>/demo-02/analysis/vtt/faceMatch",
  "metadata": "<uuid>/demo-02/analysis/metadata/faceMatch"
},
"label": {
  "id": "0559be024bcf05e65c5c219126af609b6bb614ef06881530b677734a1e4e3a7",
  "startTime": 15724287777002,
  "endTime": 1572428875221,
  "output": "<uuid>/demo-02/analysis/raw/20191030T094613/rekognition/label/output.json",
  "vtt": "<uuid>/demo-02/analysis/vtt/label",
  "metadata": "<uuid>/demo-02/analysis/metadata/label"
},
"moderation": {
  "id": "11f9cf6b520168b47daab9d715af3dd8254fb37b285acce0e29631ccaf6",
  "startTime": 15724287777110,
  "endTime": 1572428850940,
  "output": "<uuid>/demo-02/analysis/raw/20191030T094613/rekognition/moderation/output.json",
  "vtt": "<uuid>/demo-02/analysis/vtt/moderation",
  "metadata": "<uuid>/demo-02/analysis/metadata/moderation"
},
"person": {
  "id": "60e0be718f730798fd58c101122ace3632a53d1994db5b8f528a994825d4f08",
  "startTime": 1572428777997,
  "endTime": 1572428875010,
  "output": "<uuid>/demo-02/analysis/raw/20191030T094613/rekognition/person/output.json",
  "vtt": "<uuid>/demo-02/analysis/vtt/person",
  "metadata": "<uuid>/demo-02/analysis/metadata/person"
}
]
},
"audio": {
  "arn": "1c30597e-623c-4474-b822-f3ce149e1226",
  "status": "COMPLETED",
  "startTime": 1572428774757,
  "endTime": 1572428829213,
  "comprehend": {
    "entity": {
      "startTime": 1572431779336,
      "endTime": 1572431779937,
      "output": "<uuid>/demo-02/analysis/raw/20191030T013413/comprehend/entity/output.json",
      "metadata": "<uuid>/demo-02/analysis/metadata/entity/output.json"
    },
    "keyphrase": {
    ...
  }
  },
  "transcribe": {
    "startTime": 1572428777012,
    "endTime": 1572428899621,
    "output": "<uuid>/demo-02/analysis/raw/20191030T094613/transcribe/output.json",
    "vtt": "<uuid>/demo-02/analysis/vtt/transcribe",
    "metadata": "<uuid>/demo-02/analysis/metadata/transcribe" 
  },
  "transcribeAsr": {
    "startTime": 1572428777012,
    "endTime": 1572428899621,
    "output": "<uuid>/demo-02/analysis/raw/20191030T094613/transcribeAsr/output.json",
    "vtt": "<uuid>/demo-02/analysis/vtt/transcribeAsr",
    "metadata": "<uuid>/demo-02/analysis/metadata/transcribeAsr" 
  }
}
Media2Cloud Implementation Guide  
Analysis State Machine Notification Message

```
{  "startTime": 1572431779467,  "endTime": 1572431780598,  "output": "<uuid>/demo-02/analysis/raw/20191030T103413/comprehend/keyphrase/output.json",  "metadata": "<uuid>/demo-02/analysis/metadata/keyphrase/output.json" },  "sentiment": {    "startTime": 1572431779533,    "endTime": 1572431780613,    "output": "<uuid>/demo-02/analysis/raw/20191030T103413/comprehend/sentiment/output.json",    "metadata": "<uuid>/demo-02/analysis/metadata/sentiment/output.json" },  "transcribe": {    "name": "<uuid>_19aceb4c20e2f0bc",    "output": "<uuid>/demo-02/analysis/raw/20191030T094613/transcribe/output.txt",    "startTime": 1572428776512,    "endTime": 1572428824917,    "vtt": "<uuid>/demo-02/analysis/vtt/transcribe/output.vtt" },  "image": {    "status": "NOT_STARTED" },  "document": {    "status": "NOT_STARTED" },  "metrics": {    "duration": 1000,    "requestTime": 1572428773178,    "startTime": 1572428774806,    "endTime": 1572428940375 },  "indexer": {    "terms": [      "celeb",      "faceMatch",      "label",      "entity",      "keyphrase",      "sentiment"    ]  }}
```

<table>
<thead>
<tr>
<th>Key Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>operation</td>
<td>Last state of the state machine: job-completed</td>
</tr>
<tr>
<td>status</td>
<td>Status of the state machine: COMPLETED</td>
</tr>
<tr>
<td>stateMachine</td>
<td>Analysis state machine name</td>
</tr>
<tr>
<td>uuid</td>
<td>UUID of the file</td>
</tr>
<tr>
<td>progress</td>
<td>Progress of the state machine</td>
</tr>
<tr>
<td>input</td>
<td>Input parameters</td>
</tr>
<tr>
<td>input.aiOptions</td>
<td>Indicate the AI/ML options used for the analysis process</td>
</tr>
<tr>
<td>Key Name</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>-----------------------------------------------------------------------------</td>
</tr>
<tr>
<td>input.aiOptions.celeb</td>
<td>Boolean value indicates if Celebrity detection is enabled</td>
</tr>
<tr>
<td>input.video</td>
<td>Video analysis information</td>
</tr>
<tr>
<td>input.video.arn</td>
<td>Video analysis state machine execution ARN</td>
</tr>
<tr>
<td>input.video.status</td>
<td>Status of the video analysis process</td>
</tr>
<tr>
<td>input.video.startTime</td>
<td>Start time of the video analysis process</td>
</tr>
<tr>
<td>input.video.endTime</td>
<td>End time of the video analysis process</td>
</tr>
<tr>
<td>input.video.rekognition</td>
<td>Amazon Rekognition</td>
</tr>
<tr>
<td>input.video.rekognition.celeb</td>
<td>Amazon Rekognition Celebrity Recognition result</td>
</tr>
<tr>
<td>input.video.rekognition.celeb.id</td>
<td>Amazon Rekognition Celebrity Recognition Job ID</td>
</tr>
<tr>
<td>input.video.rekognition.celeb.startTime</td>
<td>Start time of the Celebrity Recognition process</td>
</tr>
<tr>
<td>input.video.rekognition.celeb.endTime</td>
<td>End time of the Celebrity Recognition process</td>
</tr>
<tr>
<td>input.video.rekognition.celeb.output</td>
<td>Location of the raw results from Celebrity Recognition process stored in proxy bucket</td>
</tr>
<tr>
<td>input.video.rekognition.celeb.vtt</td>
<td>Location of the WebVTT tracks the solution generated stored in proxy bucket</td>
</tr>
<tr>
<td>input.video.rekognition.celeb.metadata</td>
<td>Location of the metadata tracks the solution generated stored in proxy bucket</td>
</tr>
<tr>
<td>input.video.rekognition.face</td>
<td>Amazon Rekognition Face Detection result</td>
</tr>
<tr>
<td>input.video.rekognition.faceMatch</td>
<td>Amazon Rekognition Face Search result</td>
</tr>
<tr>
<td>input.video.rekognition.label</td>
<td>Amazon Rekognition Label Detection result</td>
</tr>
<tr>
<td>input.video.rekognition.moderation</td>
<td>Amazon Rekognition Moderation Detection result</td>
</tr>
<tr>
<td>input.video.rekognition.person</td>
<td>Amazon Rekognition Person Pathing result</td>
</tr>
<tr>
<td>input.audio</td>
<td>Audio analysis results</td>
</tr>
<tr>
<td>input.audio.arn</td>
<td>Audio analysis state machine execution ARN</td>
</tr>
<tr>
<td>input.audio.status</td>
<td>Status of the audio analysis process</td>
</tr>
<tr>
<td>input.audio.startTime</td>
<td>Start time of the audio analysis process</td>
</tr>
<tr>
<td>input.audio.endTime</td>
<td>End time of the audio analysis process</td>
</tr>
<tr>
<td>input.audio.transcribe</td>
<td>Amazon Transcribe results</td>
</tr>
<tr>
<td>input.audio.transcribe.name</td>
<td>Amazon Transcribe Job name</td>
</tr>
<tr>
<td>input.audio.transcribe.output</td>
<td>Location of the raw results from Amazon Transcribe service stored in proxy bucket</td>
</tr>
<tr>
<td>input.audio.transcribe.startTime</td>
<td>Start time of the transcribe process</td>
</tr>
</tbody>
</table>
### Key Name | Description
--- | ---
`input.audio.transcribe.endTime` | End time of the transcribe process
`input.audio.transcribe.vtt` | Location of the WebVTT tracks the solution generated, stored in proxy bucket
`input.audio.comprehend` | Amazon Comprehend results
`input.audio.comprehend.entity` | Amazon Comprehend Entity Detection results
`input.audio.comprehend.entity.startTime` | Start time of Entity Detection process
`input.audio.comprehend.entity.endTime` | End time of Entity Detection process
`input.audio.comprehend.entity.output` | Location of the raw results from Entity Detection process, stored in proxy bucket
`input.audio.comprehend.entity.metadata` | Location of the metadata tracks the solution generated, stored in proxy bucket
`input.audio.comprehend.keyphrase` | Amazon Comprehend Key Phrases Detection results
`input.audio.comprehend.sentiment` | Amazon Comprehend Sentiment Detection results
`input.image` | Image analysis results
`input.metrics` | Overall statistic of analysis process
`input.indexer.terms` | A list of terms indexed to Amazon Elasticsearch Service cluster

### Labeling State Machine Notification Message

```json
{
   "operation": "job-completed",
   "input": {
      "dataset": {
         "manifestUri": "s3://<proxy-bucket>/ground-truth/<uuid>/demo-02/transcoded/aiml/20191016T141758.manifest",
         "templateUri": "s3://<proxy-bucket>/ground-truth/<uuid>/demo-02/transcoded/aiml/20191016T141758.liquid",
         "items": [
            "a0fa5d7e-2bba-ffaf-4113-0b0bacc8f98d"
         ]
      },
      "labeling": {
         "name": "<uuid>-20191016T141758",
         "taskStatus": "COMPLETED",
         "outputUri": "s3://<proxy-bucket>/ground-truth/<uuid>/demo-02/transcoded/aiml/<uuid>-20191016T141758/manifests/output/output.manifest",
         "faceIds": [
            "d60a57f4-c486-4541-80aa-9e20712dc999"
         ]
      }
   }
}
```
State Machine Error Notification Message

```
{
  "uuid": "<uuid>",
  "stateMachine": "SO0050-m2csolution-gt-labeling",
  "status": "ERROR",
  "errorMessage": "Cannot destructure property 'workerId' of 'undefined' or 'null'.",
  "input": {
    "uuid": "<uuid>
  }
}
```
<table>
<thead>
<tr>
<th>Key Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>uuid</td>
<td>UUID of the file</td>
</tr>
<tr>
<td>stateMachine</td>
<td>The state machine that generated the error</td>
</tr>
<tr>
<td>status</td>
<td>ERROR</td>
</tr>
<tr>
<td>errorMessage</td>
<td>The detailed error message</td>
</tr>
<tr>
<td>input</td>
<td>The input parameter used to start the state machine</td>
</tr>
</tbody>
</table>
Appendix H: Customizing the Solution

By default, the solution creates three Amazon S3 buckets: one that stores your uploaded files, one that stores the proxy file and the analysis results, and one that stores the static web assets. If you already have an Amazon S3 bucket where you store your original content, you can modify the AWS CloudFormation template to use your own bucket.

1. Download the source code from our GitHub repository.
2. Open the project with your IDE or text editor.
3. Open deployment/media2cloud-bucket-stack.template.
4. Under Mappings, UserDefined, Bucket, change the Ingest field to your bucket name.

```
UserDefined:
  Bucket:
    Ingest: "<your-ingest-bucket>"
```

5. Optionally, to use your own Amazon S3 bucket for storing proxies and analysis results, change the Proxy field to your bucket name. Ensure the proxy bucket is different from your Ingest bucket.
6. Save the template and follow the README instructions to build your package.

Considerations

Bucket Region

If you specify your own ingest (and/or proxy) buckets, ensure the bucket(s) are in the same AWS Region as the region where the solution is deployed to avoid cross-region data transfer costs.

Bucket CORS Settings

If you decide to use your own ingest (and/or proxy) bucket, the solution updates your bucket's CORS settings to allow the web interface to perform cross origin GET/POST requests. The following example shows the CORS settings:

```xml
<?xml version="1.0" encoding="UTF-8"?>
<CORSConfiguration xmlns="http://s3.amazonaws.com/doc/2006-03-01/">
  <CORSRule>
    <AllowedOrigin>https://<cf-id>.cloudfront.net</AllowedOrigin>
    <AllowedMethod>GET</AllowedMethod>
    <AllowedMethod>PUT</AllowedMethod>
    <AllowedMethod>POST</AllowedMethod>
    <AllowedMethod>HEAD</AllowedMethod>
    <AllowedMethod>DELETE</AllowedMethod>
    <MaxAgeSeconds>3000</MaxAgeSeconds>
    <ExposeHeader>Content-Length</ExposeHeader>
    <ExposeHeader>ETag</ExposeHeader>
    <ExposeHeader>x-amz-meta-uuid</ExposeHeader>
  </CORSRule>
</CORSConfiguration>
```
<ExposeHeader>x-amz-meta-md5</ExposeHeader>
<AllowedHeader>*</AllowedHeader>
</CORSRule>
</CORSConfiguration>
Appendix I: Collection of Operational Metrics

This solution includes an option to send operational metrics to AWS. We use this data to better understand how customers use this solution and related services and products. When enabled, the following information is collected and sent to AWS:

- **Solution ID**: The AWS solution identifier
- **Unique ID (UUID)**: Randomly generated, unique identifier for each deployment
- **Timestamp**: Media file upload timestamp
- **Format**: The format of the uploaded media file
- **Size**: The size of the file the solution processes
- **Duration**: The length of the uploaded video file

Note that AWS will own the data gathered via this survey. Data collection will be subject to the AWS Privacy Policy. To opt out of this feature, modify the AWS CloudFormation template, `media2cloud.template`, under the Parameters section as follows:

```
"AnonymousUsage:
    Default: "Yes"
```

to

```
AnonymousUsage:
    Default: "No"
```
Source Code

You can visit our GitHub repository to download the templates and scripts for this solution, and to share your customizations with others.
Contributors

The following individuals contributed to this document:

- Ken Shek
- Jack Wenzinger
- Christopher Kuthan
## Revisions

<table>
<thead>
<tr>
<th>Date</th>
<th>Change</th>
</tr>
</thead>
<tbody>
<tr>
<td>January 2019</td>
<td>Initial Release</td>
</tr>
<tr>
<td>March 2019</td>
<td>Modified JSON file descriptions</td>
</tr>
<tr>
<td>November 2019</td>
<td>Updated the analysis workflow engine and added support for ingesting and analyzing images</td>
</tr>
<tr>
<td>January 2021</td>
<td>Updated the list of AWS Technology Partners</td>
</tr>
</tbody>
</table>
Notices

Customers are responsible for making their own independent assessment of the information in this document. This document: (a) is for informational purposes only, (b) represents AWS current product offerings and practices, which are subject to change without notice, and (c) does not create any commitments or assurances from AWS and its affiliates, suppliers or licensors. AWS products or services are provided “as is” without warranties, representations, or conditions of any kind, whether express or implied. AWS responsibilities and liabilities to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

The Media2Cloud solution is licensed under the terms of the Apache License Version 2.0 available at The Apache Software Foundation.